TR 72-7

A P P E N D I X

```
      PROGRAM QUEUE(INPUT,OUTPUT)
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C          AUTHOR:                                                     C
C                CHESTER CHIA-ANN LEE                                  C
C                DEPARTMENT OF COMPUTER SCIENCE                        C
C                THE UNIVERSITY OF TEXAS AT AUSTIN                     C
C                AUSTIN, TEXAS, 78712                                  C
C                                                                      C
C    PROGRAM QUEUE EMPLOYS MARKOV PROCESS TO OBTAIN STEADY-STATE PRO-  C
C BABILITIES,THROUGHPUT,AVERAGE QUEUE LENGTH,AND STANDARD DEVIATION OF C
C AVG. QUE-LENGTH FOR EACH QUEUE IN A CLOSED,CYCLE QUEUING NETWORK     C
C WHERE SERVICE DISCIPLINE FOR EACH QUEUE IS FIRST-COME-FIRST-SERVED.  C
C EACH QUEUE IN THE NETWORK HAS ONE OR MORE SERVERS.  MEAN SERVICE     C
C TIME OF EACH SERVER IS EITHER EXPONENTIAL,HYPEREXPONENTIAL, OR HYPO- C
C EXPONENTIAL.  A JOB LEAVING QUEUE I MAY NOT JOIN QUEUE I AT ONCE.    C
C                                                                      C
C    LET:                                                              C
C ND = NO. OF QUEUES IN THE NETWORK                                    C
C NJ = NO. OF JOBS IN THE NETWORK                                      C
C NTYPE(I) = TYPE OF DEVICE(S) IN QUEUE I. VALUE OF NTYPE(I) IS 0 IF   C
C   MEAN SERVICE TIME OF QUEUE I IS EXPONENTIAL, 1 IF IT IS HYPEREX-   C
C   PONENTIAL, AND 2 IF IT IS HYPOEXPONENTIAL.                         C
C NST(I) = NO. OF BRANCHES OR STATIONS OF QUEUE I                      C
C NDV(I) = NO. OF DEVICES IN QUEUE I                                   C
C M(N1,S1,S2,..SK;N2,S1,S2,...SM;...;NI;...;NN,S1,...SN) = STATE WHERE C
C   NI IS THE NO. OF JOBS IN QUEUE I AND S1,S2,...SK ARE NOS. OF JOBS  C
C   IN STATIONS 1,2,..,K OF QUEUE I. NOTE THAT THERE IS NO SI IN QUEUE C
C   I IF QUEUE I IS EXPONENTIAL                                        C
C N = TOTAL NO. OF STATES                                              C
C P(I) = STEADY-STATE PROBABILITY OF STATE I                           C
C W(I,J) = PROBABILITY OF A JOB LEAVING QUEUE I FOR QUEUE J            C
C NF(I) = FIRST POSITION OF QUEUE I IN STATE M                         C
C NS(I) = NO.OF JOBS IN QUEUE I                                        C
C MT = TOTAL NO. OF STATIONS OR BRANCHES IN THE NETWORK                C
C U(I) = MEAN SERVICE TIME OF STATION I                                C
C V(I) = ARRIVAL PROBABILITY FOR STATION I                             C
C R(I) = MEAN SERVICE TIME OF QUEUE I                                  C
C UT(I,J) = PROBABILITY THAT THERE ARE I JOBS IN QUEUE J               C
C QL(I) = AVERAGE QUEUE LENGTH OF QUEUE I                              C
C SD(I) = STANDARD DEVIATION OF AVG. Q-LENGTH FOR QUEUE I              C
C TP(I) = THROUGHPUT FROM QUEUE I                                      C
C                                                                      C
C    NOTE:                                                             C
C        USER OF THIS PROGRAM IS RESPONSIBLE TO GIVE VALUES OF ND,NJ,  C
C      NTYPE(I),NST(I),NDV(I),W(I,J),U(I),V(I), AND MT.                C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
      COMMON M(210,10),NST(5),NTYPE(5),NF(5),NS(5),NDV(5),P(210),N,ND,
     1    NJ,MT,W(5,5),U(10),V(10),UT(8,5),QL(5),SD(5),TP(5),R(5)
C
C   READ IN DATA FOR THE NETWORK
C
      READ 10, ND,NJ,MT
   10 FORMAT(40I2)
      READ 10, (NST(I),I=1,ND)
```

```
      READ 10, (NTYPE(I),I=1,ND)
      READ 10, (NDV(I),I=1,ND)
  130 READ 140, ((W(I,J),J=1,ND),I=1,ND)
  140 FORMAT(10F8.4)
      READ 140, (U(I),I=1,MT)
      READ 140, (V(I),I=1,MT)
C
C   CALL SUBROUTINE FCFS FOR QUEUING ANALYSIS
C
      CALL FCFS
C
C   PRINT RESULTS
C
      NJJ=NJ+1
      PRINT 125
  125 FORMAT(1H1////32X,3HCPU,6X,5HI/O 1,5X,5HI/O 2,5X,5HI/O 3,5X,
     1     5HI/O 4//30X,5(7H-------,3X))
      PRINT 135, (NDV(I),I=1,ND)
  135 FORMAT(//10X,16HSERVERS IN QUEUE,7X,5(I1,9X))
      PRINT 942
  942 FORMAT(/10X,16HPROB. OF ARRIVAL/)
      DO 946 I=1,ND
      PRINT 945, (W(I,J),J=1,ND)
  945 FORMAT(31X,5(F5.3,5X))
  946 CONTINUE
      PRINT 948, (R(I),I=1,ND)
  948 FORMAT(//10X,17HMEAN SERVICE TIME,5(4X,F5.3,1X))
      PRINT 950
  950 FORMAT(//10X,11HUTILIZATION )
      DO 960 I=1,NJJ
      J=I-1
      PRINT 955, J,(UT(I,K),K=1,ND)
  955 FORMAT(12X,I1,4H JOB,9X,5(5X,F5.3))
  960 CONTINUE
      PRINT 965, (QL(I),I=1,ND)
  965 FORMAT(//10X,17HAVG. QUEUE LENGTH,5(4X,F5.3,1X))
      PRINT 980, (SD(I),I=1,ND)
  980 FORMAT(//10X,17HSTD. DEV. OF Q-LN,5(4X,F5.3,1X))
      PRINT 982, (TP(I),I=1,ND)
  982 FORMAT(//10X,10HTHROUGHPUT,10X,5(F7.2,3X))
      STOP
      END
```

```
      SUBROUTINE FCFS
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE FCFS GENERATES ALL STATES OF A GIVEN NETWORK,COMPUTES C
C TRANSITIONS BETWEEN ANY TWO STATES,AND COMPUTES STEADY-STATE PROBA-  C
C BILITIES OF EACH STATE.   DEVICE UTILIZATION,THROUGHPUT,AVG. QUEUE   C
C LENGTH.AND STANDARD DEVIATION OF AVG. Q-LENGTH OF EACH QUEUE IN THE  C
C NETWORK ARE ALSO COMPUTED.                                           C
C                                                                     C
C     VARIABLES USED ARE AS FOLLOWS:                                   C
C A(I,J) = TRANSITION RATE FROM STATE I TO STATE J                     C
C LIN    = INDEX OF QUEUE THAT A JOB ENTERS                            C
C LOUT   = INDEX OF QUEUE THAT A JOB LEAVES                            C
C KIN    = INDEX OF BRANCH OR STATION THAT A JOB ENTERS                C
C KOUT   = INDEX OF BRANCH OR STATION THAT A JOB LEAVES                C
C DIAG   = A(I,I)                                                      C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
      COMMON M(210,10),NST(5),NTYPE(5),NF(5),NS(5),NDV(5),P(210),N,ND,
     1    NJ,MT,W(5,5),U(10),V(10),UT(8,5),QL(5),SD(5),TP(5),R(5)
      DIMENSION A(210,210)
C
C    INITIALIZE N AND POSITION THE FIRST INDEX OF EACH QUEUE IN STATE M
C
      N=0
      LAST=ND-1
      NF(1)=1
      DO 25 I=2,ND
      IF (NTYPE(I-1) .EQ. 0) GO TO 20
      NF(I)=NF(I-1)+NST(I-1)+1
      GO TO 25
   20 NF(I)=NF(I-1)+NST(I-1)
   25 CONTINUE
      NT=NF(ND)+NST(ND)-1
      IF (NTYPE(ND) .GT. 0) NT=NT+1
C
C    GENERATE ALL POSSIBLE COMBINATIONS OF JOBS IN EACH QUEUE
C
      DO 30 I=1,LAST
   30 NS(I)=0
      IX=LAST
      IL=IX-1
   40 NN=0
      DO 50 J=1,LAST
   50 NN=NN+NS(J)
      NS(ND)=NJ-NN
C
C    CALL SUBROUTINE PERMUTE TO GENERATE POSSIBLE STATES FOR GIVEN NS(I)
C
      CALL PERMUTE
      IF (NS(1) .EQ. NJ) GO TO 100
      IF (NS(ND) .EQ. 0) GO TO 60
      NS(IX)=NS(IX)+1
      GO TO 40
   60 IX=IX-1
      IL=IX-1
```

```
      JL=0
      IF (IX .EQ. 1) GO TO 80
      DO 70 I=1,IL
   70 JL=JL+NS(I)
   80 IF (JL+NS(IX)+1 .GT. NJ) GO TO 60
      IR=IX+1
      DO 90 J=IR,LAST
   90 NS(J)=0
      NS(IX)=NS(IX)+1
      IX=LAST
      IL=IX-1
      GO TO 40
C
C   ALL POSSIBLE STATES ARE EXHUSTED
C
  100 CONTINUE
C
C   RE-POSITION OF ARRIVAL PROB. AND MEAN SERVICE RATE FOR EACH STATION
C
      DO 160 I=1,MT
  160 U(I)=1.0/U(I)
      JT=MT
      DO 180 I=1,ND
      IF (NTYPE(I) .EQ. 0) GO TO 180
      JT=JT+1
      J=JT
  170 IF (J .EQ. NF(I)) GO TO 180
      U(J)=U(J-1)
      V(J)=V(J-1)
      J=J-1
      GO TO 170
  180 CONTINUE
C
C   CONSTRUCT TRANSITION MATRIX
C
      DO 600 I=1,N
C
C   INITIALIZE MAIN DIAGONAL ELEMENT
C
      DIAG=0.0
      DO 560 J=1,N
      IF (I .EQ. J) GO TO 560
      KT=0
      PP=1.0
      DO 500 IC=1,ND
      ID=NF(IC)
      JC=ID+1
      JD=JC+NST(IC)-1
C
C   CHECK FOR NO. OF JOBS IN EACH QUEUE OF STATES I AND J
C
      IF (IABS(M(I,ID)-M(J,ID))-1) 200,300,550
C
C   NO. OF JOBS IN QUEUE IC OF STATE I EQUALS TO QUEUE IC OF STATE J
C
  200 IF (NTYPE(IC)-1) 500,210,250
C
C   SERVER IN QUEUE IC IS HYPEREXPONENTIALLY DISTRIBUTED
```

```
C
  210 DO 220 JE=JC,JD
      IF (M(I,JE) .NE. M(J,JE)) GO TO 550
  220 CONTINUE
      GO TO 500
C
C   SERVER IN QUEUE IC IS HYPOEXPONENTIALLY DISTRIBUTED
C
  250 IX=0
      JCC=JD-1
      DO 255 JF=JC,JCC
      IF (M(I,JF) .EQ. M(J,JF)) GO TO 255
      IF (M(I,JF)-M(J,JF) .EQ. 1 .AND. M(J,JF+1)-M(I,JF+1) .EQ. 1) GO
     1    TO 252
      GO TO 550
  252 IX=IX+1
      KOUT=JF
  255 CONTINUE
C
C   CHECK FOR TRANSITION BETWEEN STAGES
C
      IF (IX-1) 500,260,550
C
C   GENERATE TRANSITION RATE
C
  260 KT=KT+2
      LOUT=IC
      LIN=IC
      PP=PP*M(I,KOUT)*U(KOUT)
      GO TO 500
C
C   THERE IS A POSSIBLE TRANSITION BETWEEN STATES I AND J
C
  300 IF (M(I,ID) .GT. M(J,ID)) GO TO 400
C
C   A JOB LEAVING STATE J ENTERS STATE I
C
      IF (NTYPE(IC)-1) 310,330,360
C
C   JOB ENTERS EXPONENTIALLY DISTRIBUTED QUEUE
C
  310 KT=KT+1
      LIN=IC
      GO TO 500
C
C   ENTERING HYPER-EXP. DISTRIBUTED QUEUE
C
  330 IX=0
      DO 340 KA=JC,JD
C
C   CHECK FOR PERMISSIBILITY OF TRANSITION
C
      IF (M(I,ID) .LT. NDV(IC)) GO TO 335
      IF (M(I,KA) .NE. M(J,KA)) GO TO 550
      GO TO 340
  335 IF (M(I,KA) .GT. M(J,KA)) GO TO 550
      IF (M(J,KA)-M(I,KA)-1) 340,337,550
  337 IX=IX+1      $      KIN=KA
```

```
  340 CONTINUE
      LIN=IC
      KT=KT+1
      IF (IX-1) 500,350,550
  350 PP=PP*V(KIN)
      GO TO 500
C
C   JOB ENTERING HYPO-EXP. DISTRIBUTED QUEUE
C
  360 IF (M(J,JC)-M(I,JC)) 370,370,550
C
C   CHECK FOR PERMISSIBILITY OF TRANSITION
C
  370 JCC=JC+1
      DO 380 JF=JCC,JD
      IF (M(I,JF) .NE. M(J,JF)) GO TO 550
  380 CONTINUE
      KT=KT+1
      LIN=IC
      GO TO 500
C
C   A JOB LEAVING STATE I ENTERS STATE J
C
  400 IF (NTYPE(IC)-1) 410,430,470
C
C   COMPUTE TRANSITION RATE FOR EXPONENTIAL QUEUE
C
  410 LOUT=IC
      KT=KT+1
      IF (M(I,ID)-NDV(IC)) 415,420,420
  415 KK=M(I,ID)      $     GO TO 425
  420 KK=NDV(IC)
  425 PP=PP*U(ID)*KK
      GO TO 500
C
C   COMPUTE TRANSITION RATE FOR HYPEREXPONENTIAL QUEUE
C
  430 IF (M(I,ID) .GT. NDV(IC)) GO TO 440
      LX=0
      DO 435 JF=JC,JD
      IF (M(I,JF)-M(J,JF)) 550,435,432
  432 LX=LX+1      $     KX=JF
  435 CONTINUE
      IF (LX .NE. 1) GO TO 550
      KT=KT+1
      LOUT=IC
      PP=PP*U(KX)*M(I,KX)
      GO TO 500
  440 RATE=0.0
      LI=0      $     LJ=0
      DO 460 JF=JC,JD
      IF (IABS(M(I,JF)-M(J,JF))-1) 445,450,550
  445 IF (M(I,JF) .EQ. 0) GO TO 460
      IF (LI .NE. 0 .OR. LJ .NE. 0) GO TO 460
      RATE=RATE+U(JF)*V(JF)*M(I,JF)
      GO TO 460
  450 IF (M(I,JF) .LT. M(J,JF)) GO TO 455
      LI=LI+1
```

```
      IF (LI .GT. 1) GO TO 550
      KOUT=JF
      GO TO 460
  455 LJ=LJ+1
      IF (LJ .GT. 1) GO TO 550
      KIN=JF
  460 CONTINUE
      IF (LI .EQ. 0 .AND. LJ .EQ. 0) GO TO 465
      RATE=U(KOUT)*V(KIN)*M(I,KOUT)
  465 PP=RATE
      KT=KT+1
      LOUT=IC
      GO TO 500
C
C   COMPUTE TRANSITION RATE FOR HYPOEXPONENTIAL QUEUE
C
  470 IF (M(I,JD)-M(J,JD) .NE. 1) GO TO 550
      IF (M(J,JC)-M(I,JC) .GT. 1) GO TO 550
      JCC=JC+1      $      JDD=JD-1
      DO 475 JFF=JCC,JDD
  475 IF (M(I,JFF) .NE. M(J,JFF)) GO TO 550
      KT=KT+1
      LOUT=IC
      PP=PP*M(I,JD)*U(JD)
  500 CONTINUE
C
C   CHECK IF MORE THAN ONE TRANSITION HAS OCCURED FOR STATES I AND J
C
      IF (KT .NE. 2) GO TO 550
C
C   GENERATE ELEMENT A(I,J)
C
      IF (LOUT .NE. LIN) GO TO 525
      A(I,J)=PP      $      GO TO 540
  525 A(I,J)=PP*W(LOUT,LIN)
  540 CONTINUE
      DIAG=DIAG-A(I,J)
      GO TO 560
  550 A(I,J)=0.0
  560 CONTINUE
      A(I,I)=DIAG
C
C   REPLACE LAST COLUMN BY 1
C
      A(I,N)=1.0
  600 CONTINUE
C
C   INVERT TRANSITION MATRIX
C
      DO 800 L=1,N
      PIVOT=A(L,L)
C
C   CHECK TO SEE IF MAIN DIAGONAL ELEMENT IS ZERO
C
      IF (ABS(PIVOT) .LE. 1.0E-200) GO TO 985
      A(L,L)=0.0
      DO 700 I=1,N
      IF (I .EQ. L) GO TO 700
```

```
      A(I,L)=A(I,L)/PIVOT
      X=A(I,L)
      DO 650 J=1,N
  650 A(I,J)=A(I,J)-X*A(L,J)
  700 CONTINUE
      DO 750 J=1,N
  750 A(L,J)=-A(L,J)/PIVOT
      A(L,L)=1.0/PIVOT
  800 CONTINUE
      NJJ=NJ+1
      DO 910 KB=1,ND
      DO 905 KA=1,NJJ
  905 UT(KA,KB)=0.0
      QL(KB)=0.0
  910 CONTINUE
      DO 925 I=1,N
C
C   LAST ROW ELEMENTS ARE STEADY-STATE PROBABILITIES
C
      P(I)=A(N,I)
C
C   COMPUTE UTILIZATIONS OF EACH QUEUE
C
      DO 920 J=1,ND
      LA=NF(J)
      KS=M(I,LA)+1
      UT(KS,J)=UT(KS,J)+P(I)
  920 CONTINUE
  925 CONTINUE
C
C   COMPUTE AVERAGE QUEUE LENGTH OF EACH QUEUE
C
      DO 935 I=1,ND
      DO 930 J=1,NJJ
  930 QL(I)=QL(I)+(J-1)*UT(J,I)
  935 CONTINUE
C
C   COMPUTE THROUGHPUT OF EACH QUEUE
C
      DO 940 I=1,ND
      IA=NF(I)
      IF (NTYPE(I) .GT. 0) IA=IA+1
      IB=IA+NST(I)-1
      R(I)=0.0
      DO 937 J=IA,IB
  937 R(I)=R(I)+V(J)/U(J)
  940 CONTINUE
      DO 960 I=1,ND
      TP(I)=0.0
      DO 955 J=2,NJJ
      IF (J-1-NDV(I)) 945,945,950
  945 TP(I)=TP(I)+UT(J,I)*(J-1)/R(I)
      GO TO 955
  950 TP(I)=TP(I)+UT(J,I)*NDV(I)/R(I)
  955 CONTINUE
  960 CONTINUE
C
C   COMPUTE STANDARD DEVIATION OF AVG. Q-LENGTH OF EACH QUEUE
```

```
C
      DO 975 I=1,ND
      SD(I)=0.0
      DO 970 J=1,NJ
  970 SD(I)=SD(I)+UT(J+1,I)*J**2
      SD(I)=SQRT(SD(I)-QL(I)**2)
  975 CONTINUE
      GO TO 995
  985 PRINT 990, L
  990 FORMAT(//10X,15HPIVOT=0 AT LOOP,I4)
  995 CONTINUE
      RETURN
      END
```

```
      FUNCTION NFACTOR(N)
C                                            C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                            C
C   THIS FUNCTION COMPUTES FACTORIAL OF N    C
C                                            C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                            C
      IF (N-1) 10,10,20
   10 NFACTOR=1
      RETURN
   20 NFACTOR=1
      DO 30 I=2,N
   30 NFACTOR=NFACTOR*I
      RETURN
      END
```

```
      SUBROUTINE PERMUTE
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE PERMUTE CONSTRUCTS ALL POSSIBLE STATES FOR FIXED NO.  C
C OF JOBS IN EACH QUEUE.                                              C
C                                                                     C
C     VARIABLES USED ARE AS FOLLOWS:                                  C
C L(I,J) = TEMPORARY LOCATIONS FOR POSSIBLE STATES                    C
C IP(I)  = REMAINING NO. OF STATES                                    C
C LP(I)  = NO. OF POSSIBLE COMBINATIONS FOR GIVEN NO. OF JOBS IN I TH. C
C          QUEUE                                                      C
C IPP    = TOTAL NO. OF POSSIBLE STATES                               C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
      COMMON M(210,10),NST(5),NTYPE(5),NF(5),NS(5),NDV(5),P(210),N,ND,
     1    NJ,MT,W(5,5),U(10),V(10),UT(8,5),QL(5),SD(5),TP(5),R(5)
      DIMENSION L(10,10),IP(10),LP(10)
      IPP=1
      DO 70 I=1,ND
      NBGN=NF(I)
      LAST=NF(I)+NST(I)-1
      IF (NTYPE(I) .NE. 0) LAST=LAST+1
C
C    CHECK FOR ZERO JOB IN QUEUE
C
      IF (NS(I) .NE. 0) GO TO 20
   10 DO 15 J=NBGN,LAST
   15 L(1,J)=0
      LP(I)=1
      GO TO 65
C
C    CONSTRUCT DIFFERENT COMBINATIONS OF JOBS IN EACH STATION OF QUEUE
C
   20 IF (NTYPE(I)-1) 22,25,25
   22 L(1,NBGN)=NS(I)
      LP(I)=1
      GO TO 65
   25 IF (NS(I)-NDV(I)) 30,35,35
   30 MJ=NS(I)
      GO TO 40
   35 MJ=NDV(I)
   40 IA=1
      MA=NF(I)+1
      MB=LAST-1
      DO 42 IB=MA,MB
   42 L(IA,IB)=0
      IX=MB
      IL=IX-1
   45 MM=0
      DO 47 JA=MA,MB
   47 MM=MM+L(IA,JA)
      L(IA,NBGN)=NS(I)
      L(IA,LAST)=MJ-MM
      IF (L(IA,MA) .EQ. MJ) GO TO 64
      IF (L(IA,LAST) .EQ. 0) GO TO 55
   50 IA=IA+1
```

```
      DO 52 IB=NBGN,IL
   52 L(IA,IB)=L(IA-1,IB)
      L(IA,IX)=L(IA-1,IX)+1
      GO TO 45
   55 IX=IX-1
      IL=IL-1
      JL=0
      IF (IX .EQ. MA) GO TO 59
      DO 57 IB=MA,IL
   57 JL=JL+L(IA,IB)
   59 IF (JL+L(IA,IX)+1 .GT. MJ) GO TO 55
      IR=IX+1
      DO 60 JA=IR,MB
   60 L(IA+1,JA)=0
      IF (IX .EQ. MA) GO TO 62
      DO 61 JB=NBGN,IL
   61 L(IA+1,JB)=L(IA,JB)
   62 L(IA+1,IX)=L(IA,IX)+1
      IX=MB
      IL=IX-1
      IA=IA+1
      GO TO 45
   64 LP(I)=IA
   65 IPP=IPP*LP(I)
   70 CONTINUE
C
C     GENERATE ALL POSSIBLE STATES
C
      NB=N+1
      N=N+IPP
      IP(1)=IPP/LP(1)
      DO 75 I=2,ND
   75 IP(I)=IP(I-1)/LP(I)
      DO 95 J=1,ND
      JX=1
      KX=1
      KA=NF(J)
      KB=KA+NST(J)-1
      IF (NTYPE(J) .GT. 0) KB=KB+1
      DO 90 K=NB,N
      IF (KX .LE. IP(J)) GO TO 80
      KX=1
      JX=JX+1
      IF (JX .LE. LP(J)) GO TO 80
      JX=1
   80 DO 85 KK=KA,KB
   85 M(K,KK)=L(JX,KK)
      KX=KX+1
   90 CONTINUE
   95 CONTINUE
      RETURN
      END
```

```
      PROGRAM LOCOBAL(INPUT,OUTPUT)
C                                                                        C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                        C
C     AUTHOR:                                                            C
C              CHESTER CHIA-ANN LEE                                      C
C              DEPT. OF COMPUTER SCIENCE                                 C
C              UNIVERSITY OF TEXAS AT AUSTIN                             C
C              AUSTIN, TEXAS                                             C
C                                                                        C
C    PROGRAM LOCOBAL EMPLOYS LOCAL BALANCE EQUATIONS TO SOLVE QUEUING    C
C NETWORK PROBLEMS.   IT GENERATES STATES FOR THE NETWORK, OBTAINS THE   C
C BTERMS OF ALL NODES IN THE NETWORK VIA TOPOLOGICAL SORT, COMPUTES      C
C STEADY-STATE PROBABILITIES OF ALL STATES, UTILIZATIONS,AVERAGE QUEUE   C
C LENGTHS,STANDARD DEVIATIONS OF AVG. QUE-LENGTHS, AND THROUGHPUTS OF    C
C ALL QUEUES IN THE NETWORK.                                             C
C                                                                        C
C    VARIABLES ARE AS FOLLOWS:                                           C
C INPUT(I,J)= INDEX OF NODES FOR A PATH IN THE NETWORK.  NODES ARE       C
C             NUMBERED FROM 1 TO N.   SOURCE NODE IS 0 AND SINK NODE     C
C             IS N+1. NO SOURCE NODE MAY APPEAR IN INPUT(I,1) AND NO     C
C             SINK NODE MAY APPEAR IN INPUT(I,2)                         C
C                                                                        C
C               *NOTE*  NODES WITHIN A QUEUE MUST NUMBERER IN            C
C                       SUCCESSIVE ORDER                                 C
C                                                                        C
C TRN(I)     = NON-ZERO PROBABILITY FROM INPUT(I,1) TO INPUT(I,2)        C
C NODE       = NO OF NODES IN THE NETWORK INCLUDING SOURCE AND SINK      C
C NJOB       = NO OF JOBS IN THE SYSTEM                                  C
C NO         = NO OF QUEUES IN THE SYSTEM                                C
C MM(I,J)    = STATE (N(1),N(2),...,N(NODE)) WHERE N(K) IS NO OF JOBS IN C
C              NODE K                                                    C
C NST(I)     = NO OF STAGES OR BRANCHES IN QUEUE I                       C
C NB(I)      = INDEX OF FIRST POSITION OF QUEUE I IN STATE M(J,I)        C
C NJ(I)      = NO OF JOBS IN QUEUE I                                     C
C K          = TOTAL NO OF STATES                                        C
C LJ         = NJOB+1                                                    C
C P(I)       = STEADY-STATE PROBABILITY OF STATE I                       C
C PTERM(I)   = MEAN SERVICE TIME OF NODE I                               C
C BTERM(I)   = PROBABILITY OF A JOB TRAVERSE FROM SOURCE TO NODE I       C
C SP(I)      = MEAN SERVICE TIME OF QUEUE I                              C
C UT(I,J)    = PROBABILITY OF I-1 JOBS IN QUEUE J                        C
C QL(I)      = AVERAGE QUEUE LENGTH OF QUEUE I                           C
C SD(I)      = STANDARD DEVIATION OF AVG. Q-LNTH OF QUEUE I              C
C TP(I)      = THROUGHPUT OF QUEUE I                                     C
C NDV(I)     = NO OF SERVERS IN QUEUE I                                  C
C INDEX      = 0 IF NETWORK CONTAINS A LOOP                              C
C              1 IF NETWORK IS FREE OF LOOP                              C
C NEDGE      = NO OF PATHS IN THE NETWORK                                C
C                                                                        C
C     *NOTE*  THIS PROGRAM APPLIES TO NETWORK WITH SINGLE SERVER         C
C                 QUEUES ONLY                                            C
C                                                                        C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                        C
      COMMON MM(1000,10),P(1000),PTERM(10),BTERM(10),UT(10,10),QL(10),
     1   SD(10),TP(10),NJ(10),NB(10),NST(10),SP(10),NJOB,NODE,NQ,K
      DIMENSION  INPUT(100,2),TRN(100)
```

```
C
C    READ IN DATA AND STORE NETWORK PATHS
C
       READ 10, NODE,NJOB,NQ
    10 FORMAT(40I2)
       READ 10, (NST(I),I=1,NQ)
       READ 20, (SP(I),I=1,NQ)
       READ 20, (PTERM(I),I=1,NODE)
    20 FORMAT(10F8.3)
       NEDGE=0
    30 READ 40, LA,LB,PC
    40 FORMAT(2I5,F5.3)
       IF (LA .EQ. 99999) GO TO 45
       NEDGE=NEDGE+1
       INPUT(NEDGE,1)=LA
       INPUT(NEDGE,2)=LB
       TRN(NEDGE)=PC
       GO TO 30
    45 INDEX=0
C
C    CALL FOR TOPOLOGICAL SORTING
C
       CALL TOPOSRT(INPUT,TRN,NEDGE,INDEX)
C
C    CHECK TO SEE IF NETWORK CONTAINS A LOOP
C
       IF (INDEX .EQ. 1) GO TO 100
C
C    CALL FOR GENERATING STATES
C
       CALL NSTATES
C
C    CALL FOR COMPUTING RESULTS
C
       CALL PSTATES
   100 CONTINUE
       STOP
       END
```

```
      SUBROU'INE TOPOSRT(INPUT,TRN,NEDGE,INDEX)
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     THIS SUBROUTINE OBTAINS BTERM BY USING TOPOLOGICAL SORTING OF THE C
C  NODES IN THE NETWORK AND IS A FORTRAN VERSION OF THE PROGRAM DONE BY C
C  KNUTH (THE ART OF COMPUTER PROGRAMMING,VOL.1,PP264-265).   THE ARRAY C
C  LINK(I) STORES ORDERED LIST OF NODES IN THE NETWORK.                C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
      COMMON MM(1000,10),P(1000),PTERM(10),BTERM(10),UT(10,10),QL(10),
     1     SD(10),TP(10),NJ(10),NB(10),NST(10),SP(10),NJOB,NODE,NQ,K
      INTEGER COUNT(20),TOP(20),QLINK(20),SUC(40),NEXT(40),R,F
      DIMENSION INPUT(100,2),TRN(100),LINK(20)
      NP=NODE+1
      LIMIT=99999
      DO 10 I=1,NODE
      QLINK(I)=0
      COUNT(I)=0
      TOP(I)=LIMIT
   10 CONTINUE
      QLINK(NP)=0
      DO 15 J=1,NEDGE
      INPUT(J,1)=INPUT(J,1)+1
      INPUT(J,2)=INPUT(J,2)+1
   15 CONTINUE
      N=NODE
      DO 20 K=1,NEDGE
      KA=INPUT(K,1)
      KB=INPUT(K,2)
      COUNT(KB)=COUNT(KB)+1
      SUC(K)=KB
      NEXT(K)=TOP(KA)
      TOP(KA)=K
   20 CONTINUE
      R=0
      NN=0
      DO 25 K=1,NODE
      IF (COUNT(K) .NE. 0) GO TO 25
      QLINK(R+1)=K
      R=K+1
   25 CONTINUE
      F=QLINK(1)
   30 NN=NN+1
      LINK(NN)=F
      IF (F .EQ. 0) GO TO 50
      N=N-1
      L=TOP(F)
   35 IF (L .EQ. LIMIT) GO TO 45
      KC=SUC(L)
      COUNT(KC)=COUNT(KC)-1
      IF (COUNT(KC) .NE. 0) GO TO 40
      QLINK(R)=KC
      R=KC+1
   40 L=NEXT(L)
      GO TO 35
   45 F=QLINK(F+1)
```

```
      GO TO 30
   50 IF (N .EQ. 0) GO TO 60
      INDEX=1
      PRINT 55
   55 FORMAT(/////5X,*NETWORK CONTAINS A LOOP*)
      RETURN
   60 CONTINUE
      BTERM(1)=1.0
      DO 120 I=2,NODE
      II=LINK(I)
      BTERM(II)=0.0
      DO 110 J=1,NEDGE
      IF (II .NE. INPUT(J,2)) GO TO 110
      JJ=INPUT(J,1)
      BTERM(II)=BTERM(II)+BTERM(JJ)*TRN(J)
  110 CONTINUE
  120 CONTINUE
      DO 130 I=1,NEDGE
      INPUT(I,1)=INPUT(I,1)-1
      INPUT(I,2)=INPUT(I,2)-1
  130 CONTINUE
      NODE=NODE-1
      DO 140 I=1,NODE
  140 LINK(I)=LINK(I+1)-1
      NODE=NODE-1
      PRINT 150, (((INPUT(I,J),J=1,2),TRN(I)),I=1,NEDGE)
  150 FORMAT(1H1/////10X,*NETWORK FLOW*//5X,4HNODE,2X,2HTO,2X,4HNODE,2X,
     1     *TRANSITION*//(5X,I3,7X,I3,6X,F5.3/))
      PRINT 160, (LINK(I),I=1,NODE)
  160 FORMAT(////5X,*ORDERED LINK LIST OF QUEUING NETWORK*//(5X,25I5/))
      DO 170 J=1,NODE
  170 BTERM(J)=BTERM(J+1)
      PRINT 180, ((I,BTERM(I)),I=1,NODE)
  180 FORMAT(///(5X,*BTERM*,*(*,I3,*) =*,F7.4))
      RETURN
      END
```

```
      SUBROUTINE NSTATES
C                                                                          C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                          C
C     THIS SUBROUTINE AUTOMATICALLY GENERATES ALL POSSIBLE STATES OF A    C
C QUEUING NETWORK.                                                         C
C     THE INDICES OF A STATE IS:  (1,2,...,IL,IX,IR,...,LAST,NODE)        C
C                                                                          C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                          C
      COMMON MM(1000,10),P(1000),PTERM(10),BTERM(10),UT(10,10),QL(10),
     1    SD(10),TP(10),NJ(10),NB(10),NST(10),SP(10),NJOB,NODE,NQ,K
      K=1
      LAST=NODE-1
      IX=LAST
      IL=IX-1
      DO 10 I=1,LAST
   10 MM(K,I)=0
   15 NF=0
      DO 20 J=1,LAST
   20 NF=NF+MM(K,J)
      MM(K,NODE)=NJOB-NF
      IF (MM(K,1) .EQ. NJOB) GO TO 50
      IF (MM(K,NODE) .EQ. 0) GO TO 35
   25 K=K+1
      DO 30 I=1,IL
   30 MM(K,I)=MM(K-1,I)
      MM(K,IX)=MM(K-1,IX)+1
      GO TO 15
   35 IX=IX-1
      IL=IX-1
      JL=0
      IF (IX .EQ. 1) GO TO 43
      DO 40 I=1,IL
   40 JL=JL+MM(K,I)
   43 IF (JL+MM(K,IX)+1 .GT. NJOB) GO TO 35
      IR=IX+1
      DO 45 J=IR,LAST
   45 MM(K+1,J)=0
      IF (IX .EQ. 1) GO TO 48
      DO 47 JJ=1,IL
   47 MM(K+1,JJ)=MM(K,JJ)
   48 MM(K+1,IX)=MM(K,IX)+1
      IX=LAST
      IL=IX-1
      K=K+1
      GO TO 15
   50 PRINT 55
   55 FORMAT(/////5X,*QUEUING NETWORK STATES*)
      DO 65 I=1,K
      PRINT 60, I, (MM(I,J),J=1,NODE)
   60 FORMAT(5X,1H(,I3,1H),30I4/)
   65 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE PSTATES
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     THIS SUBROUTINE COMPUTES STEADY-STATE PROBABILITIES,UTILIZATIONS, C
C MEAN QUEUE LENGTHS,STANDARD DEVIATIONS OF AVG. Q-LNTHS,AND THROUGH-  C
C PUTS OF ALL QUEUES IN THE NETWORK.                                   C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
      COMMON MM(1000,10),P(1000),PTERM(10),BTERM(10),UT(10,10),QL(10),
     1    SD(10),TP(10),NJ(10),NB(10),NST(10),SP(10),NJOB,NODE,NQ,K
C
C    INITIALIZING
C
      PRINT 7
    7 FORMAT(1H1////32X,3HCPU,6X,5HI/O 1,5X,5HI/O 2,5X,5HI/O 3,5X,
     1  5HI/O 4//30X,5(7H-------,3X))
      LJ=NJOB+1
      NB(1)=1
      DO 20 I=2,NQ
   20 NB(I)=NB(I-1)+NST(I-1)
      DO 50 I=1,NQ
      DO 40 J=1,LJ
   40 UT(J,I)=0.0
      QL(I)=0.0
      SD(I)=0.0
      TP(I)=0.0
   50 CONTINUE
C
C    COMPUTE STEADY-STATE PROBABILITIES
C
      G=0.0
      DO 70 I=1,K
      P(I)=1.0
      DO 65 II=1,NQ
      NJ(II)=0
      MA=NB(II)      $      MB=MA+NST(II)-1
      IF (MA .NE. MB) GO TO 55
      NJ(II)=NJ(II)+MM(I,MA)
      P(I)=P(I)*(BTERM(MA)*PTERM(MA))**NJ(II)
      GO TO 65
   55 DO 60 J=MA,MB
      NJ(II)=NJ(II)+MM(I,J)
      P(I)=P(I)*((PTERM(J)*BTERM(J))**MM(I,J)/NFACTOR(MM(I,J)))
   60 CONTINUE
      P(I)=P(I)*NFACTOR(NJ(II))
   65 CONTINUE
      G=G+P(I)
   70 CONTINUE
C
C    COMPUTE PROBABILITY FOR JOBS IN QUEUE, AVG. Q-LENGTH, AND STANDARD
C      DEVIATION FOR EACH QUEUE
C
      DO 80 I=1,K
      P(I)=P(I)/G
      DO 75 J=1,NQ
      NJ(J)=0
```

```
      MA=NB(J)          $       MB=MA+NST(J)-1
      DO 72 JA=MA,MB
   72 NJ(J)=NJ(J)+MM(I,JA)
      JJ=NJ(J)+1
      UT(JJ,J)=UT(JJ,J)+P(I)
   75 CONTINUE
   80 CONTINUE
      DO 90 I=1,NQ
      DO 85 J=1,LJ
      QL(I)=QL(I)+(J-1)*UT(J,I)
      SD(I)=SD(I)+UT(J,I)*(J-1)**2
   85 CONTINUE
      SD(I)=SQRT(SD(I)-QL(I)**2)
   90 CONTINUE
C
C    COMPUTE THROUGHPUT FOR EACH QUEUE
C
      DO 95 I=1,NQ
      TP(I)=TP(I)+(1.0-UT(1,I))/SP(I)
   95 CONTINUE
      PRINT 100, (SP(I),I=1,NQ)
  100 FORMAT(//10X,17HMEAN SERVICE TIME,5(4X,F5.3,1X))
      PRINT 105
  105 FORMAT(//10X,11HUTILIZATION)
      DO 115 I=1,LJ
      J=I-1
      PRINT 110, J,(UT(I,II),II=1,NQ)
  110 FORMAT(12X,I1,4H JOB,9X,5(5X,F5.3))
  115 CONTINUE
      PRINT 120, (QL(I),I=1,NQ)
  120 FORMAT(//10X,17HAVG. QUEUE LENGTH,5(4X,F5.3,1X))
      PRINT 125, (SD(I),I=1,NQ)
  125 FORMAT(//10X,14HSTD. DEVIATION,2X,5(5X,F5.3))
      PRINT 150, (TP(I),I=1,NQ)
  150 FORMAT(//10X,10HTHROUGHPUT,6X,5(5X,F5.2))
      RETURN
      END
```