DEFINITIONS AND PROCEDURES EMPLOYED

in the

GERT SIMULATION PROGRAM

Research Sponsored
by
Electronics Research Center
National Aeronautics and Space Administration
NASA Research Grant NGR 03-001-034

A. Alan B. Pritsker

ABSTRACT

This report describes the variables and techniques of a computer program for simulating GERT networks on a digital computer. This GERT simulation program is written in the GASP IIA simulation language and can be run on any computer with a FORTRAN compiler.

The GERT simulation program can accommodate GERT networks which have EXCLUSIVE-OR, INCLUSIVE-OR and AND logical operations associated with the input side of a node, and deterministic or probabilistic operations associated with the output side of a node. The branches of the GERT network are described in terms of a probability that the branch is realized and a time to perform the activity represented by the branch. (Time is used throughout this report to represent an additive variable.) The time associated with a branch can be a random variable.

The results obtained from the GERT simulation program are:

(1) The probability that a node is realized;

(2) The average time to realize a node;

(3) An estimate of the standard deviation of the time to realize a node;

(4) The minimum time observed to realize a node;

(5) The maximum time observed to realize a node; and

(6) A histogram of the times to realize a node.

Normally the above information is obtained for each sink node of the network. The program is written to permit the above information to be obtained for any node specified in the input data. A companion report is available which describes in detail the required input data.

The program has been exercised on the IBM 1130 which has a FORTRAN IV operation set more restrictive than the general FORTRAN IV operation set. Therefore, the GERT simulation program can be used on most computers which have FORTRAN IV compilers. On the IBM 1130 with 8192 words of core storage, GERT networks with up to 30 nodes and 40 branches can be simulated.

The program described in this report has been submitted to COSMIC.

DEFINITIONS AND PROCEDURES EMPLOYED IN THE GERT SIMULATION PROGRAM

The GERT simulation program is a general purpose program for simulating GERT networks. The input to the program is a description of the network in terms of its nodes and branches along with control information for setting up the simulation conditions. A detailed description of the input information is described in a companion report. (18)

The GERT simulation program is written in the GASP IIA simulation language. The techniques employed for simulating GERT networks are the standard techniques employed by the GASP IIA system. These techniques will not be described in this report. This report will describe the parts of the program which are special to GERT network simulation. In Appendix A, the GASP IIA subprograms and variables used within the GERT simulation program are defined. In Appendix B, the FORTRAN listings of the GASP IIA subprograms modified for use in the GERT simulation program are given. Further information on GASP IIA is contained in references (22, 23).

The GERT simulation program receives first the network description and the number of simulations to be performed. It then performs a simulation by selecting branches to be traversed based on random numbers and branch times from samples of the distributions inserted as input information. Statistics are automatically collected on nodes as specified by the user of the program. When all simulations as requested by the user have been completed, a summary report which describes the network simulated

and the final statistical results are printed. The final results include:

(1) The probability that a specified node is realized;

(2) The average time to realize the specified node;

(3) An estimate of standard deviation of the time to realize the specified node;

(4) The minimum time observed to realize the specified node;

(5) The maximum time observed to realize the specified node; and

(6) A histogram of the times to realize the specified node.

The current simulation program takes advantage of the concepts included in GASP IIA to obtain an efficient utilization of storage space and a relatively fast execution time. Desirable features from earlier versions of the program (1,14) for simulating GERT networks have been incorporated. The GERT simulation program reported herein is a general purpose network simulation program which has been tested for numerous examples.

Description of Overall Program Operation

The GERT simulation program performs a simulation by advancing time from event to event. In simulation parlance this is termed a next event simulation. The events associated with a GERT network are:

(1) Start of the simulation and (2) End of an activity.

The start event causes all source nodes to be realized and schedules the activities emanating from the source nodes according to the output type of the source node. The output type for all nodes is either deterministic or probabilistic. In the former case, all activities emanating from the node are scheduled and in the latter case, only one of the activities emanating from the node is scheduled. By scheduling an

activity is meant that an event "end of activity" is caused to occur at some future point in time. A next event simulation proceeds from event to event until the conditions which indicate that the simulation is completed are obtained. For the simulation of GERT networks, all events involve end of activity times and the simulation of the network proceeds from one end of activity time to another.

As part of the input data, the number of releases required to realize a node is specified. Each time an end of activity event occurs, the number of releases for the end node of that activity is decreased by one. When the number of releases remaining is zero, the node is realized and the activities emanating from the node are scheduled. Again, the number of activities scheduled depends on the output type for the node. End of activity events are put in an event file in chronological order. Since feedback branches are permitted, it is possible that a node already realized will be released. When this occurs, a check is made to determine if the node can be realized again. If it can, the activities emanating from the node are scheduled according to the output operation for the node. The assumption is made that it only requires one release in order to realize a node after it has been realized once.

As indicated above, the end of activity events are stored in chronological order in an event file. The events are removed from the event file one at a time and at each removal instant, tests are performed to determine if a node is realized. If no node is realized the next event is removed from the event file. If a node is realized, activities

are scheduled and the simulation is continued. The simulation ends when a prescribed number of sink nodes have been realized. The sink nodes can be either terminal nodes of the network or nodes on which statistics are being collected. As part of the input data, the node numbers on which statistics are collected and the number of nodes required to realize the network are defined.

The above process describes one simulation of a network. The program is written to allow multiple simulations to be performed. The number of simulation runs to be performed is part of the input data. The GERT simulation program automatically initializes the pertinent variables in order that consecutive simulations of the same network can be performed and, if desired, permits simulations of different networks to be performed consecutively.

The heart of the GERT simulation program is a filing system which contains an event file for all end of activity events scheduled to occur, and a file for each node of the set of activities emanating from that node. Thus, if a network contains 20 nodes there will be 21 files. File 1 is always the event file. File J for J>1, contains the set of activities emanating from node J. (Thus all node numbers must be greater than 1)

The filing system is designed so that all locations of the file can store entries inserted into the file. This permits a significant reduction in storage requirements since space need not be allocated to each file. In addition, a limit need not be set on the number of activities emanating from a particular node. By providing a file for all activities emanating from a node, no searching need be done to find the activities

which emanate from a node. This significantly reduces execution time for the GERT simulation program over past simulation programs (1, 14).

The file is divided into two parts: a fixed point array called NSET and a floating point array called QSET. Each entry in the file has associated with it three fixed point attributes and one floating point attribute. To insert an entry into a file, the buffer storage vectors JTRIB and ATRIB dimensioned to 3 and 1 respectively are used. The values to be given to these buffer storage vectors for the two types of files is shown below.

File No.	Description	JTRIB(1)	JTRIB(2)	JTRIB(3)	ATRIB(1)
1	Event file	End node no.	Not used	Not used	End of activity time
J	Set of activities emanating from node J	End node no. of activity	Parameter set associated with activity	Distribution type associated with activity	Cumulative probability of taking this activity or an activity emanating from node J stored prior to this activity

The start node number of an activity need not be stored in the file since the file number describes the start node number. The parameter set associated with an activity specifies the row number in the array PARAM(I,J) where the parameters associated with the distribution describing the time required to perform the activity are stored. The distribution

type associated with an activity specifies which subroutine is used for obtaining the time to perform the activity. Four distribution types have been included in the IBM 1130 version of the GERT simulation program. These are: (1) constant (2) normal (3) uniform (4) Erlang. The parameters required in the parameter set for each of these distribution types is given below.

Distribution Types	Parameters			
	1	2	3	4
1. Constant	Constant	---	---	---
2. Normal	Mean	Minimum	Maximum	Standard Deviation
3. Uniform	---	Minimum	Maximum	---
4. Erlang	Mean/k	Minimum	Maximum	k

The GERT simulation program has been written to facilitate the adding of distribution types. FORTRAN subroutines for sampling from other distributions are given in references (15, 23).

Definitions of Non-GASP Variables

The variables which are in COMMON in the GERT simulation program but are not a part of GASP IIA are given below.

NN	Largest node number in the network (smallest node number is 2)
NSKS	Number of sink nodes in the network (maximum of 5)
NSRC	Number of source nodes in the network
NSKST	Number of sink nodes to be realized to realize the network
NSKSR	Number of sink nodes yet to be realized to realize the network for a simulation run

NSORC(J) The node number of source node J, J=1, ···, NSRC

NSINK(K) The node number of sink node K, K=1, ···, NSKS

NRELP(I) Number of branches necessary to realize node I (If NRELP(I)>1, I is an AND node)

NREL(I) Number of branches yet to be realized to realize node I

NTYPE(I) If NTYPE(I) = 1, I has a deterministic output side and once realized does not release activities emanating from it.
2, I has a probabilistic output side and once realized does not release activities emanating from it.
3, I has a deterministic output side and can be realized many times. (Once realized it only takes one release for it to be realized again)
4, I has a probabilistic output side and can be realized many times.

XLOW(K) Lower limit for histogram for sink node K, K=1, ···, NSKS

WIDTH(K) Width of a cell for histogram of sink node K, K=1, ···, NSKS

The definitions of the GASP IIA variables are given in Appendix A.

The GERT Simulation Program

The GERT simulation program consists of a main program and three specialized subroutines. These subroutines are:

(1) EVNTS which is called when an end of activity event occurs;

(2) SCHAT, the scheduling of end of an activity event when a node is realized; and

(3) SAMPL, a subroutine for obtaining a deviate from a specified distribution with specified parameters.

The GASP subroutine SUMRY has been modified and provides a description and the activity parameters for the network in addition to the final results of the simulation.

The Main Program

A flow chart of the main program is given in Fig. 1. The main program first defines the equipment numbers for the card reader and printer. These variables are NCRDR and NPRNT. For the IBM 1130 system the values are 2 and 3 respectively. All read and write statements reference the variable names. The non-GASP variables are then initialized as described in reference (18). The first six data cards represent the initialization of the non-GASP variables. The main program then calls subroutine GASP which controls the simulation until all the runs requested have been completed.

A general flow chart of the functions performed by subroutine GASP and the GASP subprograms it calls is shown in Fig. 2. Subroutine GASP has not been rewritten for the GERT simulation program and is a general subroutine included in the GASP IIA programming language. The GASP subroutine has the GASP variables initialized in GASP IIA subroutine DATAN. An echo check of the parameters and files associated with the simulation programs are then printed. This provides a convenient check that all input data for a given network has been inserted into the computer. GASP then starts and controls the simulation. The first event is removed from the event file and current time is advanced to the time of the event. In the GERT simulation program, the initial event is always an event which releases the source nodes. GASP then calls subroutine EVNTS which releases nodes, schedules end of activity events, and determines when a complete simulation of a network has been accomplished. Subroutine events also passes this information back to subroutine GASP when all simulations of the network have been made. GASP then calls

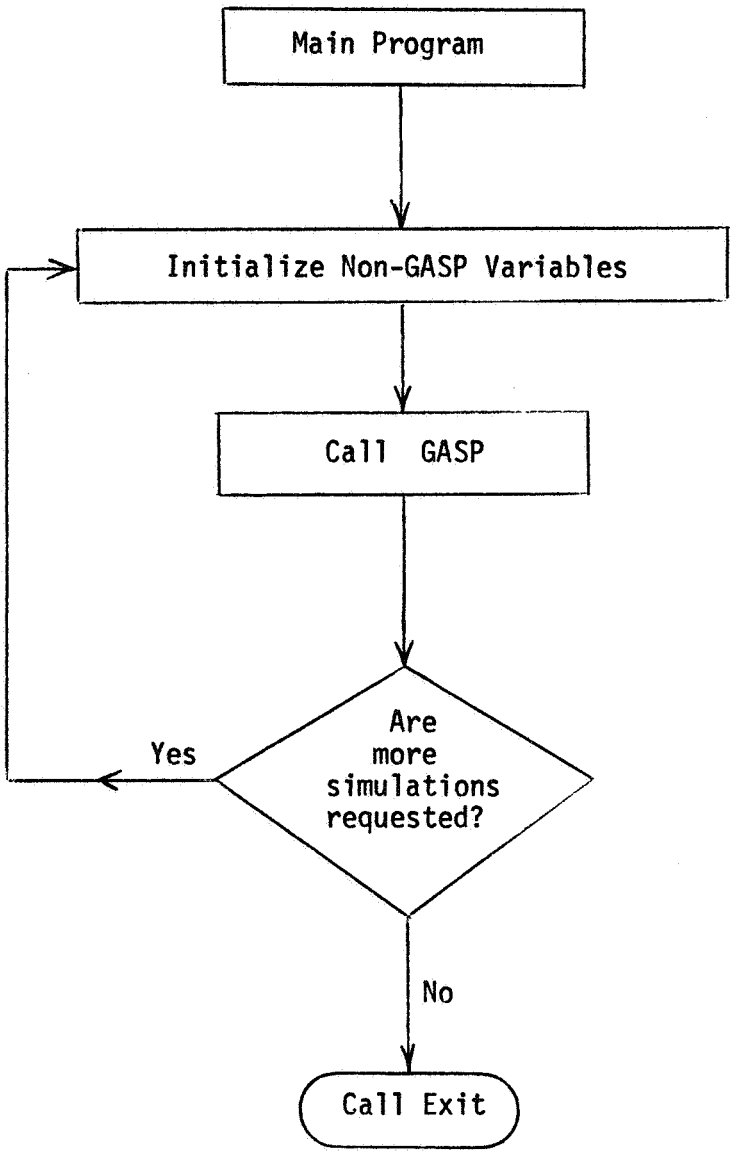

Fig. 1 Flow Chart of the Main GERT Simulation Program

subroutine SUMRY which prints the final GERT summary report. GASP also calls subroutine OTPUT which is a dummy subroutine which the programmer can use to print additional information. Subroutine OTPUT currently contains only a return statement. Since subroutine GASP calls it, an OTPUT subroutine is required.

If all simulations of the network have not been made, the next end of activity event is removed from the event file and the simulation continued through the loop shown in Fig. 2.

The FORTRAN listing for the main program, subroutine SUMRY and subroutine OTPUT are given in Figs.3, 4, and 5. Subroutine GASP is given in Appendix B along with the other GASP IIA subprograms.

Subroutine EVNTS(NEND,NSET,QSET)

The argument NEND defines the end node of the activity which has just been completed. NSET and QSET are vectors which are included in arguments to all subprograms to facilitate the changing of the size of the vectors without recompiling all subprograms. This is critical since NSET and QSET are used to store the integer and real portions of the filing array respectively.

The flow chart for subroutine EVNTS is given in Fig. 6. First a test is made to determine if this is the start of a new simulation run. This occurs if NEND is zero in which case activities from all source nodes are scheduled in accordance with the node type of the source node. The scheduling of the end of an activity event is accomplished by calling subroutine SCHAT with NEND as one of its arguments.

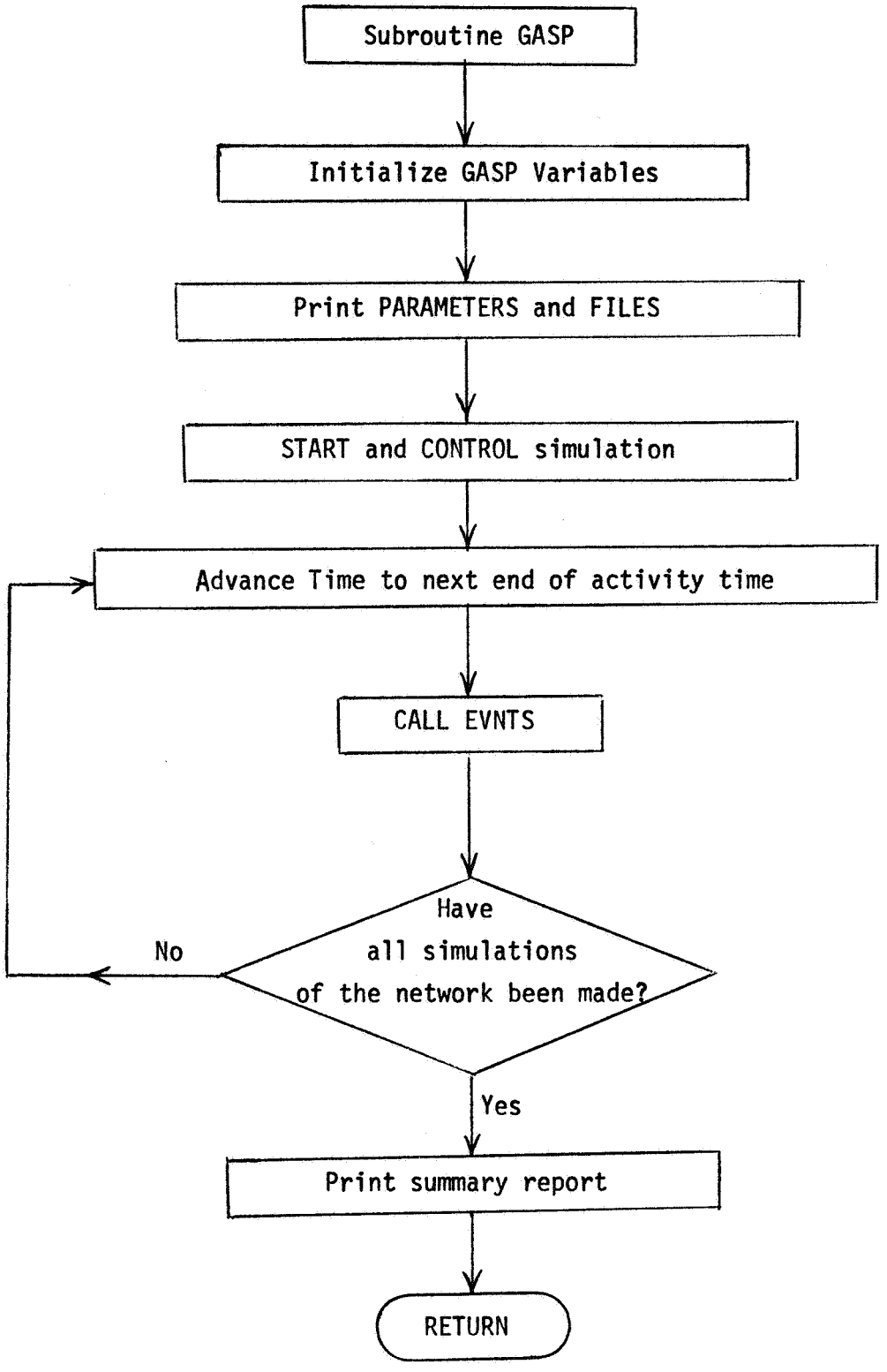

Fig. 2 Flow Chart of Subroutine GASP Described in Terms Appropriate for the GERT Simulation Program

```
// FOR
* LIST SOURCE PROGRAM
* ONE WORD INTEGERS
* IOCS (1132 PRINTER,CARD)
      DIMENSION NSET(200),QSET(40)                                          GRTS  10
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,           GRTS  20
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                  GRTS  30
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS                 GRTS  40
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),      GRTS  50
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),       GRTS  60
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)        GRTS  70
      COMMON NSINK(5),NSKST,NSKSR,NSKS,NSORC(5),NRELP(30),NREL(30),         GRTS  80
     1NTYPE(30),XLOW(5),WIDTH(5),TOTIM,NN ,NSRC                             GRTS  90
   10 FORMAT(40I2)                                                          GRTS 100
   15 FORMAT(10F10.2)                                                       GRTS 110
C
C*****SET EQUIPMENT NUMBERS FOR CARD READER(NCRDR) AND PRINTER (NPRNT).     GRTS 120
C
      NCRDR=2                                                               GRTS 130
      NPRNT=3                                                               GRTS 140
C
C*****READ LARGEST NODE NUMBER, NO. OF SINK NODES, NO. OF SOURCE NODES,     GRTS 150
C*****NO. OF NODES NECESSARY TO REALIZE THE NETWORK.                        GRTS 160
C
    7 READ(NCRDR,10)NN,NSKS,NSRC,NSKST                                      GRTS 170
C
C*****END RUN IF LARGEST NODE NO. IS NEG. OR ZERO.                          GRTS 180
C
      IF(NN)20,20,8                                                         GRTS 190
    8 NSKSR=NSKST                                                           GRTS 200
C
C*****READ SOURCE AND SINK NODE NOS.                                        GRTS 210
C
      READ(NCRDR,10) (NSORC(J),J=1,NSRC), (NSINK(K),K=1,NSKS)               GRTS 220
C
C*****READ NO. OF RELEASES FOR EACH NODE. NODE NOS. MUST BE .GE. TO 2.      GRTS 230
C
      READ(NCRDR,10)(NRELP(I),I=2,NN)                                       GRTS 240
      DO 11 I=2,NN                                                          GRTS 250
   11 NREL(I)=NRELP(I)                                                      GRTS 260
C
C*****READ NODE TYPE.  1 AND 3 ARE DETERMINISTIC.  2 AND 4 ARE PROBABILIGRTS 270
C*****3 AND 4 CAN BE REALIZED MORE THAN ONCE.                               GRTS 280
C
      READ(NCRDR,10)(NTYPE(I),I=2,NN)                                       GRTS 290
C
C*****READ LOWER LIMIT AND CELL WIDTH FOR HISTOGRAMS.                       GRTS 300
C
      READ(NCRDR,15)(XLOW(K),K=1,NSKS)                                      GRTS 310
      READ (NCRDR,15)(WIDTH(K),K=1,NSKS)                                    GRTS 320
      CALL GASP(NSET,QSET)                                                  GRTS 330
      GO TO 7                                                               GRTS 340
   20 CALL EXIT                                                             GRTS 350
      END                                                                   GRTS 360

FEATURES SUPPORTED
 ONE WORD INTEGERS
 IOCS

CORE REQUIREMENTS FOR
 COMMON   1072  VARIABLES    286  PROGRAM    212

END OF COMPILATION
```

Fig 3. FORTRAN Listing of the Main GERT Simulation Program

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE SUMRY (NSET,QSET)                                          SMRY  10
      DIMENSION NSET(1),QSET(1)                                             SMRY  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,           SMRY  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                  SMRY  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS                 SMRY  50
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),      SMRY  60
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),       SMRY  70
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLP,JTPIB(5)        SMRY  80
      COMMON NSINK(5),NSKST,NSKSR,NSKS,NSORC(5),NRELP(30),NREL(30),         SMRY  90
     1NTYPE(30),XLOW(5),WIDTH(5),TOTIM,NN ,NSRC                             SMRY 100
   21 FORMAT(1H1)                                                           SMRY 110
   10 FORMAT(29X,23HGERT SIMULATION PROJECT,I3,4H BY ,6A2/44X,4HDATE,I3,    SMRY 120
     1 1H/,I3,1H/,I5//)                                                     SMRY 130
   11 FORMAT(39X,23H**NETWORK DESCRIPTION**//16X5HSTART,6X3HEND,6X,9H ACS   SMRY 140
     1TIVITY6X,12HDISTRIBUTION,6X,11HPROBABILITY/17X4HNODE,6X,4HNODE,7X,    SMRY 150
     26HNUMBER,11X4HTYPE/)                                                  SMRY 160
   12 FORMAT(I19,I11,I12,I16,F19.4)                                         SMRY 170
   13 FORMAT(//39X23H**ACTIVITY PARAMETERS**//16X8HACTIVITY,32X,10HPARAM    SMRY 180
     1ETERS/17X6HNUMBER,16X1H1,13X1H2,13X,1H3,13X1H4/)                      SMRY 190
   14 FORMAT(I21,7X,4F14.4)                                                 SMRY 200
   15 FORMAT(//31X19H**FINAL RESULTS FOR,I5,14H SIMULATIONS**/)             SMRY 210
   16 FORMAT(          16X4HNODE,7X5HPROB,5X4HMEAN,4X8HSTD.DEV.,    4X4H    SMRY 220
     1MIN,5X4HMAX,/)                                                        SMRY 230
   17 FORMAT(I19,3X,6F10.4)                                                 SMRY 240
   18 FORMAT(//43X,14H**HISTOGRAMS**//20X         ,5HLOWER,3X,4HCELL/13X   SMRY 250
     1,4HNODE,3X,5HLIMIT,3X,5HWIDTH,31X,11HFREQUENCIES )                    SMRY 260
   19 FORMAT(/I16,F9.0,F8.1,5X,11I6/38X,11I6)                               SMRY 270
  199 FORMAT(///36X26HERROR EXIT, TYPE 98 ERROR.)                           SMRY 280
      XRUNS = NRUN                                                          SMRY 290
      WRITE (NPRNT,21)                                                      SMRY 300
      WRITE (NPRNT,10) NPROJ,NAME,MON,NDAY,NYR                              SMRY 310
      WRITE (NPRNT,11)                                                      SMRY 320
      DO 40 JQ=2,NOQ                                                        SMRY 330
      PROB = 0.0                                                            SMRY 340
      LINE = MFE(JQ)                                                        SMRY 350
      NT=NTYPE(JQ)                                                          SMRY 360
      IF (LINE) 40,40,42                                                    SMRY 370
   42 IB = (LINE-1) * MXX + 1                                               SMRY 380
      IE = IB + 2                                                           SMRY 390
      IQ = (LINE-1) * IMM + 1                                               SMRY 400
      DPROB=QSET(IQ)                                                        SMRY 410
      GO TO (72,73,72,73),NT                                                SMRY 420
   73 DPROB = QSET(IQ)-PROB                                                 SMRY 430
      PROB = PROB + DPROB                                                   SMRY 440
   72 WRITE (NPRNT,12) JQ,(NSET(I),I=IB,IE),DPROB                           SMRY 450
      LINE = NSET(IE+1)                                                     SMRY 460
      IF (LINE - 7777) 42,40,5                                              SMRY 470
    5 WRITE (NPRNT,199)                                                     SMRY 480
   40 CONTINUE                                                              SMRY 490
      WRITE (NPRNT,13)                                                      SMRY 500
      DO 50 I = 1,NPRMS                                                     SMRY 510
   50 WRITE (NPRNT,14) I,(PARAM(I,J),J=1,4)                                 SMRY 520
      WRITE(NPRNT,21)                                                       SMRY 530
      WRITE (NPRNT,10) NPROJ,NAME,MON,NDAY,NYR                              SMRY 540
      WRITE (NPRNT,15) NRUN                                                 SMRY 550
      WRITE(NPRNT,16)                                                       SMRY 560
      DO 2 I = 1,NCLCT                                                      SMRY 570
      IF(SUMA(I,3)) 5,62,61                                                 SMRY 580
   62 WRITE(NPRNT,63) NSINK(I)                                              SMRY 590
   63 FORMAT(I19,21X   18HNO VALUES RECORDED)                               SMRY 600
      GO TO 2                                                               SMRY 610
   61 XS=SUMA(I,1)                                                          SMRY 620
      XSS=SUMA(I,2)                                                         SMRY 630
      XN = SUMA(I,3)                                                        SMRY 640
      AVG=XS/XN                                                             SMRY 650
      PROB = XN/XRUNS                                                       SMRY 660
      STD=(((XN*XSS)-(XS*XS))/(XN*(XN-1.0)))**0.5                           SMRY 670
      WRITE (NPRNT,17) NSINK(I),PROB,AVG,STD,SUMA(I,4),SUMA(I,5)            SMRY 680
    2 CONTINUE                                                              SMRY 690
      WRITE(NPRNT,18)                                                       SMRY 700
      DO 82 I=1,NHIST                                                       SMRY 710
      NCL=NCELS(I)+2                                                        SMRY 720
      WRITE(NPRNT, 19) NSINK(I),XLOW(I),WIDTH(I),(JCELS(I,J),J=1,NCL)       SMRY 730
   82 CONTINUE                                                              SMRY 740
      WRITE(NPRNT,21)                                                       SMRY 750

      RETURN                                                                SMRY 760
      END                                                                   SMRY 770

 FEATURES SUPPORTED
  ONE WORD INTEGERS

 CORE REQUIREMENTS FOR SUMRY
  COMMON   1072  VARIABLES     36  PROGRAM    786

 END OF COMPILATION
```

Fig. 4 FORTRAN Listing of the GERT Version of Subroutine SUMRY

```
// FOR
* ONE WORD INTEGERS
* LIST SOURCE PROGRAM
      SUBROUTINE OTPUT(NSET,QSET)                                          OTPT  10
      DIMENSION NSET(1),QSET(1)                                            OTPT  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,          OTPT  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                 OTPT  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS                OTPT  50
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),     OTPT  60
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),      OTPT  70
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)       OTPT  80
      COMMON NSINK(5),NSKST,NSKSR,NSKS,NSORC(5),NRELP(30),NREL(30),        OTPT  90
     1NTYPE(30),XLOW(5),WIDTH(5),TOTIM,NN ,NSRC                            OTPT 100
      RETURN                                                               OTPT 110
      END                                                                  OTPT 120

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR OTPUT
 COMMON    1072  VARIABLES       0  PROGRAM       8

END OF COMPILATION
```

Fig. 5 FORTRAN Listing of the Dummy Subroutine OTPUT

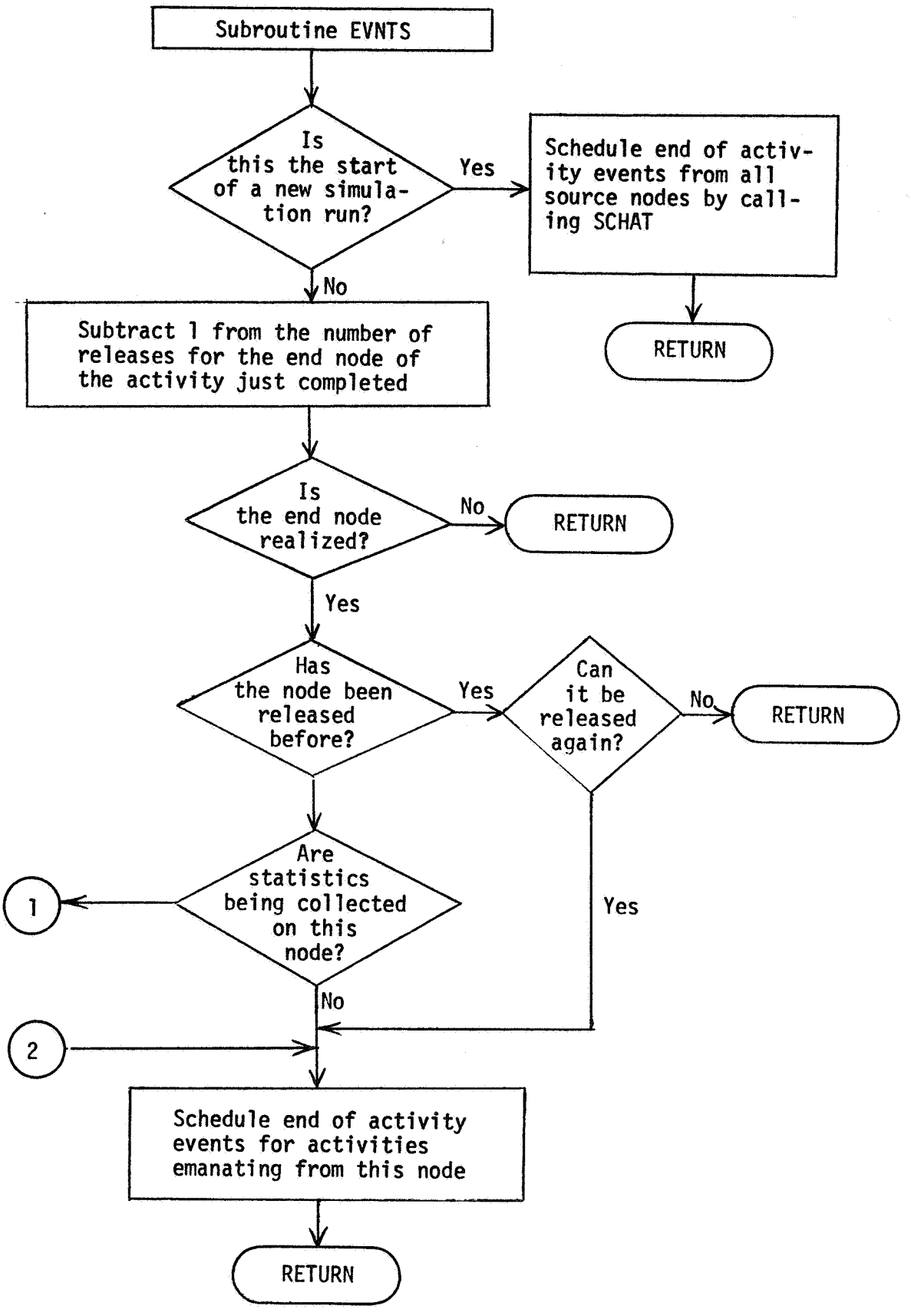

Fig. 6 Flow Chart of Subroutine EVNTS

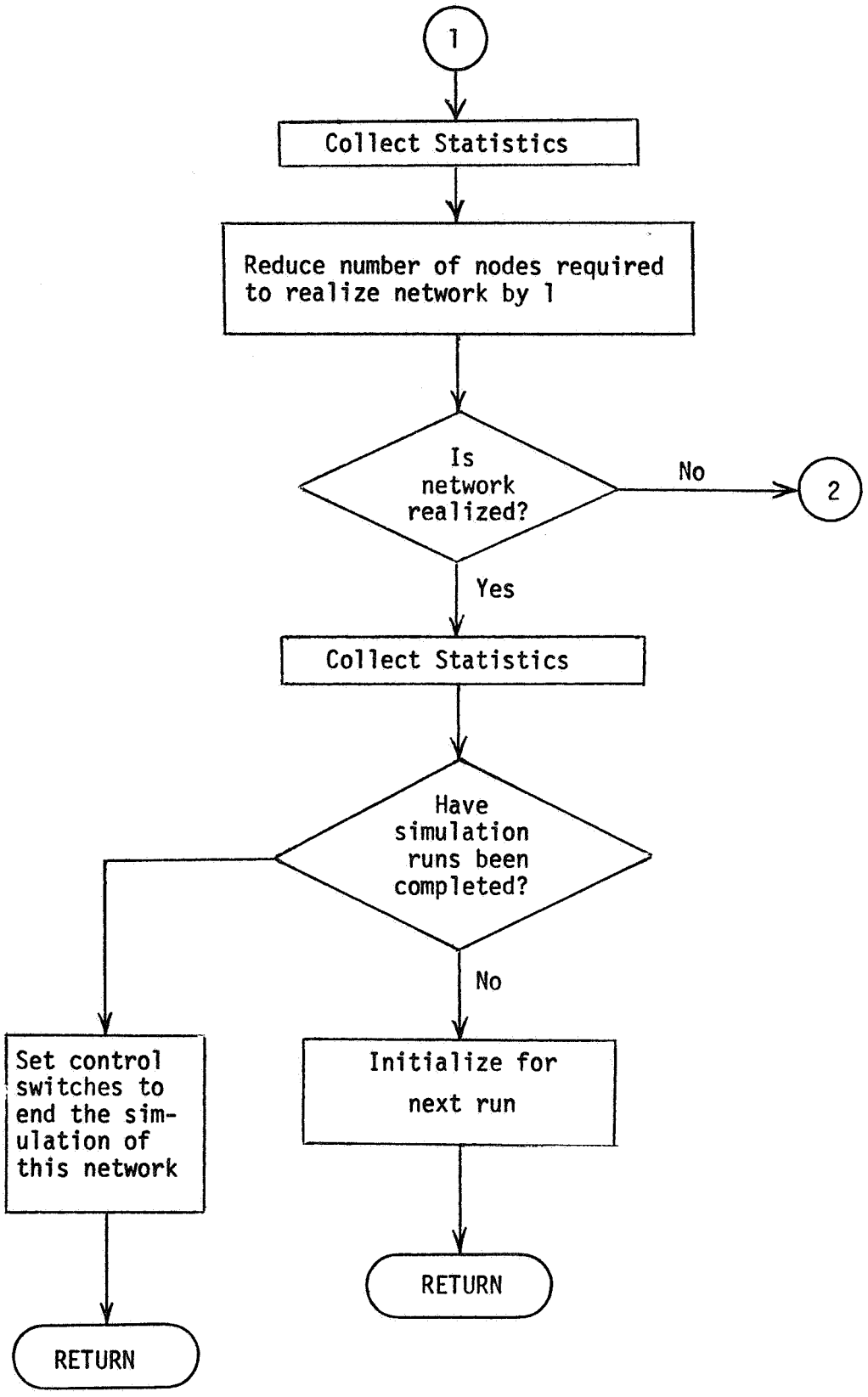

Fig. 6 Flow Chart of Subroutine EVNTS (continued)

When NEND is not zero, the number of releases associated with NEND is decremented by one, and a test is made to determine if there are more releases for NEND. If the end node is not realized, a return is made to subroutine GASP. When the number of releases is less than or equal to zero, a test is made to determine if the node has been realized previously in this simulation run. If it has, a test is made to determine if it can be released again, which will be the case if its node type is either three or four. When this is the case, end of activity events are scheduled in accordance with the node type by calling subroutine SCHAT.

If the node is being released for the first time, a test is made to determine if statistics are to be collected for this node. If not, end of activity events for activities emanating from the node are scheduled in accordance with its node type.

If the node is one on which statistics are being collected, GASP subroutine COLCT is used to obtain data from which the average, standard deviation, minimum and maximum time and the probability that the node is realized can be computed. Subroutine HISTO is used to prepare a histogram on the times the node was realized. Each node on which statistics are collected is considered as a potential terminal node. The number of terminal nodes required to realize the network is reduced by one and when this value is zero the network is realized. If the network is not realized, end of activity events are scheduled if there are activities emanating from the node.

When the network is realized, one complete simulation of the network has been completed. At this point network statistics are calculated and if more simulation runs are to be made for this network, the event file is cleared and a start-up event inserted into it. The number of releases associated with each node is reset and control returned to GASP. When all simulation runs have been made, the control variable MSTOP is set to -1 to communicate to subroutine GASP that all runs have been made.

The FORTRAN listing for subroutine EVNTS is given in Fig.7.

Subroutine SCHAT(NODE,NSET,QSET)

Subroutine SCHAT determines which activities from the node, NODE, should be realized and schedules end of activity events for each of these activities. The flow chart for SCHAT is given in Fig. 8. Many of the statements in this subroutine are required to find the location in the one dimensional arrays NSET and QSET of an activity. These statements do not add significantly to the execution time of the program since if two dimensional arrays were used, the FORTRAN compiling system would perform the function.

If the node from which activities are to be scheduled is a deterministic node, it will have a node type of either 1 or 3. When this is the case, all activities emanating from the node will be scheduled to occur and an end of activity event will be inserted into the

```
// FOR
* ONE WORD INTEGERS
* LIST SOURCE PROGRAM
      SUBROUTINE EVNTS(NEND,NSET,QSET)                                  EVNT  10
      DIMENSION NSET(1),QSET(1)                                         EVNT  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,       EVNT  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,              EVNT  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS             EVNT  50
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),  EVNT  60
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),   EVNT  70
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)    EVNT  80
      COMMON NSINK(5),NSKST,NSKSR,NSKS,NSORC(5),NRELP(30),NREL(30),     EVNT  90
     1NTYPE(30),XLOW(5),WIDTH(5),TOTIM,NN ,NSRC                         EVNT 100
C
C*****IF END NODE IS ZERO START NETWORK BY SCHEDULING ACTIVITIES         EVNT 110
C*****FROM EACH SOURCE NODE                                             EVNT 120
C
      IF(NEND)4,4,5                                                     EVNT 130
    4 DO 10 N=1,NSRC                                                    EVNT 140
      M=NSORC(N)                                                        EVNT 150
      CALL SCHAT(M,NSET,QSET)                                           EVNT 160
   10 CONTINUE                                                          EVNT 170
   99 RETURN                                                            EVNT 180
C
C*****REDUCE NO. OF RELEASES FOR END NODE BY 1.                         EVNT 190
C*****TEST TO SEE IF NODE IS RELEASED.                                  EVNT 200
C
    5 NREL(NEND)=NREL(NEND)-1                                           EVNT 210
      IF(NREL(NEND)) 9,7,99                                             EVNT 220
C
C*****TEST TO SEE IF NODE CAN BE RELEASED MORE THAN ONCE.               EVNT 230
C
    9 IF(NTYPE(NEND)-2) 99,99,50                                        EVNT 240
C
C*****TEST TO SEE IF STATISTICS ARE TO BE COLLECTED ON THE NODE.        EVNT 250
C
    7 DO 8 K=1,NSKS                                                     EVNT 260
      IF(NEND-NSINK(K)) 8,40,8                                          EVNT 270
    8 CONTINUE                                                          EVNT 280
   50 CALL SCHAT(NEND,NSET,QSET)                                        EVNT 290
      RETURN                                                            EVNT 300
C
C*****REDUCE NO. OF SINK NODES REALIZED BY ONE.  COLLECT STATISTICS.    EVNT 310
C
   40 NSKSR=NSKSR-1                                                     EVNT 320
      CALL COLCT(TNOW,K,NSET,QSET)                                      EVNT 330
      CALL HISTO(TNOW,XLOW(K),WIDTH(K),K)                               EVNT 340
C
C*****TEST TO SEE IF NETWORK SIMULATION IS COMPLETE.                    EVNT 350
C
      IF(NSKSR) 22,100,999                                              EVNT 360
  999 IF(NQ(NEND)) 22,99,50                                             EVNT 370
   22 CALL ERROR(22,NSET,QSET)                                          EVNT 380
C
C*****NETWORK HAS BEEN SIMULATED ONE MORE TIME.  ADD TNOW TO TOTAL TIME.EVNT 390
C
  100 TOTIM=TOTIM+TNOW                                                  EVNT 400
C
C*****TEST TO SEE IF ALL RUNS HAVE BEEN MADE.                           EVNT 410
C
      IF(NRUNS-NRUN)110,110,115                                         EVNT 420
C
C*****REINITIALE FOR ANOTHER RUN BY REMOVING ALL EVENTS FROM EVENT      EVNT 430
C*****FILE AND RESETTING NETWORK VALUES.                                EVNT 440
C
  115 IF(NQ(1))22,120,116                                               EVNT 450
  116 CALL RMOVE(MFE(1),1,NSET,QSET)                                    EVNT 460
      GO TO 115                                                         EVNT 470
  120 NRUN=NRUN+1                                                       EVNT 480
      JTRIB(1)=0                                                        EVNT 490
      ATRIB(1)=0.                                                       EVNT 500
      CALL FILEM(1,NSET,QSET)                                           EVNT 510
      TNOW=0.                                                           EVNT 520
      DO 36 IJ=1,NOQ                                                    EVNT 530
   36 QTIME(IJ)=0.0                                                     EVNT 540
      NSKSR=NSKST                                                       EVNT 550
      DO 37 IJ=1,NN                                                     EVNT 560
   37 NREL(IJ)=NRELP(IJ)                                                EVNT 570
      RETURN                                                            EVNT 580

  110 MSTOP=-1                                                          EVNT 590
      NRUNS=1                                                           EVNT 600
      TNOW=TOTIM                                                        EVNT 610
      DO 142 IJ=1,NOQ                                                   EVNT 620
  142 QTIME(IJ)=TOTIM                                                   EVNT 630
      RETURN                                                            EVNT 640
      END                                                               EVNT 650

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR EVNTS
 COMMON   1072  VARIABLES      6  PROGRAM    334

END OF COMPILATION
```

Fig. 7 FORTRAN Listing of Subroutine EVNTS

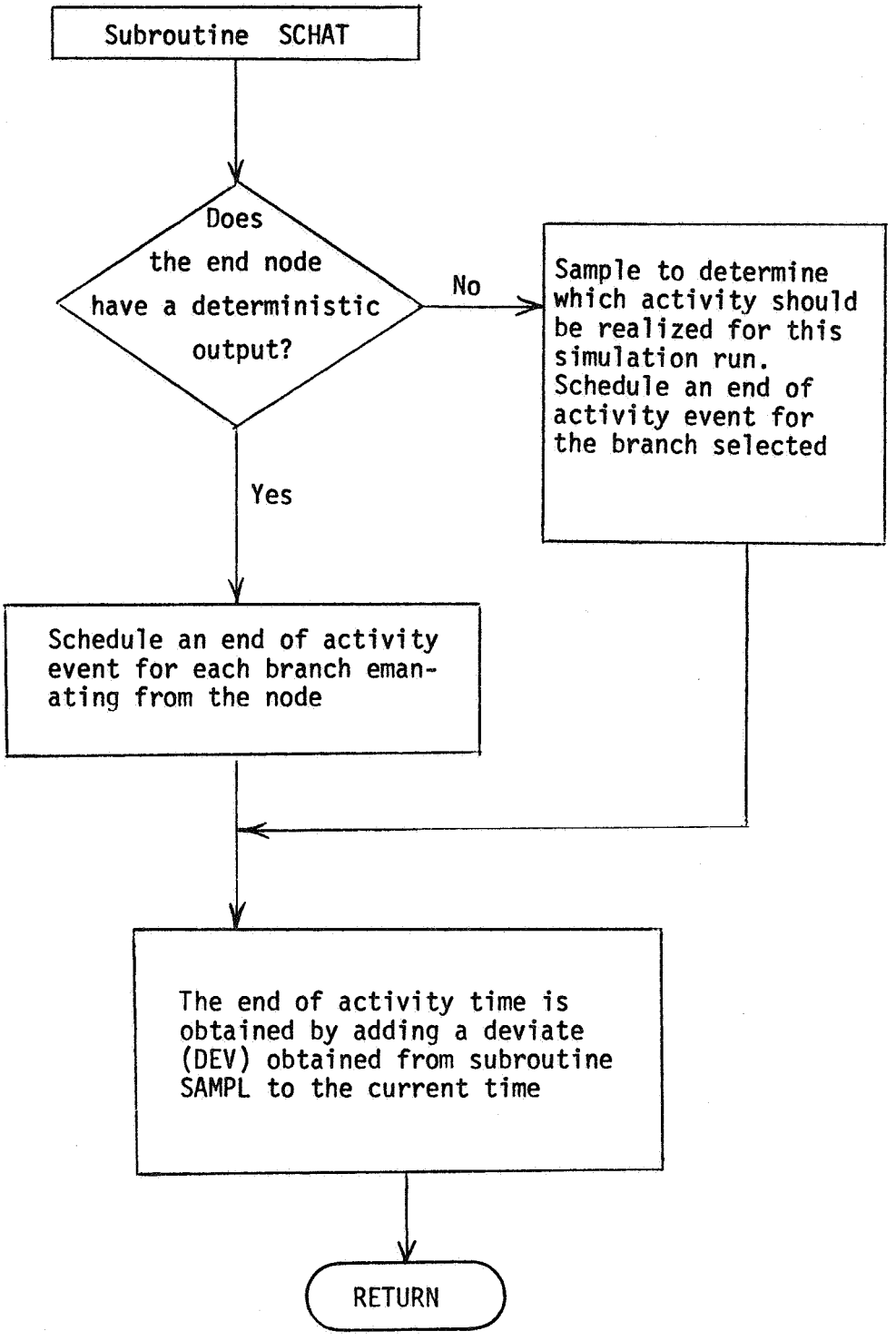

Fig. 8 Flow Chart of Subroutine SCHAT for Scheduling End of Activity Events

event file for each activity. The end node of the activity is stored in JTRIB(1) as shown in the statement sequenced as SCAT 230. The parameter set for the activity is stored in JTRIB(2) and the distribution type in JTRIB(3) so that these values will be available to subroutine SAMPL when it is called in the statement sequenced as SCAT 250. After these values are defined, subroutine SAMPL is called and the time to perform the activity is returned in the variable DEV. DEV is added to current time, TNOW, to determine the time at which the end of the activity will occur. This is stored in the buffer storage vector ATRIB(1). The event is inserted into the event file by calling the GASP IIA subroutine FILEM with a file number of 1. This procedure is continued for deterministic nodes until all activities emanating from the deterministic node are scheduled.

When the node type is probabilistic (node type 2 or 4) then a sampling procedure is used to determine which one of the activities should occur. A random number is generated and tested against the cumulative probability that a branch or one of the previous branches stored in the file of the node will be taken. By testing in the order in which the activities are stored in the file, the activities will be selected with the same relative frequency as indicated by the probabilities on the branches. When one activity is selected to occur it is scheduled in the same manner as the activities emanating from a deterministic node. A return is made at this point.

The subroutine DRAND is used to generate a random number. This subroutine uses the system random number generator. For the IBM 1130,

this is subroutine RANDU and has two arguments: ISEED, an initial random integer; and RNUM, the random number generated. If the system random number generator is different, then only subroutine DRAND need be changed in order to use the system random number routine.

The FORTRAN listing for subroutine SCHAT is given in Fig. 9.

Subroutine SAMPL(DEV,NSET,QSET)

Subroutine SAMPL obtains a deviate, DEV, which represents the time to perform an activity. The value of DEV is determined by sampling from the distribution type of the activity defined in JTRIB(3) with the parameter set given by JTRIB(2). Subroutine SAMPL is only called from subroutine SCHAT where JTRIB(2) and JTRIB(3) are set. In subroutine SAMPL, these values are assigned to the variables JP and JD respectively.

A computed GO TO statement based on the value of JD is used to determine the distribution type from which the deviate is to be obtained. The flow chart for subroutine SAMPL is shown in Fig. 10, and the FORTRAN listing is given in Fig. 11. The computed GO TO statement for specifying the distribution type is shown in the statement sequenced as SAMPL 130. From Fig. 10 and 11 the following distribution type codes can be identified:

(1) The deviate will be a constant value;

(2) The deviate will be drawn from a normal distribution;

(3) The deviate will be drawn from a uniform distribution;

(4) The deviate will be drawn from an Erlang distribution.

This subroutine has been written to permit the addition of other distributions by inserting additional numbers in the computed GO TO

```
// FOR
* ONE WORD INTEGERS
* LIST SOURCE PROGRAM
      SUBROUTINE SCHAT(NODE,NSET,QSET)                                  SCAT  10
      DIMENSION NSET(1),QSET(1)                                         SCAT  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,       SCAT  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,              SCAT  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS             SCAT  50
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),  SCAT  60
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),   SCAT  70
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)    SCAT  80
      COMMON NSINK(5),NSKST,NSKSR,NSKS,NSORC(5),NRELP(30),NREL(30),     SCAT  90
     1NTYPE(30),XLOW(5),WIDTH(5),TOTIM,NN ,NSRC                         SCAT 100
C
C*****NEXT IS LOCATION OF FIRST ENTRY IN FILE OF ACTIVITIES WITH START  SCAT 110
C*****NODE- NODE. NT IS THE NODE TYPE.                                  SCAT 120
C
      NEXT=MFE(NODE)                                                    SCAT 130
      NT=NTYPE(NODE)                                                    SCAT 140
C
C*****GENERATE A RANDOM NUMBER RNUM.                                    SCAT 150
C
      CALL DRAND(ISEED,RNUM)                                            SCAT 160
C
C*****OBTAIN STARTING LOCATION IN NSET OF ENTRY.                        SCAT 170
C
    1 INDXN=(NEXT-1)*MXX                                                SCAT 180
C
C*****TEST IF DETERMINISTIC OR PROBABILISTIC NODE.                      SCAT 190
C
      GO TO (100,200,100,200),NT                                        SCAT 200
  100 DO 10 I=1,IM                                                      SCAT 210
      INDX=INDXN+I                                                      SCAT 220
   10 JTRIB(I)=NSET(INDX)                                               SCAT 230
C
C*****OBTAIN SAMPLE FOR ACTIVITY.                                       SCAT 240
C
      CALL SAMPL(DEV,NSET,QSET)                                         SCAT 250
      ATRIB(1)=TNOW+DEV                                                 SCAT 260
C
C*****FILE END OF ACTIVITY EVENT IN EVENT FILE.                         SCAT 270
C
      CALL FILEM(1,NSET,QSET)                                           SCAT 280
      GO TO (110,4,110,4),NT                                            SCAT 290
C
C*****DETERMINE IF OTHER ACTIVITIES ARE IN FILE.                        SCAT 300
C
  110 INDX=INDXN+MX                                                     SCAT 310
      NEXT=NSET(INDX)                                                   SCAT 320
      IF(NEXT-7777)1,4,23                                               SCAT 330
   23 CALL ERROR(23,NSET,QSET)                                          SCAT 340
    4 RETURN                                                            SCAT 350
  200 INDX=(NEXT-1)*IMM+1                                               SCAT 360
C
C*****TEST RNUM AGAINST PROBABILITY (CUM.) OF REALIZING THE ACTIVITY.   SCAT 370
C
      IF(QSET(INDX)-RNUM)110,100,100                                    SCAT 380
      END                                                               SCAT 390

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR SCHAT
 COMMON    1072  VARIABLES      12  PROGRAM     170

END OF COMPILATION
```

Fig. 9 FORTRAN Listing of Subroutine SCHAT

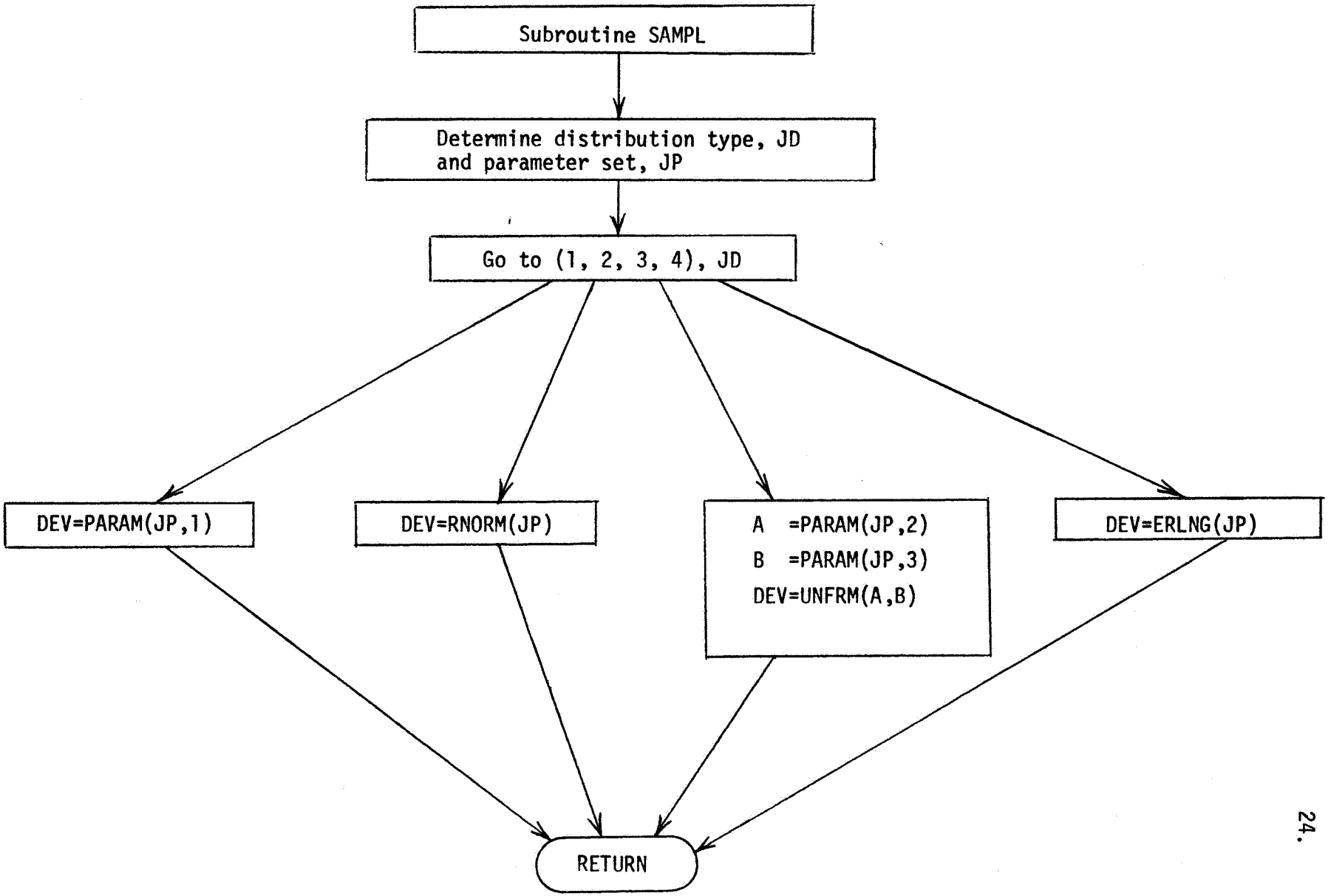

Fig. 10 Flow Chart of Subroutine SAMPL for Obtaining Deviates from Specified Distributions

```
// FOR
* ONE WORD INTEGERS
* LIST SOURCE PROGRAM
      SUBROUTINE SAMPL(DEV,NSET,QSET)                                    SAMP  10
      DIMENSION NSET(1),QSET(1)                                          SAMP  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,        SAMP  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,               SAMP  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS              SAMP  50
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),   SAMP  60
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),    SAMP  70
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)     SAMP  80
      COMMON NSINK(5),NSKST,NSKSR,NSKS,NSORC(5),NRELP(30),NREL(30),      SAMP  90
     1NTYPE(30),XLOW(5),WIDTH(5),TOTIM,NN ,NSRC                          SAMP 100
      JP=JTRIB(2)                                                        SAMP 110
      JD=JTRIB(3)                                                        SAMP 120
      GO TO (1,2,3,4),JD                                                 SAMP 130
    1 DEV=PARAM(JP,1)                                                    SAMP 140
      RETURN                                                             SAMP 150
    2 DEV=RNORM(JP)                                                      SAMP 160
      RETURN                                                             SAMP 170
    3 A=PARAM(JP,2)                                                      SAMP 180
      B=PARAM(JP,3)                                                      SAMP 190
      DEV=UNFRM(A,B)                                                     SAMP 200
      RETURN                                                             SAMP 210
    4 DEV=ERLNG(JP)                                                      SAMP 220
      RETURN                                                             SAMP 230
      END                                                                SAMP 240

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR SAMPL
 COMMON    1072  VARIABLES       8  PROGRAM      82

END OF COMPILATION
```

Fig. 11 FORTRAN Listing of Subroutine SAMPL

statement and providing subprograms for the calculation of DEV based on the new distribution types. Although not included in subroutine SAMPL, GASP IIA has subprograms to sample from the Poisson distribution and the log-normal distribution. These were not included in the IBM 1130 version due to storage limitations.*

The array PARAM (40,4) contains a maximum of four attributes associated with a distrubution. A maximum of 40 parameter sets has been included in the IBM 1130 version of the GERT simulation program. This permits a network with 40 activities each with a different parameter set to be simulated. Normally, the number of parameter sets required is significantly less than the number of activities of the network since many activities could use the same parameter set. In addition, if a constant is equal to the mean of the normal distribution, then the same parameter set for the normal distribution can be used to obtain a deviate which is equal to a constant value. This is seen in statement SAMPL 140 where DEV is set equal to PARAM (JP,1). A similar discussion holds for the uniform distribution where only the minimum and maximum values of the parameter set are required.

Limitations and Requirements

The GERT simulation program presented in this report has been programmed and run on the IBM 1130 system with 8192 words of core storage and 1 disc unit. In order to fit the program on this computer, use

---

*Since these subprograms call other GASP subprograms which are put in LOCAL storage, they could not be put in LOCAL storage. Thus, they would take additional core storage. For larger machines, these subprograms can easily be included.

is made of the LOCAL and SOCAL features of the computer. A loading map for the situation where the number of activities of the network and the maximum number of simultaneous end of activity events is less than or equal to 30 (ID is set at 30) is shown in Fig. 12. In order to increase the value of ID to 40, subroutine RMOVE is also put in LOCAL. From a general point of view, an ID value of 40 is the limit for simulating general GERT networks since a maximum of 40 different parameter sets has also been set by the dimension of the array PARAM. When the number of parameter sets is less than 40, simulations have been performed on the IBM 1130 with an ID value equal to 48. (This was done without changing the dimensions on PARAM.)

The GERT simulation program uses the system random number generator. For the IBM 1130 the system random number generator used in subroutine RANDU(ISEED,RNUM). ISEED is the starting value and is a fixed point variable. RNUM is the pseudo-random number obtained and is the interval (0,1). The only subprogram which calls RANDU is subroutine DRAND. Thus, it is only necessary to change subroutine DRAND in order to include a particular random number generator. DRAND is required to be a subroutine for the IBM 1130 since a function cannot call other subprograms in this system. If DRAND can be made a function, then the subprograms for obtaining deviates from specified distributions can be simplified by putting DRAND directly within the statement used for obtaining the deviate.

The largest node number which can be used with the GERT simulation program is 31. Since node numbers must be greater than or equal to 2,

```
// XEQ       L  1
*LOCAL,DATAN,SUMRY,OTPUT,MONTR,HISTO,COLCT,RNORM,ERLNG

R 40  0315 (HEX) ADDITIONAL CORE REQ D                                    28.
R 43  032A (HEX) ARITH/FUNC OVERLAY SIZE
R 44  05E2 (HEX)  FIO & I/O OVERLAY SIZE
R 47  0006 (HEX) WORDS AVAILABLE
R 03 LOAD REQUIRES SYSTEM LOCAL, LEVEL 1

          CALL TRANSFER VECTOR

          UNFRM    1152
          SAMPL    10FC
          SCHAT    103D
          FXPN     17C4  SOCAL
          FLN      1688  SOCAL
          XMAX     0FCC
          RANDU    0F25
          EVNTS    0DD2
          RMOVE    0D3C
          FCOS     170E  SOCAL
          FALOG    1686  SOCAL
          AMAX     0D12
          AMIN     0CF0
          FAXB     1655  SOCAL
          FILEM    0C3D
          SET      06CE
          DRAND    0698
          ERROR    0560
          GASP     0406
          ERLNG    11FC  LOCAL
          RNORM    11FC  LOCAL
          COLCT    11EF  LOCAL
          HISTO    11FE  LOCAL
          MONTR    1282  LOCAL
          OTPUT    11E4  LOCAL
          SUMRY    1350  LOCAL
          DATAN    1253  LOCAL

          LIBF TRANSFER VECTOR

          TTEST    10E4
          EBCTB    1B03  SOCAL
          HOLTB    1ACD  SOCAL
          GETAD    1A8C  SOCAL
          FGETP    1838  SOCAL
          XMDS     17A6  SOCAL
          FARC     1784  SOCAL
          NORM     1004
          SGOTO    0FEE
          SFIF     0FB0
          SFAR     0F90
          SIARX    0F88
          SFARX    0FAA
          SIAR     0F6C
          SIIF     0F54
          HOLEZ    1A56  SOCAL
          FSUBX    1531  SOCAL
          FMPYX    15E8  SOCAL
          FADDX    153D  SOCAL
          IFIX     0CC2
          FDVR     163A  SOCAL
          FSBR     1620  SOCAL
          FMPY     15EC  SOCAL
          FDIV     1596  SOCAL
          FADD     1537  SOCAL
          FSUB     152C  SOCAL
          FLDX     03B4
          FLOAT    0CB8
          SCOMP    15F0  SOCAL
          SWRT     1525  SOCAL
          SNR      1524  SOCAL
          FSTOX    0398
          SIOF     15EC  SOCAL
          SIOAI    160B  SOCAL
          SUBIN    04C8
          SIOFX    1602  SOCAL
          SIOIX    15AB  SOCAL
          SUBSC    03CE
          SIOI     1606  SOCAL
          SRED     152A  SOCAL
          FSTO     039C
          FLD      03B8
          PRNTZ    198E  SOCAL
          CARDZ    192D  SOCAL
          SFIO     1615  SOCAL
          DISKZ    00F4

          SYSTEM ROUTINES

          ILS02    1B07
          FLIPR    119C
     02CD (HEX) IS THE EXECUTION ADDR.
```

Fig. 12 Loading Map for the GERT Simulation Program for the IBM 1130

this permits a 30 node network to be simulated on the IBM 1130 system. For computers with larger storage capacity, the number of nodes permitted is easily extended by changing the dimension of the following variables: VNQ, ENQ, INN, KRANK, MAXNQ, MFE, MLC, MLE, NQ, QTIME, NRELP, NREL, and NTYPE. The last three variables are non-GASP variables and need only be changed in the non-GASP subprograms.

The maximum number of source nodes is five. To increase this number, the non-GASP array NSORC should be redimensioned to the largest number of source nodes desired. The largest number of nodes on which statistics can be collected is five. In order to increase this number the following variables must be redimensioned: JCELS (only the first subscript), NCELS, SUMA (only the first subscript), NSINK, XLOW, and WIDTH. The last three variables named are non-GASP variables and need only be changed in the non-GASP subprograms. To increase the number of different parameter sets which describe the distribution of the times to perform an activity, the first subscript of the array PARAM should be increased. To increase the largest number of activities plus simultaneous end of activity events, the variable ID must be read in during initialization at the largest value desired. In the main program, NSET should be dimensioned to (5*ID) and QSET should be dimensioned at ID. The GERT simulation program is written so that only the dimension statement in the main program need be changed to accomplish this increase in the number of entries in the filing array. This is the reason for carrying NSET and QSET as arguments to all subprograms.

No general statements can be made about the time required to simulate a network with a specified number of nodes and activities. The

simulation time depends on:

(1) The number of nodes;

(2) The number of activities;

(3) The number of feedback branches in a network;

(4) The probability of taking the feedback branches;

(5) The number of subprograms put in LOCAL; and

(6) The number of system operations put in SOCAL.

For a network with a single feedback branch and three nodes, it took less than three minutes to perform 1,000 simulations of the network. For a network with ID=48 and many feedback branches, it took approximately 30 minutes to perform 100 simulations of the network.

BIBLIOGRAPHY ON GERT AND RELATED TOPICS

1. Clapp, D. E., "The Application of Stochastic Networks," Masters Report, Arizona State University, May 1965.

2. Deutsch, D., "The Theory of Blocking," Masters Report, Arizona State University, May, 1966.

3. Drezner, S. M. and A. A. B. Pritsker, Network Analysis of Countdown, The RAND Corporation, RM-4976-NASA, March, 1966.

4. Eisner, Howard, "A Generalized Network Approach to the Planning and Scheduling of Research Program," Journal of Operations Research Society, Vol. 10, No. 1, 1962.

5. Elmaghraby, Salah E., "An Algebra for the Analysis of Generalized Activity Networks," Management Science, Vol. 10, No. 3, April 1964, pp. 494-514.

6. _____. "On Generalized Activity Networks," Journal of Industrial Engineering, Vol. XVII, No. 11, November, 1966, pp. 621-631.

7. Enlow, R. A. and A. A. B. Pritsker, Planning R&D Projects Using GERT, NASA/ERC, NGR 03-001-034, Arizona State University, July, 1968.

8. Graham, P., "Profit Probability Analysis of Research and Development Expenditures," The Journal of Industrial Engineering, Vol. XVI, No. 3, May-June, 1965, pp. 186-191.

9. Happ, W. W., "Flowgraph Techniques for Closed Systems," IEEE Transactions, AES, Vol. 2, May, 1966, pp. 252-264.

10. Hill, T. W., "System Optimization: A Sensitivity Approach Using GERT," unpublished Masters Research Report, Arizona State University, Tempe, Arizona, 1966 (Also in Bulletin #3, Industrial Engineering Research at Arizona State University, 1966)

11. Ishmael, P. C. and A. A. B. Pritsker, User Manual for GERT EXCLUSIVE-OR Program, NASA/ERC, NGR 03-001-034, Arizona State University, July, 1968.

12. _____. Definitions and Procedures Employed in the GERT EXCLUSIVE-OR Program, NASA/ERC, NGR 03-001-034, Arizona State University, July, 1968.

13. Ledbetter, L. R., "A Study of Computational Aspects of Network Models for Planning and Control," Masters Thesis, Lehigh University, 1967.

14. McDonald, W. W., "Analysis of Stochastic Networks," in GRAPH THEORY, Significant Books in Operations Research Management Information Services, Detroit, Michigan. (Also Masters Thesis, Arizona State University, 1966.)

15. Naylor, T. H., et. al., Computer Simulation Techniques, New York: John Wiley & Sons, Inc., 1966.

16. Powell, G. E., "Development, Evaluation and Selection of a Dodge Continuous Sampling Plan When the Rectifying Operation is Not Perfect," Masters Thesis, Lehigh University, 1967.

17. Pritsker, A.A.B., GERT: Graphical Evaluation and Review Techniques, The RAND Corporation, RM-4973-NASA, April, 1966.

18. _____, User's Manual for GERT Simulation Program, NASA/ERC, NGR 03-001-034, Arizona State University, July, 1968.

19. _____, Definitions and Procedures for GERT Simulation Program, NASA/ERC, NGR 03-001-034, Arizona State University, July, 1968.

20. _____, and W. W. Happ, "GERT: Graphical Evaluation and Review Technique, Part I Fundamentals," The Journal of Industrial Engineering, Vol. XVII, No. 5, pp. 267, May, 1966.

21. _____, and G. E. Whitehouse, "GERT: Graphical Evaluation and Review Technique, Part II Probabilistic and Industrial Engineering Applications," Journal of Industrial Engineering, Vol. XVII, No. 6, June, 1966.

22. _____, and P. J. Kiviat, GASP II, A FORTRAN Based Simulation Language, Department of Industrial Engineering, Arizona State University, September, 1967.

23. _____ _____, Simulation Using GASP II, to be published by Prentice-Hall, Inc., 1968.

24. Smith, R. L., "Stochastic Analysis of Personnel Movement in Formal Organizations," Ph.D. Dissertation, Arizona State University, August, 1967.

25. Thomas, J. H., "An Investigation into the Output Distribution of GERT Networks by the Use of Simulation," (Masters Thesis), Lehigh University, 1967.

26. Werberger, G., "Optimality Considerations of GERT," Masters Thesis, Lehigh University, 1967.

27. Whitehouse, Gary E., "Extensions, New Developments, and Applications of GERT," Ph.D. Dissertation, Arizona State University, August, 1965.

APPENDIX A

GLOSSARY OF GASP IIA SUBPROGRAMS AND VARIABLES

GASP IIA Subprograms

NAME	FUNCTION
AMAX(ARG1, ARG2)	Sets AMAX to the maximum of ARG1 and ARG2.
XMAX(IARG1, IARG2)	Sets XMAX to the maximum of IARG1 and IARG2.
AMIN(ARG1, ARG2)	Sets AMIN to the minimum of ARG1 and ARG2.
COLCT(X, N, NSET, QSET)	Sets X_i = X and collects: $\sum_i X_i$ in SUMA(N, 1) $\sum_i X_i^2$ in SUMA(N, 2) Number of observations of type N in SUMA(N, 3) $\min_i (X_i)$ in SUMA(N, 4) $\max_i (X_i)$ in SUMA(N, 5)
DATAN(NSET, QSET)	Data input and initialization routine
DRAND(ISEED, RNUM)	Generates a random number RNUM, using the computer system's random number routine.
ERLNG(J)	Generates a random deviate from an ERLNG distribution with parameters in PARAM(J,I), I = 1, 2, 3, 4.
ERROR(J, NSET, QSET)	Prints error code J, NSET, and File 1 and calls SUMRY. Then calls EXIT.

EVNTS(I, NSET, QSET)	Written by the programmer to call the subroutine defining event I.
FILEM(JQ, NSET, QSET)	Stores the vector ATRIB in QSET and JTRIB in NSET of File JQ.
FINDN(NVAL, MCODE, JQ, JATT, KCOL, NSET, QSET)	Locates KCOL of File JQ that has an attribute value in row JATT of NSET related to NVAL according to code MCODE. KCOL is the unknown entry.
FINDQ(QVAL, MCODE, JQ, JATT, TOL, KCOL, NSET, QSET)	Locates KCOL of File JQ that has an attribute value in row JATT or QSET related to QVAL according to code MCODE within a tolerance of TOL, KCOL is the unknown entry.
GASP(NSET, QSET)	Selects the next scheduled event and controls simulation from start to end.
HISTO(X1, A, W, N)	X1 is a value to be used to form histogram N. A is the lower limit. W is the cell width.
LOCAT(JSW, K, JATT, NSET, QSET)	Converts from a one dimensional representation to two dimensional representation for JSW = 1, 2. Converts from a two dimensional representation to a one dimensional representation for JSW = 3, 4. JATT is always the attribute number. For JSW = 1 or 2, K is the column number and LOCAT is set to a cell number or index of QSET or NSET. For JSW = 3 or 4, K is the cell number of QSET or NSET and LOCAT is set at the column number.
MONTR(NSET, QSET)	The subroutine that enables a programmer to print NSET at any time during a simulation or to trace the events during a portion of a simulation.
NPOSN(J, NPSSN)	Generates a random deviate from a Poisson distribution with parameter PARAM (J,I), I = 1, 2, 3, 4.
OTPUT(NSET, QSET)	A subroutine written by a programmer to perform calculations and provide additional output at the end of a simulation run.

PRNTQ(JQ, NSET, QSET)	Prints average, standard deviation, maximum number of entries and contents of in File JQ.
RLOGN(J)	Generates a random deviate from a log-normal distribution with parameters PARAM (J, I), I = 1, 2, 3, 4.
RMOVE(KCOL, JQ, NSET, QSET)	Removes column KCOL from File JQ and stores attribute of NSET in the vector JTRIB and attributes of QSET in ATRIB.
RNORM(J)	Generates a random deviate from a normal distribution with parameters PARAM(J, I), I = 1, 2, 3, 4.
SET(JQ, NSET, QSET)	Performs functions related to entry storage and retrieval for File JQ.
SUMRY(NSET, QSET)	Prints a final GASP Summary report.
TMST(X, T, N, NSET, QSET)	Sets X_i = X and computes the time since the last change in variable N, TT_i, and collects: $\sum_i TT_i$ in SSUMA(N, 1) $\sum_i X_i*TT_i$ in SSUMA(N, 2) $\sum_i X_i^2*TT_i$ in SSUMA(N, 2) $\min_i(X_i)$ in SSUMA(N, 4) $\max_i(X_i)$ in SSUMA(N, 5)
UNFRM(A, B)	Generates a random deviate from a uniform distribution between A and B.

GASP IIA Variables

The following is an alphabetized list of GASP IIA variables stored in COMMON. These variables must be used by a programmer with great care. All GASP variables are in COMMON except the arrays NSET and QSET. Variables are dimensioned as required by the GERT simulation program.

ATRIB(1)	Buffer for attribute value being stored in or retrieved from QSET.
ENQ(NN)	Time integrated number of entries in a file.
ID	Number of columns in total filing array, limited only by available storage .
IM	Number of attribute rows in NSET (IM=3)
IMM	Number of attribute rows of QSET (IMM=1)
INIT	An indicator. The statements INIT=1 CALL SET (1,NSET) initialize NSET.
INN(NN)	A file indicator. If INN(J)=1, entries in File J are ordered by row KRANK(J) from lowest value to highest value (LVF). If INN(J)=2, the entries in File J are ordered by row KRANK(J) from the highest value to lowest value (HVF). For the GERT simulation program all INN(J) values are 1.
ISEED	The initial random number.
JCELS(NHIST, MXC)	Storage array for histograms. (NHIST $\leq$ 5, MXC $\leq$ 22).

JCLR	An indicator. If JCLR ≤ 0, the simulation is repeated without changing the condition of the statistical storage areas. If JCLR > 0, the statistical storage areas are initialized prior to repeating a simulation run.
JEVNT	Code of event to be processed. Also used as a control in subroutine MONTR where if JEVNT=101, NSET is printed, and if JEVNT=100, the next event is printed until another event with JEVNT=100 occurs.
JMNIT	An indicator. If JMNIT=1, each event is monitored. If JMNIT=0, no monitoring occurs.
JSEED	A local variable used in subroutine DATAN that reads in the initial random number seed value. JSEED must be positive for TNOW to be set to TBEG.
JTRIB(3)	Buffer for attribute values being stored in or retrieved from NSET.
KRANK(NN)	KRANK(J) is the attribute row on which file J is ranked. In the GERT simulation program, KRANK(J) always is 1.
MAXNQ(NN)	MAXNQ(J) is the maximum number of entries in File J.
MFA	Identifies the first column in NSET available for storing an event or entity.
MFE(NN)	MFE(J) is the first entry in File J.
MLC(NN)	MLC(J) is the next entry in File J to be removed. If not specified, MLC(J) is set equal to MFE(J).
MLE(NN)	MLE(J) is the last entry in File J.
MSTOP	An indicator for specifying method of ending the simulation.

	If MSTOP=0, an end of simulation event must be included in which MSTOP is made negative to stop the simulation. NORPT can then be set to call SUMRY.
	If MSTOP > 0, simulation ended when TNOW ≥ TFIN.
MX	Successor row in array NSET. MX=IM+1=4
MXC	Largest number of cells to be used in any histogram (MXC ≤ 22).
MXX	Predecessor row in array NSET. MXX=IM+2=5.
NCELS(NHIST)	NCELS(J) is the number of cells in histogram J not including end cells (NCELS(J) ≤ 20)
NCLCT	The number of sets of statistics that can be collected in COLCT (NCLCT ≤ 5).
NEP	An indicator used in DATAN for initialization. NEP specifies the Data Card Type at which reading of initialization cards is to begin for the next simulation run.
NHIST	The number of histograms that can be generated by HISTO (NHIST ≤ 5)
NOQ	The number of files in filing array. (Equal to NN).
NORPT	An indicator. If NORPT >0, SUMRY and OTPUT are bypassed. If NORPT=0, SUMRY and OTPUT are used.
NOT	An indicator. If NOT=0, simulation starts from beginning. If NOT > 0, a check on NEP is made.
NPRMS	Number of sets of parameters (NPRMS ≤ 40)

NQ(NN)	NQ(J) is the current number of entries in File J.
NRUN	The number of the current simulation run.
NRUNS	The number of runs remaining, including the one being processed.
NSET(ID*5)	The integer part of the filing array. (ID limited by available storage).
NSTAT	The number of sets of statistics that can be collected in TMST. (NSTAT=0 in GERT simulation program)
OUT	An indicator. If OUT=1, an entry is to be removed from NSET. If OUT=0, an entry is to be stored in NSET.
PARAM(NPRMS,4)	Array for storing parameter values. (NPRMS $\leq$ 40)
QTIME(NN)	QTIME(J) is the time of the last use of File J.
QSET(ID*IMM)	The real valued part of the filing array. (IM $\leq$ 10, ID limited by available storage.)
SSUMA(NSTAT,J)	Array for storing time statistics generated by TMST (NSTAT $\leq$ 5).
SUMA(NCLCT,J)	Array for storing statistics generated by COLCT. Not used in GERT simulation program.
TBEG	Initial value of TNOW.
TFIN	Time to end the simulation if MSTOP > 0.
TNOW	The current time of a simulation.
VNQ(NN)	Time integrated square of the number of entries in a file.

APPENDIX B

FORTRAN Listing of GASP IIA Subprograms used in the GERT Simulation Program

```
// FOR
*LIST SOURCE PROGRAM
*ONE WORD INTEGERS
      FUNCTION AMIN  (ARG1,ARG2)                                    AMNA  10
      IF (ARG1-ARG2) 1,1,2                                          AMNA  20
    1 AMIN  = ARG1                                                  AMNA  30
      RETURN                                                        AMNA  40
    2 AMIN  = ARG2                                                  AMNA  50
      RETURN                                                        AMNA  60
      END                                                           AMNA  70

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR AMIN
 COMMON       0  VARIABLES       2  PROGRAM      32

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      FUNCTION XMAX  (IARG1,IARG2)                                  XMXA  10
      IF (IARG1 - IARG2) 2,2,1                                      XMXA  20
    1 XMAX  = IARG1                                                 XMXA  30
      RETURN                                                        XMXA  40
    2 XMAX  = IARG2                                                 XMXA  50
      RETURN                                                        XMXA  60
      END                                                           XMXA  70

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR XMAX
 COMMON       0  VARIABLES       2  PROGRAM      34

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      FUNCTION AMAX  (ARG1,ARG2)                                    AMXA  10
      IF (ARG1 - ARG2) 2,1,1                                        AMXA  20
    1 AMAX  = ARG1                                                  AMXA  30
      RETURN                                                        AMXA  40
    2 AMAX  = ARG2                                                  AMXA  50
      RETURN                                                        AMXA  60
      END                                                           AMXA  70

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR AMAX
 COMMON       0  VARIABLES       2  PROGRAM      32

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE COLCT (X,N,NSET,QSET)                                    CLTA  10
      DIMENSION NSET(1),QSET(1)                                           CLTA  20
      COMMON ID,IV,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,         CLTA  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                CLTA  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,MNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 NFE(31),NLC(31),NLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
      IF (N) 2,2,1                                                        CLTA  90
    2 CALL ERROR(90,NSET,QSET)                                            CLTA 100
    1 IF (N- NCLCT) 3,3,2                                                 CLTA 110
    3 SUMA(N,1) = SUMA(N,1)+X                                             CLTA 120
      SUMA(N,2) = SUMA(N,2)+X*X                                           CLTA 130
      SUMA(N,3) = SUMA(N,3)+1.0                                           CLTA 140
      SUMA(N,4) = AMIN (SUMA(N,4),X)                                      CLTA 150
      SUMA(N,5) = AMAX (SUMA(N,5),X)                                      CLTA 160
      RETURN                                                              CLTA 170
      END                                                                 CLTA 180

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR COLCT
 COMMON     944  VARIABLES       8  PROGRAM     124

END OF COMPILATION
```

```
// FOR
*LIST SOURCE PROGRAM
*ONE WORD INTEGERS
      SUBROUTINE DATAN(NSET,QSET)                                          DTNA  10
      DIMENSION NSET(1),QSET(1)                                            DTNA  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,          DTNA  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                 DTNA  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
      IF (NOT)23,1,2                                                       DTNA  90
C*****NEP IS A CONTROL VARIABLE FOR DETERMINING THE STARTING CARD          DTNA 100
C*****TYPE FOR MULTIPLE RUN PROBLEMS.  THE VALUE OF NEP SPECIFIES THE      DTNA 110

C*****STARTING CARD TYPE.                                                  DTNA 120
    2 NT=NEP                                                               DTNA 130
      GO TO (1,5,6,41,42,8,43,299,15,20),NT                                DTNA 140
   23 CALL ERROR(95,NSET,QSET)                                             DTNA 150
    1 NOT = 1                                                              DTNA 160
      NRUN = 1                                                             DTNA 170
C*****DATA CARD TYPE ONE                                                   DTNA 180
      READ (NCRDR,101) NAME,NPROJ,MON,NDAY,NYR,NRUNS                       DTNA 190
  101 FORMAT (6A2,I4,I2,I2,I4,I4)                                          DTNA 200
      IF(NRUNS) 30,30,5                                                    DTNA 210
   30 CALL EXIT                                                            DTNA 220
C*****DATA CARD TYPE TWO                                                   DTNA 230
    5 READ (NCRDR,803) NPRMS,NHIST,NCLCT,NSTAT,ID,IM,NOQ,MXC,IMM           DTNA 240
  803 FORMAT (9I5)                                                         DTNA 250
      IF (NHIST) 41,41,6                                                   DTNA 260
C*****DATA CARD TYPE THREE IS USED ONLY IF NHIST IS GREATER THAN ZERO       DTNA 270
C*****SPECIFY NUMBER OF CELLS IN HISTOGRAMS NOT INCLUDING END CELLS        DTNA 280
    6 READ (NCRDR,103) (NCELS(I),I=1,NHIST)                                DTNA 290
  103 FORMAT (10I5)                                                        DTNA 300
C*****DATA CARD TYPE FOUR                                                  DTNA 310
C*****SPECIFY KRANK=RANKING ROW                                            DTNA 320
C*****KRANK = 1,2,...IM FOR ROW KRANK IN QSET                              DTNA 330
C*****KRANK = 101,102,...,100+IMM FOR ROW (KRANK-100) IN NSET              DTNA 340
   41 READ (NCRDR,103) (KRANK(I),I=1,NOQ)                                  DTNA 350
C*****DATA CARD TYPE FIVE                                                  DTNA 360
C*****SPECIFY INN=1 FOR FIFO, INN=2 FOR LIFO                               DTNA 370
   42 READ (NCRDR,103) (INN(I),I=1,NOQ)                                    DTNA 380
      IF (NPRMS) 23,43,8                                                   DTNA 390
    8 DO 9 I = 1,NPRMS                                                     DTNA 400
C*****DATA CARD TYPE SIX IS USED ONLY IF NPRMS IS GREATER THAN ZERO        DTNA 410
      READ (NCRDR,106) (PARAM(I,J),J=1,4)                                  DTNA 420
  106 FORMAT(4F10.4)                                                       DTNA 430
    9 CONTINUE                                                             DTNA 440
C*****DATA CARD TYPE SEVEN.  THE NEP VALUE IS FOR THE NEXT RUN.  SET       DTNA 450
C*****JSEED GREATER THAN ZERO TO SET TNOW EQUAL TO TBEG.                   DTNA 460
   43 READ (NCRDR, 104) MSTOP,JCLR,NORPT,NEP,TBEG,TFIN,JSEED              DTNA 470
  104 FORMAT (4I5,2F10.3,I4)                                               DTNA 480
      IF (JSEED) 26,26,27                                                  DTNA 490
   27 ISEED=JSEED                                                          DTNA 500
      CALL DRAND(ISEED,RNUM)                                               DTNA 510
      TNOW = TBEG                                                          DTNA 520
      DO 142 J=1,NOQ                                                       DTNA 530
```

```
  142 QTIME(J)=TNOW                                                          DTNA 540
   26 JMNIT = 0                                                              DTNA 550
C*****INITIALIZE NSET                                                       DTNA 560
C*****SPECIFY INPUTS FOR NEXT RUN                                           DTNA 570
C*****READ IN INITIAL EVENTS                                                DTNA 580
  299 DO 300 JS = 1,ID                                                      DTNA 590
C*****DATA CARD TYPE 8                                                      DTNA 600
C*****INITIALIZE NSET BY JQ EQUAL TO A NEGATIVE VALUE ON FIRST EVENT        DTNA 610
C*****CARD                                                                  DTNA 620
C*****READ IN INITIAL EVENTS.  END INITIAL EVENTS AND ENTITIES WITH JQ      DTNA 630
C*****EQUAL TO ZERO                                                         DTNA 640
      READ (NCRDR,1110)JQ,(JTRIB(JK),JK=1,IM)                               DTNA 650
 1110 FORMAT(8I10)                                                          DTNA 660
      IF(JQ) 44,15,320                                                      DTNA 670
   44 INIT=1                                                                DTNA 680
      CALL SET(1,NSET,QSET)                                                 DTNA 690
      GO TO 300                                                             DTNA 700
  320 READ(NCRDR,1111) (ATRIB(JK),JK=1,IMM)                                 DTNA 710
 1111 FORMAT(8F10.4)                                                        DTNA 720
      CALL FILEM(JQ,NSET,QSET)                                              DTNA 730
  300 CONTINUE                                                              DTNA 740
C*****JCLR BE POSITIVE FOR INITIALIZATION OF STORAGE ARRAYS.                DTNA 750
   15 IF( JCLR )20,20,10                                                    DTNA 760
   10 IF(NCLCT)23,110,116                                                   DTNA 770
  116 DO 18 I = 1,NCLCT                                                     DTNA 780
      DO 17 J =1,3                                                          DTNA 790
   17 SUMA(I,J) = 0.                                                        DTNA 800
      SUMA(I,4) = 1.0E20                                                    DTNA 810
   18 SUMA(I,5)= -1.0E20                                                    DTNA 820
  110 IF (NSTAT)23,111,117                                                  DTNA 830
  117 DO 360 I = 1,NSTAT                                                    DTNA 840
      DO 370 J = 1,3                                                        DTNA 850
  370 SSUMA(I,J) = 0.                                                       DTNA 860
      SSUMA(I,4) = 1.0E20                                                   DTNA 870
  360 SSUMA(I,5) = -1.0E20                                                  DTNA 880
  111 IF(NHIST)23,20,118                                                    DTNA 890
  118 DO 380 K = 1,NHIST                                                    DTNA 900
      DO 380 L = 1,MXC                                                      DTNA 910
  380 JCELS(K,L) = 0                                                        DTNA 920
C*****PRINT OUT PROGRAM IDENTIFICATION INFORMATION                          DTNA 930
   20 WRITE (NPRNT,102) NPROJ,NAME,MON,NDAY,NYR,NRUN                        DTNA 940
  102 FORMAT (1H1,12X,22HSIMULATION PROJECT NO.,I4,2X,2HBY,2X,              DTNA 950
     1 6A2//,12X,4HDATE,I3,1H/,I3,1H/,I5,12X,10HRUN NUMBER,I5//)            DTNA 960
C*****PRINT PARAMETER VALUES AND SCALE                                      DTNA 970
      IF(NPRMS ) 60,60,62                                                   DTNA 980
   62 DO 64 I=1,NPRMS                                                       DTNA 990
   64 WRITE (NPRNT,107) I,(PARAM(I,J),J=1,4)                                DTNA1000
  107 FORMAT(20X,14H PARAMETER NO.,I5,4F12.4)                               DTNA1010
   60 RETURN                                                                DTNA1020
      END                                                                   DTNA1030

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR DATAN
 COMMON     944  VARIABLES      14  PROGRAM      664

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE DRAND(ISEED,RNUM)                                    DRDA  10
      CALL RANDU (ISEED,RNUM)                                         DRDA  20
      RETURN                                                          DRDA  30
      END                                                             DRDA  40

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR DRAND
 COMMON       0  VARIABLES       0  PROGRAM      14

END OF COMPILATION
```

```
      // FOR
      *ONE WORD INTEGERS
      *LIST SOURCE PROGRAM
            FUNCTION ERLNG (J)                                                ELNG  10
            COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,
           1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,
           2TBEG,TFIN,VXX,NPRNT,NCRDP,NEP,VNQ(31),IMM,MAXQS,MAXNS
            COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
           1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
           2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
            K = PARAM(J,4)                                                    ELNG  80
            IF(K-1) 8,10,10                                                   ELNG  90
          8 WRITE (3,20) J                                                    ELNG 100
         20 FORMAT(/16HK = 0 FOR ERLNG,I7)                                    ELNG 110
            CALL EXIT                                                         ELNG 120
         10 ERLNG = 0.                                                        ELNG 130
            DO 2 I = 1,K                                                      ELNG 140
            CALL DRAND (ISEED,RNUM)                                           ELNG 150
          2 ERLNG = ERLNG - PARAM(J,1)*ALOG(RNUM)                             ELNG 160
            IF(ERLNG-PARAM(J,2))7,5,6                                         ELNG 170
          7 ERLNG  = PARAM (J,2)                                              ELNG 180
          5 RETURN                                                            ELNG 190
          6 IF(ERLNG  - PARAM (J,3))5,5,4                                     ELNG 200
          4 ERLNG  = PARAM (J,3)                                              ELNG 210
            RETURN                                                            ELNG 220
            END                                                               ELNG 230

      FEATURES SUPPORTED
       ONE WORD INTEGERS

      CORE REQUIREMENTS FOR ERLNG
       COMMON     944  VARIABLES       8  PROGRAM     138

      END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE ERROR(J,NSET,QSET)                                          ERRA  10
      DIMENSION NSET(1),QSET(1)                                              ERRA  20
      COMMON ID,IV,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,            ERRA  30
     1NOQ,NORPT,NCT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                   ERRA  40
     2TREG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 NFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
      WRITE (NPRNT,100) J,TNOW                                               ERRA  90
  100 FORMAT(//36X16HERROR EXIT, TYPE,I3,7H ERROR.//21H FILE STATUS AT       ERRA 100
     1TIME,F10.4//)                                                          ERRA 110
      WRITE (NPRNT,200)                                                      ERRA 120
  200 FORMAT(30X,4HNSET/)                                                    ERRA 130
      DO 210 I=1,ID                                                          ERRA 140
      IL=(I-1)*MXX+1                                                         ERRA 150
      IV=IL+MXX-1                                                            ERRA 160
  210 WRITE(NPRNT,90) I,(NSET(IJ),IJ=IL,IV)                                  ERRA 170
   90 FORMAT(13X,I5,5X,1218)                                                 ERRA 180
      WRITE (NPRNT,202)                                                      ERRA 190
  202 FORMAT(//30X,4HQSET/)                                                  ERRA 200
      DO 215 I=1,ID                                                          ERRA 210
      IL=(I-1)*IMM+1                                                         ERRA 220
      IV=IL+IMM-1                                                            ERRA 230
  215 WRITE(NPRNT,95) I,(QSET(IJ),IJ=IL,IV)                                  ERRA 240
   95 FORMAT(13X,I5,4X,10(E12.6,2X))                                         ERRA 250
      WRITE (NPRNT,99)                                                       ERRA 260
   99 FORMAT(1H1)                                                            ERRA 270
      IF(NCLCT) 7,7,8                                                        ERRA 280
    8 WRITE (NPRNT,98)                                                       ERRA 290
   98 FORMAT(//11H ARRAY SUMA,/)                                             ERRA 300
      DO 110 I=1,NCLCT                                                       ERRA 310
  110 WRITE (NPRNT,80) I,(SUMA(I,K),K=1,5)                                   ERRA 320
   80 FORMAT(I10,5F10.4)                                                     ERRA 330
      WRITE (NPRNT,99)                                                       ERRA 340
    7 IF(NSTAT)9,9,10                                                        ERRA 350
   10 WRITE (NPRNT,97)                                                       ERRA 360
   97 FORMAT(/12H ARRAY SSUMA/)                                              ERRA 370
      DO 111 I=1,NSTAT                                                       ERRA 380
  111 WRITE (NPRNT,80) I,(SSUMA(I,K),K=1,5)                                  ERRA 390
      WRITE (NPRNT,99)                                                       ERRA 400
    9 IF(NHIST) 11,11,12                                                     ERRA 410
   12 WRITE (NPRNT,96)                                                       ERRA 420
   96 FORMAT (/12H ARRAY JCELS/)                                             ERRA 430
      DO 112 I=1,NHIST                                                       ERRA 440
      NCL=NCELS (I)+2                                                        ERRA 450
  112 WRITE (NPRNT,26) I,(JCELS(I,K),K=1,NCL)                                ERRA 460
   26 FORMAT(17X,I3,5X,2314)                                                 ERRA 470
   11 NFOOL = 0                                                              ERRA 480
      IF (NFOOL) 3,4,3                                                       ERRA 490
    3 RETURN                                                                 ERRA 500
    4 CALL EXIT                                                              ERRA 510
      END                                                                    ERRA 520

FEATURES SUPPORTED

 ONE WORD INTEGERS

CORE REQUIREMENTS FOR ERROR
 COMMON     944  VARIABLES      10  PROGRAM      422

END OF COMPILATION
```

```
// FOR
* ONE WORD INTEGERS
* LIST SOURCE PROGRAM
      SUBROUTINE FILEM (JQ,NSET,QSET)                                    FILA  10      48
      DIMENSION NSET(1),QSET(1)                                          FILA  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,        FILA  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,               FILA  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
C
C*****TEST TO SEE IF THERE IS AN AVAILABLE COLUMN FOR STORAGE            FILA  90
C
      IF (MFA - ID ) 2,2,3                                               FILA 100
    3 WRITE (NPRNT,4)                                                    FILA 110
    4 FORMAT (//24H OVERLAP SET GIVEN BELOW/)                            FILA 120
      CALL ERROR (87,NSET,QSET)                                          FILA 130
C
C*****PUT ATTRIBUTE VALUES IN FILE                                       FILA 140
C
    2 INDX = (MFA - 1) * IMM                                             FILA 150
      DO 1 I = 1,IMM                                                     FILA 160
      INDX = INDX + 1                                                    FILA 170
    1 QSET(INDX) = ATRIB(I)                                              FILA 180
      INDX = (MFA - 1) * MXX                                             FILA 190
      DO 10 I = 1,IM                                                     FILA 200
      INDX = INDX + 1                                                    FILA 210
   10 NSET(INDX) = JTRIB(I)                                              FILA 220
      CALL SET (JQ,NSET,QSET)                                            FILA 230
      RETURN                                                             FILA 240
      END                                                                FILA 250

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR FILEM
 COMMON     944  VARIABLES       4  PROGRAM     140

END OF COMPILATION
```

```
// FOR
* LIST SOURCE PROGRAM
*ONE WORD INTEGERS
      SUBROUTINE GASP(NSET,QSET)                                        GSPA  10
      DIMENSION NSET(1),QSET(1)                                         GSPA  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,       GSPA  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,              GSPA  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
      NOT = 0                                                           GSPA  90
    1 CALL DATAN(NSET,QSET)                                             GSPA 100
C
C*****PRINT OUT FILING ARRAY                                            GSPA 110
C
      JEVNT = 101                                                       GSPA 120
      CALL MONTR (NSET,QSET)                                            GSPA 130
      WRITE (NPRNT,403)                                                 GSPA 140
  403 FORMAT(1H1,38X,24H**INTERMEDIATE RESULTS**//)                     GSPA 150
C
C*****OBTAIN NEXT EVENT WHICH IS FIRST ENTRY IN FILE 1, ATRIB(1) IS      GSPA 160
C*****EVENT TIME, ATRIB(2) IS EVENT CODE                                GSPA 170
C
   10 CALL RMOVE(MFE(1),1,NSET,QSET)                                    GSPA 180
      TNOW = ATRIB(1)                                                   GSPA 190
      JEVNT = JTRIB(1)                                                  GSPA 200
C
C*****TEST TO SEE IF THIS EVENT IS A MONITOR EVENT                       GSPA 210
C
      IF(JEVNT - 100)13,12,6                                            GSPA 220
   13 I = JEVNT                                                         GSPA 230
C
C*****CALL PROGRAMMERS EVENT ROUTINES                                   GSPA 240
C
      CALL EVNTS (I,NSET,QSET)                                          GSPA 250
C
C*****TEST METHOD FOR STOPPING                                          GSPA 260
C
      IF (MSTOP) 40,8,20                                                GSPA 270
   40 MSTOP = 0                                                         GSPA 280
C
C*****TEST FOR NO SUMMARY REPORT                                        GSPA 290
C
      IF (NORPT) 14,22,42                                               GSPA 300
   20 IF(TNOW-TFIN)8,22,22                                              GSPA 310
   22 CALL SUMRY(NSET,QSET)                                             GSPA 320
      CALL OTPUT (NSET,QSET)                                            GSPA 330
C
C*****TEST NUMBER OF RUNS REMAINING                                     GSPA 340
C
   42 IF(NRUNS-1)14,9,23                                                GSPA 350
   23 NRUNS = NRUNS - 1                                                 GSPA 360
      NRUN = NRUN + 1                                                   GSPA 370
      GO TO 1                                                           GSPA 380
   14 CALL ERROR(93,NSET,QSET)                                          GSPA 390
    6 CALL MONTR(NSET,QSET)                                             GSPA 400
      GO TO 10                                                          GSPA 410
C
C*****RESET JMNIT                                                       GSPA 420
C
   12 IF(JMNIT)14,30,31                                                 GSPA 430
   30 JMNIT = 1                                                         GSPA 440
      GO TO 10                                                          GSPA 450
   31 JMNIT = 0                                                         GSPA 460
      GO TO 10                                                          GSPA 470
C
C*****TEST TO SEE IF EVENT INFORMATION IS TO BE PRINTED                 GSPA 480
C
    8 IF(JMNIT)14,10,32                                                 GSPA 490
   32 JTRIB(1) = JEVNT                                                  GSPA 500
      JEVNT = 100                                                       GSPA 510
      CALL MONTR(NSET,QSET)                                             GSPA 520
      GO TO 10                                                          GSPA 530
C
C*****IF ALL RUNS ARE COMPLETED RETURN TO MAIN PROGRAM FOR INSTRUCTIONS GSPA 540
C
    9 RETURN                                                            GSPA 550
      END                                                               GSPA 560

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR GASP
 COMMON    944  VARIABLES      2  PROGRAM    218

END OF COMPILATION
```

```
// FOR
*LIST SOURCE PROGRAM
*ONE WORD INTEGERS
      SUBROUTINE HISTO (X1,A,W,N)                                          HSTA  10
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,          HSTA  20
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                 HSTA  30
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
      IF (N-NHIST) 11,11,2                                                 HSTA  80
    2 WRITE (3,250) N                                                      HSTA  90
  250 FORMAT(19H ERROR IN HISTOGRAM,I4//)                                  HSTA 100
      CALL EXIT                                                            HSTA 110
   11 IF(N)2,2,3                                                           HSTA 120
C
C*****TRANSLATE X1 BY SUBTRACTING A IF X.LE.A THEN ADD 1 TO FIRST CELL     HSTA 130
C
    3 X = X1 - A                                                           HSTA 140
      IF (X)6,7,7                                                          HSTA 150
    6 IC = 1                                                               HSTA 160
      GO TO 8                                                              HSTA 170
C
C*****DETERMINE CELL NUMBER IC. ADD 1 FOR LOWER LIMIT CELL AND 1 FOR       HSTA 180
C*****TRUNCATION                                                           HSTA 190
C
    7 IC = X/W + 2.  +.0001                                                HSTA 200
      IF (IC - NCELS(N) - 1) 8,8,9                                         HSTA 210
    9 IC = NCELS(N)+2                                                      HSTA 220
    8 JCELS(N,IC) = JCELS(N,IC) + 1                                        HSTA 230
      RETURN                                                               HSTA 240
      END                                                                  HSTA 250

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR HISTO
 COMMON     944  VARIABLES       4  PROGRAM     122

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE MONTR(NSET,QSET)                                          MONA  10        51.
      DIMENSION NSET(1),QSET(1)                                            MONA  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,          MONA  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                 MONA  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
C
C*****IF JEVNT .GE. 101, PRINT NSET                                        MONA  90
C
      IF (JEVNT - 101) 9,7,9                                               MONA 100
    7 WRITE (NPRNT,100) TNOW                                               MONA 110
  100 FORMAT(1H1,10X31H**GASP JOB STORAGE AREA DUMP AT,F10.4,               MONA 120
     1 2X,12HTIME UNITS**//)                                               MONA 130
      WRITE (NPRNT,200)                                                    MONA 140
  200 FORMAT(30X,4HNSET/)                                                  MONA 150
      DO 210 I=1,ID                                                        MONA 160
      IL=(I-1)*MXX+1                                                       MONA 170
      IV=IL+MXX-1                                                          MONA 180
  210 WRITE(NPRNT,90) I,(NSET(IJ),IJ=IL,IV)                                MONA 190
   90 FORMAT(13X,I5,5X,12I8)                                               MONA 200
      WRITE (NPRNT,202)                                                    MONA 210
  202 FORMAT(//30X,4HQSET/)                                                MONA 220
      DO 215 I=1,ID                                                        MONA 230
      IL=(I-1)*IMM+1                                                       MONA 240
      IV=IL+IMM-1                                                          MONA 250
  215 WRITE(NPRNT,95) I,(QSET(IJ),IJ=IL,IV)                                MONA 260
   95 FORMAT(13X,I5,4X,10(E12.6,2X))                                       MONA 270
      RETURN                                                               MONA 280
    9 IF(MFE(1))3,6,1                                                      MONA 290
C
C*****IF JMNIT = 1,PRINT TNOW,CURRENT EVENT CODE, AND ALL ATTRIBUTES OF    MONA 300
C*****THE NEXT EVENT                                                       MONA 310
C
    1 IF (JMNIT - 1) 5,4,3                                                 MONA 320
    3 WRITE (NPRNT,199)                                                    MONA 330
  199 FORMAT(///36X26H ERROR EXIT,TYPE 99 ERROR.)                          MONA 340
      CALL EXIT                                                            MONA 350
    4 INDX=MFE(1)                                                          MONA 360
      IL=(INDX-1)*MXX+1                                                    MONA 370
      IV=IL+MXX-1                                                          MONA 380
      WRITE (NPRNT,103) TNOW,JTRIB(1),(NSET(I),I=IL,IV)                    MONA 390
  103 FORMAT (//20X23HCURRENT EVENT....TIME =,F8.2,5X7HEVENT =,I7,         MONA 400
     1/20X17HNEXT EVENT(NSET).,(8I10))                                     MONA 410
      IL=(INDX-1)*IMM+1                                                    MONA 420
      IV=IL+IMM-1                                                          MONA 430
      WRITE(NPRNT,120)(QSET(I),I=IL,IV)                                    MONA 440
  120 FORMAT(/20X19HNEXT EVENT(QSET)...,(6E12.4))                          MONA 450
    5 RETURN                                                               MONA 460
    6 WRITE (NPRNT,104) TNOW                                               MONA 470
  104 FORMAT (10X,19H FILE 1 IS EMPTY AT,F10.2)                            MONA 480
      GO TO 5                                                              MONA 490
      END                                                                  MONA 500

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR MONTR
 COMMON     944  VARIABLES       8  PROGRAM     420

END OF COMPILATION
```

```
// FOR
*LIST SOURCE PROGRAM
*ONE WORD INTEGERS
      SUBROUTINE NPOSN(J,NPSSN)                                          PSNA  10
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,        PSNA  20
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,               PSNA  30
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLF(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
      NPSSN = 0                                                          PSNA  80
      P = PARAM (J,1)                                                    PSNA  90
      IF (P-6.0) 2,2,4                                                   PSNA 100
    2 Y = EXP (-P)                                                       PSNA 110
      X = 1.0                                                            PSNA 120
    3 CALL DRAND(ISEED,RNUM)                                             PSNA 130
      X=X*RNUM                                                           PSNA 140
      IF (X-Y) 6,8,8                                                     PSNA 150
    8 NPSSN = NPSSN+1                                                    PSNA 160
      GO TO 3                                                            PSNA 170
    4 TEMP=PARAM (J,4)                                                   PSNA 180
      PARAM(J,4) = (PARAM(J,1))**.5                                      PSNA 190
      NPSSN=RNORM(J)                                                     PSNA 200
      PARAM (J,4)=TEMP                                                   PSNA 210
      IF(NPSSN)4,6,6                                                     PSNA 220
    6 KK=PARAM (J,2)                                                     PSNA 230
      KKK=PARAM (J,3)                                                    PSNA 240
      NPSSN=KK+NPSSN                                                     PSNA 250
      IF(NPSSN-KKK)7,7,9                                                 PSNA 260
    9 NPSSN = PARAM (J,3)                                                PSNA 270
    7 RETURN                                                             PSNA 280
      END                                                                PSNA 290

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR NPOSN
 COMMON     944  VARIABLES     16  PROGRAM     172

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE PRNTQ (JQ,NSET,QSET)                                    PRQA  10
      DIMENSION NSET(1),QSET(1)                                          PRQA  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,        PRQA  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,               PRQA  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
      WRITE (NPRNT,100) JQ                                               PRQA  90
      IF (TNOW - TBEG) 12,12,13                                          PRQA 100
   12 WRITE (NPRNT,105)                                                  PRQA 110
  105 FORMAT(/35X,25H NO PRINTOUT TNOW = TBEG //)                        PRQA 120
      GO TO 2                                                            PRQA 130
C
C*****COMPUTE EXPECT NO. IN FILE JQ UP TO PRESENT THIS MAY BE USEFUL     PRQA 140
C*****IN SETTING THE VALUE OF ID                                         PRQA 150
C
   13 XNQ=NQ(JQ)                                                         PRQA 160
      X=(ENQ(JQ)+XNQ*(TNOW-QTIME(JQ)))/(TNOW-TBEG)                       PRQA 170
      STD=((VNQ(JQ)+XNQ*XNQ*(TNOW-QTIME(JQ)))/(TNOW-TBEG)-X*X)**0.5      PRQA 180
      WRITE (NPRNT,104) X,STD,MAXNQ(JQ)                                  PRQA 190
      WRITE (NPRNT,101)                                                  PRQA 200
C
C*****PRINT FILE IN PROPER ORDER REQUIRES TRACING THROUGH THE POINTERS   PRQA 210
C*****OF THE FILE                                                        PRQA 220
C
      NSQ = 1                                                            PRQA 230
      WRITE(NPRNT,200)                                                   PRQA 240
  200 FORMAT(30X,4HNSET/)                                                PRQA 250
  230 LINE = MFE(JQ)                                                     PRQA 260
      IF (LINE-1) 4,1,1                                                  PRQA 270
    4 WRITE (NPRNT,102)                                                  PRQA 280
    2 RETURN                                                             PRQA 290
    1 L1 = LINE - 1                                                      PRQA 300
      GO TO (202,201),NSQ                                                PRQA 310
  202 INDX = L1 * MXX                                                    PRQA 320
      IB = INDX + 1                                                      PRQA 330
      IE = INDX + MXX                                                    PRQA 340
      WRITE (NPRNT,106) LINE, (NSET(I),I=IB,IE)                          PRQA 350
      GO TO 210                                                          PRQA 360
  201 INDX = L1 * IMM                                                    PRQA 370
      IB = INDX + 1                                                      PRQA 380
      IE = INDX + IMV                                                    PRQA 390
      WRITE (NPRNT,103) LINE, (QSET(I),I=IB,IE)                          PRQA 400
  210 INDX = LINE * MXX - 1                                              PRQA 410
      LINE = NSET(INDX)                                                  PRQA 420
      IF (LINE-7777) 1,2220,5                                            PRQA 430
 2220 IF (NSQ-2) 221,2,2                                                 PRQA 440
  221 NSQ = NSQ + 1                                                      PRQA 450
      WRITE (NPRNT,205)                                                  PRQA 460
  205 FORMAT (//30X,4HQSET/)                                             PRQA 470
      GO TO 230                                                          PRQA 480
    5 WRITE (NPRNT,199)                                                  PRQA 490
  199 FORMAT(///36X26HERROR EXIT, TYPE 94 ERROR.)                        PRQA 500

  100 FORMAT(//39X25H FILE PRINTOUT, FILE  NO.,I3)                       PRQA 510
  101 FORMAT (/45X14H FILE CONTENTS//)                                   PRQA 520
  102 FORMAT(/43X18HTHE FILE  IS EMPTY//)                                PRQA 530
  103 FORMAT (13X,I5,4X,10(E12.6,2X))                                    PRQA 540
  104 FORMAT(/35X,27HAVERAGE NUMBER IN FILE  WAS,F10.4,/35X,9HSTD. DEV.,PRQA 550
     1 18X,F10.4,/35X,7HMAXIMUM,24X,I4)                                  PRQA 560
  106 FORMAT (13X,I5,5X,12I8)                                            PRQA 570
      CALL EXIT                                                          PRQA 580
      END                                                                PRQA 590
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE RMOVE  (KCOLL,JQ,NSET,QSET)                          RMVA  10      54.
      DIMENSION NSET(1),QSET(1),KCOLL(1)                              RMVA  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,     RMVA  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,            RMVA  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
C
C*****THE DUMMY ARRAY KCOLL IS USED AS AN ARGUMENT TO FORCE THE CALL   RMVA  90
C*****    BY NAME OPTION ON COMPUTERS SUCH AS THE IBM 360.  THE CALL   RMVA 100
C*****    BY NAME OPTION IS REQUIRED FOR PROPER OPERATION OF THIS      RMVA 110
C*****    SUBROUTINE.                                                  RMVA 120
C
      KCOL = KCOLL(1)                                                 RMVA 130
      IF (KCOL) 16,16,2                                               RMVA 140
   16 CALL ERROR (97,NSET,QSET)                                       RMVA 150
    2 MLC(JQ) = KCOL                                                  RMVA 160
C
C*****PUT VALUES OF KCOL IN ATTRIB                                    RMVA 170
C
      INDX = (KCOL - 1) * IMM                                         RMVA 180
      DO 3 I = 1,IMM                                                  RMVA 190
      INDX = INDX + 1                                                 RMVA 200
    3 ATRIB(I) = QSET(INDX)                                           RMVA 210
      INDX = (KCOL - 1) * MXX                                         RMVA 220
C
      DO 10 I =1,IM                                                   RMVA 230
      INDX = INDX + 1                                                 RMVA 240
   10 JTRIB(I) = NSET(INDX)                                           RMVA 250
C*****SET OUT=1 AND CALL SET TO REMOVE ENTRY FROM NSET                RMVA 260
C
      OUT = 1.                                                        RMVA 270
      CALL SET (JQ,NSET,QSET)                                         RMVA 280
      RETURN                                                          RMVA 290
      END                                                             RMVA 300

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR RMOVE
 COMMON     944  VARIABLES      6  PROGRAM     142

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      FUNCTION RNORM (J)                                                    NORM  10    55.
      COMMON ID,IV,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,MNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
      CALL DRAND(ISEED,RA)                                                  NORM  80
      CALL DRAND(ISEED,RB)                                                  NORM  90
      V=(-2.0*ALOG(RA))**0.5*COS (6.283*RB)                                 NORM 100
      RNORM  = V*PARAM (J,4) + PARAM (J,1)                                  NORM 110
      IF (RNORM -PARAM (J,2)) 6,7,8                                         NORM 120
    6 RNORM  = PARAM (J,2)                                                  NORM 130
    7 RETURN                                                                NORM 140
    8 IF (RNORM -PARAM (J,3)) 7,7,9                                         NORM 150
    9 RNORM  = PARAM (J,3)                                                  NORM 160
      RETURN                                                                NORM 170
      END                                                                   NORM 180

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR RNORM
 COMMON     944  VARIABLES      18  PROGRAM     122

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      FUNCTION RLOGN (J)                                                    LOGN  10
C
C*****THE PARAMETERS USED WITH RLOGN ARE THE MEAN AND STANDARD DEVIATIONLOGN  15
C*****OF A NORMAL DISTRIBUTION                                              LOGN  16
C
      VA= RNORM (J)                                                         LOGN  20
      RLOGN=EXP(VA)                                                         LOGN  30
      RETURN                                                                LOGN  40
      END                                                                   LOGN  50

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR RLOGN
 COMMON       0  VARIABLES       4  PROGRAM      20

END OF COMPILATION
```

```
// FOR
* ARITHMETIC TRACE
* TRANSFER TRACE
* LIST SOURCE PROGRAM
*ONE WORD INTEGERS
      SUBROUTINE SET (JQ,NSET,QSET)                                    SETA  10
      DIMENSION NSET(1),QSET(1)                                        SETA  20
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,      SETA  30
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,             SETA  40
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(31),IMM,MAXQS,MAXNS
      COMMON ATRIB(1),ENQ(31),INN(31),JCELS(5,22),KRANK(31),MAXNQ(31),
     1 MFE(31),MLC(31),MLE(31),NCELS(5),NQ(31),PARAM(40,4),QTIME(31),
     2 SSUMA(1,1),SUMA(5,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(5)
C
C*****HE SINGLE ARRAY NSET OF FIXED POINT GASP IS REPLACED BY THE TWO   SETA  90
C*****  ARRAYS QSET AND NSET.  FOR FILE 1, ATRIB(1) IS THE EVENT TIME   SETA 100
C*****  AND JTRIB(1) THE EVENT CODE FOR AN EVENT.  SUCCESSOR AND PREDEC- SETA 110
C*****  ESSOR POINTERS ARE STORED AS USUAL IN NSET.  QSET AND NSET ARE  SETA 120
C*****  ONE DIMENSIONAL ARRAYS BUT MAY BE THOUGHT OF AS TWO DIMENSIONAL SETA 130
C*****  ARRAYS WITH BOTH NUMBER OF ROWS AND COLUMNS VARIABLE AT EXECU-  SETA 140
C*****  TION TIME.  FUNCTION LOCAT WILL LINK (ROW,COL) TO (INDEX).      SETA 150
C
      IF (INIT-1) 27,28,27                                             SETA 160
C
C*****INITIALIZE FILE TO ZERO.  SET UP POINTERS                        SETA 170
C*****MUST INITIALIZE KRANK(JQ)                                        SETA 180
C*****MUST INITIALIZE INN(JQ)****INN(JQ)=1 IS FIFO**INN(JQ)=2 IS LIFO  SETA 190
C
   28 KOL = 7777                                                       SETA 200
      KOF = 8888                                                       SETA 210
      KLE = 9999                                                       SETA 220
      MX  = IM +1                                                      SETA 230
      MXX=IM+2                                                         SETA 240
      MAXQS = ID * IMM                                                 SETA 250
      MAXNS = ID * MXX                                                 SETA 260
C
C*****INITIALIZE POINTERS IN NSET AND ZERO OTHER CELLS IN QSET AND NSET SETA 270
C
      DO 2 J = 1,MAXQS                                                 SETA 280
    2 QSET(J) = 0.0                                                    SETA 290
      DO 4 J = 1,MAXNS                                                 SETA 300
    4 NSET(J) = 0                                                      SETA 310
      DO 1 I = 1,ID                                                    SETA 320
      INDX = I * MXX                                                   SETA 330
      NSET(INDX - 1) = I + 1                                           SETA 340
    1 NSET(INDX) = I - 1                                               SETA 350
      NSET(MAXNS - 1) = KOF                                            SETA 360
      DO 3 K = 1,NOQ                                                   SETA 370
      NQ(K)=0                                                          SETA 380
      MLC(K)=0                                                         SETA 390
      MFE(K)=0                                                         SETA 400
      MAXNQ(K) = 0                                                     SETA 410
      MLE(K)=0                                                         SETA 420
      ENQ(K)=0.0                                                       SETA 430
      VNQ(K)=0.0                                                       SETA 440
    3 QTIME(K)=TNOW                                                    SETA 450
C
C*****FIRST AVAILABLE COLUMN = 1                                       SETA 460
C
      MFA = 1                                                          SETA 470
      INIT = 0                                                         SETA 480
      OUT = 0.0                                                        SETA 490
      RETURN                                                           SETA 500
C
C*****MFEX IS FIRST ENTRY IN FILE WHICH HAS NOT BEEN COMPARED WITH ITEM SETA 510
C*****TO BE INSERTED                                                   SETA 520
C
   27 MFEX = MFE(JQ)                                                   SETA 530
C
C*****KNT IS A CHECK CODE TO INDICATE THAT NO COMPARISONS HAVE BEEN MADESETA 540
C
      KNT = 2                                                          SETA 550
C
C*****KS IS THE ROW ON WHICH ITEMS OF FILE JQ ARE RANKED               SETA 560
C
      KS = KRANK(JQ)                                                   SETA 570
      KSJ = 1                                                          SETA 580
      IF (KS - 100) 1020,100,1000                                      SETA 590
 1000 KSJ = 2                                                          SETA 600
      KS = KS - 100                                                    SETA 610
C
C*****TEST FOR PUTTING VALUE IN OR OUT                                 SETA 620
C*****IF OUT EQUALS ONE AN ITEM IS TO BE REMOVED FROM FILE JQ.  IF OUT  SETA 630
C*****IS LESS THAN ONE AN ITEM IS TO BE INSERTED IN FILE JQ             SETA 640
C
 1020 IF (OUT - 1.0) 8,5,100                                           SETA 650
```

```
C
C*****PUTTING AN ENTRY IN FILE JQ                                       SETA 660
C*****THE ITEM TO BE INSERTED WILL BE PUT IN COLUMN MFA                 SETA 670
C
    8 INDX = MFA * MXX - 1                                              SETA 680
      NXFA = NSET(INDX)                                                 SETA 690
C
C*****IF INN(JQ) EQUALS TWO THE FILE IS A LIFO FILE.  IF INN(JQ) IS     SETA 700
C*****ONE THE FILE IS A FIFO FILE.  FOR FIFO FILES TRY TO INSERT         SETA 710
C*****STARTING AT END OF FILE.  MLEX IS LAST ENTRY IN FILE WHICH HAS     SETA 720
C*****NOT BEEN COMPARED WITH ITEMS TO BE INSERTED.                      SETA 730
C
      IF (INN(JQ)-1) 100,7,6                                            SETA 740
    7 MLEX=MLE(JQ)                                                      SETA 750
C
C*****IF MLEX IS ZERO FILE IS EMPTY.  ITEM TO BE INSERTED WILL BE ONLY  SETA 760
C*****ITEM IN FILE.                                                     SETA 770
C
      IF (MLEX) 100,10,11                                               SETA 780
   10 INDX = MFA * MXX                                                  SETA 790
      NSET(INDX) = KLE                                                  SETA 800
      MFE(JQ) = MFA                                                     SETA 810
C
C*****THERE IS NO SUCCESSOR OF ITEM INSERTED.  SINCE ITEM WAS INSERTED  SETA 820
C*****IN COLUMN MFA THE LAST ENTRY OF FILE JQ IS IN COLUMN MFA.         SETA 830
C
   17 INDX = MFA * MXX - 1                                              SETA 840
      NSET(INDX) = KOL                                                  SETA 850
      MLE(JQ) = MFA                                                     SETA 860
C
C*****SET NEW MFA EQUAL TO SUCCESSOR OF OLD MFA.  THAT IS NXFA.  THE    SETA 870
C*****NEW MFA HAS NO PREDECESSOR SINCE IT IS THE FIRST AVAILABLE COLUMN SETA 880
C*****FOR STORAGE.                                                      SETA 890
C
   14 MFA =NXFA                                                         SETA 900
      IF(MFA-KOF) 237,238,238                                           SETA 905
  237 INDX=NXFA*MXX                                                     SETA 910
      NSET(INDX) = KLE                                                  SETA 920
C
C*****UPDATE STATISTICS OF FILE JQ                                      SETA 930
C
  238 XNQ=NQ(JQ)                                                        SETA 940
      ENQ(JQ) = ENQ(JQ)+XNQ*(TNOW-QTIME(JQ))                           SETA 950
      VNQ(JQ)=VNQ(JQ)+XNQ*XNQ*(TNOW-QTIME(JQ))                         SETA 960
      QTIME(JQ) = TNOW                                                  SETA 970
      NQ(JQ) = NQ(JQ) + 1                                               SETA 980
      MAXNQ(JQ)=XMAX(MAXNQ(JQ),NQ(JQ))                                 SETA 990
      MLC(JQ)=MFE(JQ)                                                   SETA1000
      RETURN                                                            SETA1010
C
C*****TEST TO DETERMINE IF RANKING ATTRIBUTE IS IN QSET (KRANK.LT.100)  SETA1020
C*****OR NSET (KRANK .GT.100).                                          SETA1030
C
   11 GO TO (1100,1120),KSJ                                             SETA1040
 1100 INDX1 = (MFA - 1) * IMM + KS                                      SETA1050
      INDX2 = (MLEX - 1) * IMM + KS                                     SETA1060
      IF (QSET(INDX1) - QSET(INDX2)) 12,13,13                           SETA1070
 1120 INDX1 = (MFA - 1) * MXX + KS                                      SETA1080
      INDX2 = (MLEX - 1) * MXX + KS                                     SETA1090
C
C*****TEST RANKING VALUE OF NEW ITEM AGAINST VALUE OF ITEM IN COLUMN    SETA1100
C*****MLEX                                                              SETA1110
C
      IF (NSET(INDX1) - NSET(INDX2)) 12,13,13                           SETA1120
C
C*****INSERT ITEM AFTER COLUMN MLEX.  LET SUCCESSOR OF MLEX BE MSU.     SETA1130
C
   13 INDX = MLEX * MXX - 1                                             SETA1140
      MSU = NSET(INDX)                                                  SETA1150
      NSET(INDX) = MFA                                                  SETA1160
      INDX = MFA * MXX                                                  SETA1170
      NSET(INDX) = MLEX                                                 SETA1180
      GO TO (18,17),KNT                                                 SETA1190
C
C*****SINCE KNT EQUALS ONE A COMPARISON WAS MADE AND THERE IS A          SETA1200
C*****SUCCESSOR TO MLEX, I.E., MSU IS NOT EQUAL TO KOL.  POINT COLUMN   SETA1210
C*****MFA TO MSU AND VICE VERSA.                                        SETA1220
C
   18 INDX = MFA * MXX - 1                                              SETA1230
      NSET(INDX) = MSU                                                  SETA1240
```

```
      INDX = MSU * MXX                                                  SETA1250
      NSET(INDX) = MFA                                                  SETA1260
      GO TO 14                                                          SETA1270
C
C*****SET KNT TO ONE SINCE A COMPARISON WAS MADE.                       SETA1280
C
   12 KNT = 1                                                           SETA1290
C
C*****TEST MFA AGAINST PREDECESSOR OF MLEX BY LETTING MLEX EQUAL        SETA1300
C*****PREDECESSOR OF MLEX.                                             SETA1310
C
      INDX = MLEX * MXX                                                 SETA1320
      MLEX = NSET(INDX)                                                 SETA1330
      IF(MLEX-KLE)  11,16,11                                            SETA1340
C
C*****IF MLEX HAD NO PREDECESSOR MFA IS FIRST IN FILE.                  SETA1350
C
   16 INDX = MFA * MXX                                                  SETA1360
      NSET(INDX) = KLE                                                  SETA1370
      MFE(JQ) = MFA                                                     SETA1380
C
C*****SUCCESSOR OF MFA IS MFEX AND PREDECESSOR OF MFEX IS MFA. (NOTE AT SETA1390
C*****THIS POINT MLEX = MFEX IF FIFO WAS USED).                         SETA1400
C
   26 INDX = MFA * MXX - 1                                              SETA1410
      NSET(INDX) = MFEX                                                 SETA1420
      INDX = MFEX * MXX                                                 SETA1430
      NSET(INDX) = MFA                                                  SETA1440
      GO TO 14                                                          SETA1450
C
C*****FOR LIFO OPERATION TRY TO INSERT ITEM STARTING AT BEGINNING OF    SETA1460
C*****FILE JQ.                                                         SETA1470
C*****IF MFEX IS 0, NO ENTRIES ARE IN FILE JQ.  THIS CASE WAS CONSIDEREDSETA1480
C*****PREVIOUSLY AT STATEMENT 10.                                      SETA1490
C
    6 IF (MFEX) 100,10,19                                               SETA1500
C
C*****TEST RANKING VALUE OF NEW ITEM AGAINST VALUE OF ITEM IN COLUMN    SETA1510
C*****MFEX.                                                            SETA1520
C
   19 GO TO (1200,1220),KSJ                                             SETA1530
 1200 INDX1 = (MFA - 1) * IMM + KS                                      SETA1540
      INDX2 = (MFEX - 1) * IMM + KS                                     SETA1550
      IF (QSET(INDX1) - QSET(INDX2)) 20,21,21                           SETA1560
 1220 INDX1 = (MFA - 1) * MXX + KS                                      SETA1570
      INDX2 = (MFEX - 1) * MXX + KS                                     SETA1580
      IF (NSET(INDX1) - NSET(INDX2)) 20,21,21                           SETA1590
C
C*****IF NEW VALUE IF LOWER, MFA MUST BE COMPARED AGAINST SUCCESSOR OF  SETA1600
C*****MFEX.                                                            SETA1610
C
   20 KNT = 1                                                           SETA1620
C
C*****LET MPRE = MFEX AND LET MFEX BE THE SUCCESSOR OF MFEX.            SETA1630
C
      MPRE = MFEX                                                       SETA1640
      INDX = MFEX * MXX - 1                                             SETA1650
      MFEX = NSET(INDX)                                                 SETA1660
      IF (MFEX-KOL)  19,24,19                                           SETA1670
C
C*****IF NEW VALUE IS HIGHER, IT SHOULD BE INSERTED BETWEEN MFEX AND ITSSETA1680
C*****PREDECESSOR.                                                     SETA1690
C*****IF KNT = 2, MFEX HAS NO PREDECESSOR.  GO TO STATEMENT 16.  IF KNT SETA1700
C*****= 1, A COMPARISON WAS MADE AND A VALUE OF MPRE HAS ALREADY BEEN   SETA1710
C*****OBTAINED ON THE PREVIOUS ITERATION.  SET KNT = 2 TO INDICATE THIS.SETA1720
C
   21 GO TO (22,16),KNT                                                 SETA1730
   22 KNT = 2                                                           SETA1740
C
C*****MFA IS TO BE INSERTED AFTER MPRE.  MAKE MPRE THE PREDECESSOR OF   SETA1750
C*****MFA AND MFA THE SUCCESSOR OF MPRE.                                SETA1760
C
   24 INDX = MFA * MXX                                                  SETA1770
      NSET(INDX) = MPRE                                                 SETA1780
      INDX = MPRE * MXX - 1                                             SETA1790
      NSET(INDX) = MFA                                                  SETA1800
C
C*****IF KNT WAS NOT RESET TO 2, THERE IS NO SUCCESSOR OF MFA.  POINTERSSETA1810
C*****ARE UPDATED AT STATEMENT 17.  IF KNT = 2, IT WAS RESET AND THE    SETA1820
C*****SUCCESSOR OF MFA IS MFEX.                                         SETA1830
C
      GO TO (17,26), KNT                                                SETA1840
C
C*****REMOVAL OF AN ITEM FROM FILE JQ.                                  SETA1850
C*****RESET OUT TO 0 AND CLEAR COLUMN REMOVED.  LET JL EQUAL SUCCESSOR  SETA1860
C*****OF COLUMN REMOVED AND JK EQUAL PREDECESSOR OF COLUMN REMOVED.     SETA1870
C*****IF JL = KOL, MLC WAS LAST ENTRY.  IF JK = KLE, MLC WAS FIRST ENTRYSETA1880
C*****MLC WAS NOT FIRST OR LAST ENTRY.  UPDATE POINTERS SO THAT JL IS   SETA1890
C*****SUCCESSOR OF JK AND JK IS PREDECESSOR OF JL.                      SETA1900
C
```

```
    5 OUT = 0.0                                                          SETA1910
C
C*****UPDATE POINTING SYSTEM TO ACCOUNT FOR REMOVAL OF MLC (JQ).  COLUMNSETA1920
C*****REMOVED IS ALWAYS SET TO MLC(JQ) BY SUBROUTINE RMOVE.              SETA1930
C
      INDX = (MLC(JQ) - 1) * IMM                                         SETA1940
      DO 32 I=1,IMM                                                      SETA1950
      INDX = INDX + 1                                                    SETA1960
   32 QSET(INDX) = 0.0                                                   SETA1970
      INDX = (MLC(JQ) - 1) * MXX                                         SETA1980
      DO 1300 I= 1,IM                                                    SETA1990
      INDX = INDX + 1                                                    SETA2000
 1300 NSET(INDX) = 0                                                     SETA2010
      INDX = MLC(JQ) * MXX                                               SETA2020
      JL = NSET(INDX - 1)                                                SETA2030
      JK = NSET(INDX)                                                    SETA2040
      IF (JL- KOL) 33,34,33                                              SETA2050
   33 IF (JK- KLE) 35,36,35                                              SETA2060
   35 INDX = JK * MXX - 1                                                SETA2070
      NSET(INDX) = JL                                                    SETA2080

      INDX = JL * MXX                                                    SETA2090
      NSET(INDX) = JK                                                    SETA2100
C
C*****UPDATE POINTERS.                                                   SETA2110
C
   37 INDX = MLC(JQ) * MXX - 1                                           SETA2120
      NSET(INDX) = MFA                                                   SETA2130
      NSET(INDX+1) = KLE                                                 SETA2140
      INDX=MFA*MXX                                                       SETA2150
      NSET(INDX)=MLC(JQ)                                                 SETA2160
      MFA = MLC(JQ)                                                      SETA2170
      MLC(JQ) = MFE(JQ)                                                  SETA2180
C
C*****UPDATING FILE STATISTICS                                           SETA2190
C
      XNQ = NQ(JQ)                                                       SETA2200
      ENQ(JQ)=ENQ(JQ)+XNQ*(TNOW-QTIME(JQ))                              SETA2210
      VNQ(JQ)=VNQ(JQ)+XNQ*XNQ*(TNOW-QTIME(JQ))                          SETA2220
      QTIME(JQ) = TNOW                                                   SETA2230
      NQ(JQ) = NQ(JQ)-1                                                  SETA2240
      RETURN                                                             SETA2250
C
C*****MLC WAS FIRST ENTRY BUT NOT LAST ENTRY.  UPDATE POINTERS.          SETA2260
C
   36 INDX = JL * MXX                                                    SETA2270
      NSET(INDX) = KLE                                                   SETA2280
      MFE(JQ) = JL                                                       SETA2290
      GO TO 37                                                           SETA2300
   34 IF (JK-KLE)  38,39,38                                              SETA2310
C
C*****MLC WAS LAST ENTRY BUT NOT FIRST ENTRY.  UPDATE POINTERS.          SETA2320
C
   38 INDX = JK *MXX - 1                                                 SETA2330
      NSET(INDX) = KOL                                                   SETA2340
      MLE(JQ) = JK                                                       SETA2350
      GO TO 37                                                           SETA2360
C
C*****MLC WAS BOTH THE LAST AND FIRST ENTRY, THEREFORE, IT IS THE ONLY   SETA2370
C*****ENTRY.                                                             SETA2380
C
   39 MFE(JQ) = 0                                                        SETA2390
      MLE(JQ) = 0                                                        SETA2400
      GO TO 37                                                           SETA2410
  100 CALL ERROR (88,NSET,QSET)                                          SETA2420
      RETURN                                                             SETA2430
      END                                                                SETA2440

FEATURES SUPPORTED
 TRANSFER TRACE
 ARITHMETIC TRACE
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR SET
 COMMON     944  VARIABLES     28  PROGRAM    1388

END OF COMPILATION
```

```
// FOR
*LIST SOURCE PROGRAM
*ONE WORD INTEGERS
      FUNCTION UNFRM (A,B)                                                    UNFA  10
      COMMON ID,IM,INIT,JEVNT,JMNIT,MFA,MSTOP,MX,MXC,NCLCT,NHIST,             UNFA  20
     1NOQ,NORPT,NOT,NPRMS,NRUN,NRUNS,NSTAT,OUT,ISEED,TNOW,                    UNFA  30
     2TBEG,TFIN,MXX,NPRNT,NCRDR,NEP,VNQ(4),IMM,MAXQS,MAXNS                    UNFA  40
      COMMON ATRIB(10),ENQ(4),INN(4),JCELS(5,22),KRANK(4),MAXNQ(4),M          UNFA  50
     1FE(4),MLC(4),MLE(4),NCELS(5),NQ(4),PARAM(20,4),QTIME(4),SSUMA           UNFA  60
     2(10,5),SUMA(10,5),NAME(6),NPROJ,MON,NDAY,NYR,JCLR,JTRIB(12)             UNFA  70
      CALL DRAND (ISEED,RNUM)                                                 UNFA  80
      UNFRM  = A+(B-A)*RNUM                                                   UNFA  90
      RETURN                                                                  UNFA 100
      END                                                                     UNFA 110

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR UNFRM
 COMMON     608  VARIABLES       4  PROGRAM      26

END OF COMPILATION
```