8B

BNWL-2146
UC-32

RAFT: A Computer Program for Fault Tree Risk Calculations

By
G. D. Seybold

November 1977

Prepared for the
U.S. Department of Energy
under Contract EY-76-C-06-1830

BNWL-2146

Battelle
Pacific Northwest Laboratories

PACIFIC NORTHWEST LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
Under Contract EY-76-C-06-1830

Printed in the United States of America
Available from
National Technical Information Service
United States Department of Commerce
5285 Port Royal Road
Springfield, Virginia 22151

Price: Printed Copy $____*; Microfiche $3.00

*Pages	NTIS Selling Price
001-025	$4.50
026-050	$5.00
051-075	$5.50
076-100	$6.00
101-125	$6.50
126-150	$7.00
151-175	$7.75
176-200	$8.50
201-225	$8.75
226-250	$9.00
251-275	$10.00
276-300	$10.25

BNWL-2146
UC-32

RAFT: A COMPUTER PROGRAM FOR FAULT TREE RISK CALCULATIONS

by
G. D. Seybold

November 1977

BATTELLE
Pacific Northwest Laboratories
Richland, Washington 99352

CONTENTS

ACKNOWLEDGMENTS. vii

SUMMARY ix

1.0 INTRODUCTION 1

2.0 INFORMATION FOR THE USER 3

2.1 GENERAL DESCRIPTION OF THE PROGRAM 3

2.2 INPUT INSTRUCTIONS FOR RAFT 5

2.3 DESCRIPTION OF OUTPUT 11

2.4 SAMPLE CASE. 11

3.0 PROCEDURE FOR SOLUTION. 19

3.1 CALCULATION OF THE RISK MEASURE 19

3.2 REDUNDANCY CHECKING 21

4.0 ADDITIONAL PROGRAMMING INFORMATION. 23

4.1 GENERAL INFORMATION 23

4.2 SUBROUTINE DESCRIPTIONS. 23

4.3 CALCULATIONAL DEPENDENCY 31

4.4 MACHINE DEPENDENCY 32

4.5 EXTERNAL FILES. 33

4.6 CASE REQUIREMENTS. 35

4.7 TIMING 35

REFERENCES 37

APPENDIX A - NAMELIST. 39

APPENDIX B - RAFT CROSS-REFERENCE TABLE. 47

FIGURES

1 Risk Assessment Computer Package 2

2 Flow Chart for RAFT. 24

3 Flow Chart for RECORD 26

4 Flow Chart for WEED. 29

ACKNOWLEDGMENTS

Expertise concerning fault tree methodology was extended primarily by T. H. Smith, W. K. Winegardner, and P. J. Pelto. Their concern and direction were the principal motivating factors behind the development of this method to fault tree analysis.

Effective utilization of the hashing algorithms and the NAMELIST input package was accomplished through the helpful suggestions of K. B. Stewart and B. H. Duane.

SUMMARY

A description and user instructions are presented for RAFT, a FORTRAN computer code for calculation of a risk measure for fault tree cut sets. RAFT calculates release quantities and a risk measure based on the product of probability and release quantity for cut sets of fault trees modeling the accidental release of radioactive material from a nuclear fuel cycle facility.

Cut sets and their probabilities are supplied as input to RAFT from an external fault tree analysis code. Using the total inventory available of radioactive material, along with release fractions for each event in a cut set, the release terms are calculated for each cut set. Each release term is multiplied by the cut set probability yielding the cut set risk measure. RAFT orders the dominant cut sets on the risk measure. The total risk measure of processed cut sets and their fractional contributions are supplied as output.

Input options are available to eliminate redundant cut sets, apply threshold values on cut set probability and risk, and control the total number of cut sets output. Hash addressing is used to remove redundant cut sets from the analysis.

Computer hardware and software restrictions are given along with a sample problem and cross-reference table of the code. Except for the use of file management utilities, RAFT is written exclusively in FORTRAN language and is operational on a Control Data, CYBER 74-18, series computer system.

1.0 INTRODUCTION

A risk-based fault tree analysis method has been developed at Pacific Northwest Laboratories (PNL) for analysis of nuclear fuel cycle operations.(1) This method was developed for the Energy Research and Development Administration (ERDA)-sponsored risk analysis of systems for managing high-level waste. A series of three computer codes has been developed to assist in the performance of a risk assessment: ACORN(2) (plots fault trees), MFAULT(3) (analyzes fault trees), and RAFT (calculates risk measures). Figure 1 gives a summary of the input, output, and interrelationships of these programs.

This report gives a description and user instructions for RAFT, a computer code for calculation of a risk measure for fault tree cut sets. Section 2 gives a basic description of the code and input and output information while Sections 3 and 4 discuss the calculational procedures and additional programming details, respectively.

Reference 1 provides a background discussion of the application of RAFT to assist in performing a risk assessment. Some familiarity with fault tree methodology is assumed. References 1, 4, and 5 will supply the reader a comprehensive overview of systems safety analysis and fault tree analysis.

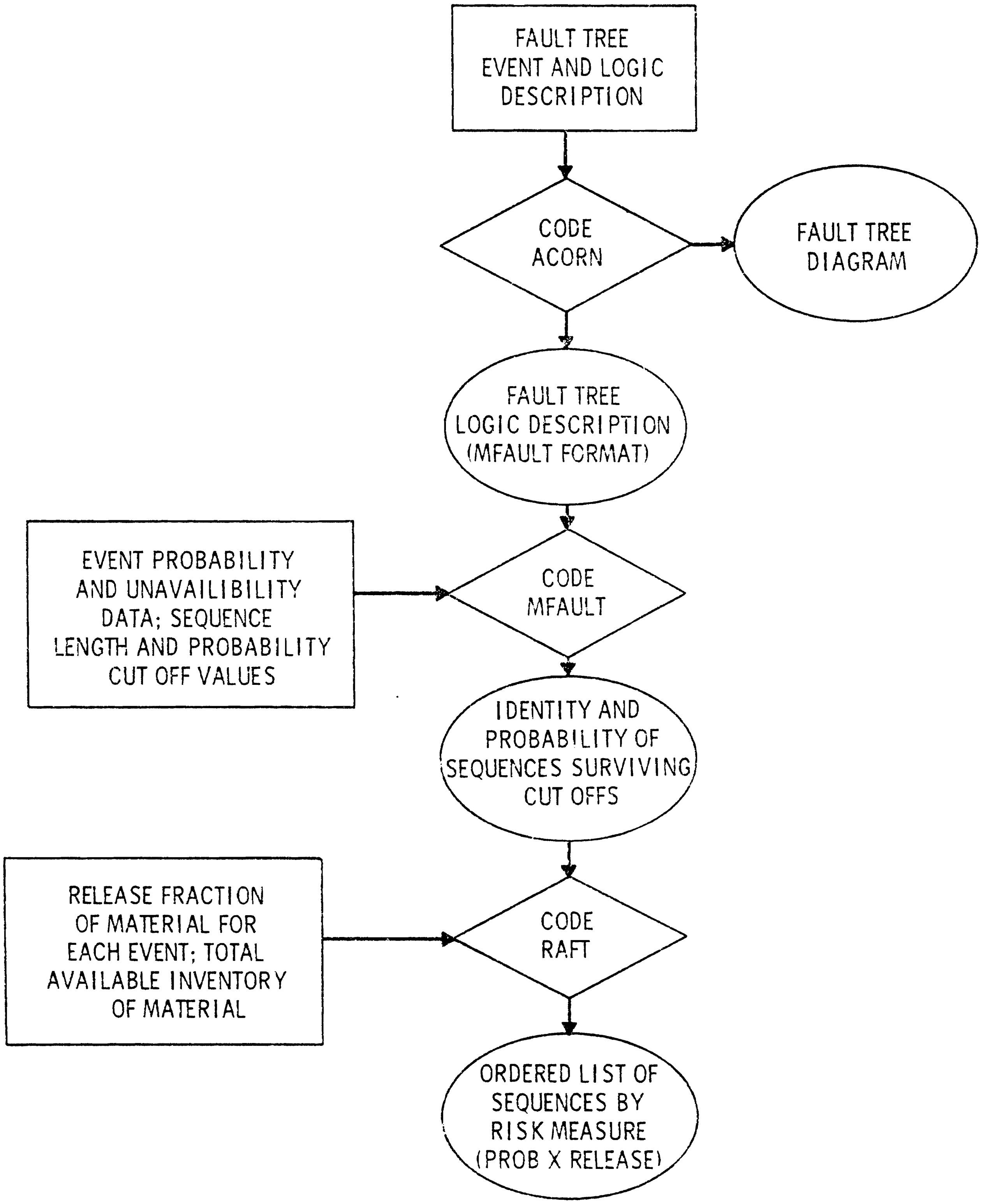
FAULT TREE EVENT AND LOGIC DESCRIPTION
CODE ACORN
FAULT TREE DIAGRAM
FAULT TREE LOGIC DESCRIPTION (MFAULT FORMAT)
EVENT PROBABILITY AND UNAVAILIBILITY DATA; SEQUENCE LENGTH AND PROBABILITY CUT OFF VALUES
CODE MFAULT
IDENTITY AND PROBABILITY OF SEQUENCES SURVIVING CUT OFFS
RELEASE FRACTION OF MATERIAL FOR EACH EVENT; TOTAL AVAILABLE INVENTORY OF MATERIAL
CODE RAFT
ORDERED LIST OF SEQUENCES BY RISK MEASURE (PROB X RELEASE)

FIGURE 1. Risk Assessment Computer Package

2.0 INFORMATION FOR THE USER

2.1 GENERAL DESCRIPTION OF THE PROGRAM

The RAFT computer code calculates release quantities and a risk measure based on the product of probability and release quantity for cut sets of fault trees modeling the accidental release of radioactive material from a nuclear fuel cycle facility.

Cut sets and their probabilities are supplied as input to RAFT from an external fault tree analysis code (e.g., MFAULT[3]). Using the total inventory available of radioactive material, along with release fractions for each event in a cut set, the release terms are calculated for each cut set. Each release term is multiplied by the cut set probability yielding the cut set risk measure. RAFT orders the dominant cut sets on the risk measure. The total risk measure of processed cut sets and their fractional contributions are supplied as output.

Input options are available to eliminate redundant cut sets, apply threshold values on cut set probability and risk, and control the total number of cut sets output. Hash addressing is used to remove redundant cut sets from the analysis.

External cut set files, such as those supplied by MFAULT,[3] will normally contain the cut set probability. However, RAFT possesses the option of recomputing these probabilities, thereby allowing the user to investigate cut set dominance dependent upon selective choices of input quantities. The cut set probabilities are computed exactly as shown in References 1 and 3. The user should be familiar with terms such as cut set availability and/or unavailability, repairable and/or nonrepairable components, and conditional probabilities, all of which are explained in the same references.

Release fractions are also part of the input used to obtain a risk measure. The code will allow up to four distributed release fractions and their associated conditional probabilities per event. The risk measures derived from each sub-cut set (addition terms resulting from the distributed release fractions) are ordered with as many as 37 optionally retained for output.

Depending on the type of program generating the list of cut sets supplied to RAFT and the user options selected, the potential exists for duplicate cut sets. RAFT uses a hash addressing scheme to check for cut set repetition. The scheme is optional and is capable of handling large groups of cut sets before computational restrictions become severe. Treatment of the hashing scheme is explained in Section 3.2.

Two threshold values can be used by RAFT to determine the relative significance of each cut set. The first is a probability cutoff. The second is a risk threshold for sub-cut sets in order to minimize the printed output. A cut set print limit is also available to minimize the total number of cut sets for the output file.

Two quantities contributing to the total risk-measure sum are calculated. One sum is related to the cut sets (and sub-cut sets) which have a greater risk measure than the threshold value. The other, the "branch residual sum", represents the risk measure of cut sets (and sub-cut sets) eliminated by the risk threshold. The "branch residual sum" is valuable in assessing the effectiveness of the risk threshold.

RAFT utilizes two word addressable files. One is for the hash addressing scheme to guarantee cut set uniqueness, while the other is used for maintaining output information from cut sets whose risk measures are most dominant.

During the process of ranking individual sub-cut sets within a cut set, the output file is updated by ranking cut sets on their accumulative risk measure. The file is maintained by simultaneously assigning index keys while ranking cut sets on risk measure. The index keys are subsequently used to retrieve the dominant cut sets. Only those cut sets surviving the probability cutoff will have been evaluated for risk measure, ranked by its dominance, and allocated for storage retrieval. The entire process eliminates the necessity to perform sorting methods which can become cumbersome when applied to large problems.

An abbreviated flowchart of the main program and a more detailed chart of the subroutine (RECORD) which manages the calculational procedures are included in Figures 2 and 3 in Section 4, in order to exhibit an overview of the operations performed in RAFT.

2.2 INPUT INSTRUCTIONS FOR RAFT

Input to RAFT is composed of two primary sources. The largest is normally an external file containing the cut set sequences and their probabilities. The file can reside on magnetic tape or disk, however, small problems can be supplied via input cards. The second source of input is supplied by a specially written version of NAMELIST.[a] This version of NAMELIST is described in Appendix A.

A third source of input is also available which allows for identification of cut set events on the output file. Made up of cards from the ACORN[2] code, these descriptions are optional and are used to help clarify the output.

If a disk or tape device is used to maintain the cut set file, certain storage properties must be observed to make the file legible to RAFT. Tape 7 is the local file name for the input cut sets. The cut set file must be partitioned. This means that cut sets with a like number of basic events must exist in a contiguous manner. Second, the code has been arranged so that groups of cut sets will be read from the storage media in blocks of one hundred (100) cut sets or until an end-of-file mark is encountered, terminating the file. When hashing options are activated, partition boundaries become important, in that restart problems must begin on one of these boundaries, otherwise uniqueness cannot be guaranteed.

An additional set of input data cards may be used to describe the events on the output. The event descriptors contain the event index for correlation with the basic events. Therefore, they may be loaded in any order. Nonexistent descriptors will be identified as such on the output and need not be of concern to the user. Normally they are the same cards as used by the ACORN[2] code, and as such, have been treated with no change in format and external to the NAMELIST input data.

NAMELIST control parameters, and input quantities relevant to various options are defined in the following pages. Default quantities are given where appropriate.

[a] NAMELIST is normally a system supplied software package, but is used throughout this document in reference to the one supplied in the appendices. However, a basic knowledge of the use of NAMELIST is assumed because this package adheres to the FORTRAN rules for NAMELIST.

NAMELIST Parameter	Implicit Type	Descriptions, Limits, Defaults
$RAFT	None	NAMELIST group name
TIA=	Real	Total inventory available of radioactive material Preset TIA=1.0
CUTOFA=	Real	Cut set probability threshold Preset CUTOFA=0
CUTOFF=	Real	Risk measure threshold Preset CUTOFF=0
RF(1,1)=	Real	RF(I,J), Release fractions I=1 to 4 release fractions J= index used to identify the event Preset ((RF(I,J),I=1,4),J=1,500)=0 at initialization. Not preset for subsequent multiple case execution.
CP(1,1)=	Real	CP(I,J), Conditional probabilities I=1 to 4 probabilities in one-to-one correspondence with RF(I,J). J= index used to identify the event Preset ((CP(I,J),I=1,4),J=1,500)=0 at initialization. Not preset for subsequent multiple case execution.
MODIFY=	Alpha-numeric	Decision flag to allow recomputation of cut set probabilities. ='YES', cut set probabilities are computed as a function of PQ and MODE. ='NO', cut set probabilities supplied as input are not altered. Preset MODIFY='NO'
PQ(1,1)=	Real	PQ(I,J), Probability/Unavailability I=1; event component probability I=2; event component unavailability J= index used to identify the event Required input when MODIFY='YES' Not preset

MODE(1)=	Integer	MODE(J), Repairable or nonrepairable component flag =0; event component is repairable =1; event component is nonrepairable Required input when MODIFY='YES' Not preset
HSHING=	Alpha-numeric	Decision flag to initiate hashing algorithms in order to determine cut set uniqueness. ='YES'; hashing routines are activated and uniqueness of cut sets is determined. ='NO'; no uniqueness tests are performed on the cut sets, cut set sequences are not ordered, repeated event structure is not checked, and the code is not input limited with respect to cut set volume. Preset HSHING='YES'
DEBUG=	Alpha-numeric	Decision flag for printing hashed cut sets. This option is only useful when the hashing algorithms fail to exhibit a well balanced load factor, α, within the hash table. The option is primarily programmer oriented and the user need not normally be concerned about its use. ='YES'; table of hashed cut sets is printed for investigation. ='NO'; no table is printed. Preset DEBUG='NO'
CUTSET=	Integer	Defines the number of the cut set with which the execution will start. This parameter is useful when large fault trees are partitioned for multiple case executions and restart capability. Preset CUTSET=1
TREES=	Real	Restart parameter available for multiple case executions of the same fault tree. For example, if a

fault tree is executed necessitating multiple runs, the total tree sum (including the branch residual sum) can be input into TREES for the subsequent execution. This can be important if the analyst is interested in correct accumulative cut set fractions relative to the entire tree.
Preset TREES=0.

LUDISK=	Integer	Logical unit which defines the mass storage device to be used for ranking release fractions. Preset LUDISK=1
MEDIA=	Alpha-numeric	Identifies the input media used to supply cut sets. ='DISK'; cut sets can be read from either disk or magnetic tape, assuming nonformatted binary records. EVENTS, PROBAB, and INDICE(N) [N=1, EVENTS] are obtained where EVENTS= number of events contained in the cut set PROBAB= cut set probability which can be recomputed as stated earlier by the use of MODIFY='YES' INDICE(N)= numeric index identifying each event within the cut set ='CARDS'; the same quantities are obtained through a formatted read (I2,E10.0,10I5). Preset MEDIA='DISK'
TITLE(1)=	Alpha-numeric	(TITLE(I),I=1,8) Problem description title. Preset TITLE=8*'bbbbbbbbb'
GATE=	Alpha-numeric	Flag to indicate presence of gate name on cut set input file. The gate name, if present, is not used by RAFT. Its existence is used solely for identification of the data file. ='YES'; then RAFT will read the gate name. (MEDIA= 'DISK' only) ='NØ'; RAFT will not expect a gate name. Preset GATE='YES'

READ=	Integer	Decision flag used to indicate that RAFT should expect hollerith definitions for each basic event. The descriptors are used only for the output printer and then only when PRINT='YES'. ='YES'; ACORN(2) description cards are expected immediately following the NAMELIST data. Columns 1 and 2 are ignored, columns 3, 4, and 5 define the event subscript which is used to correlate the event with its description, and the remaining part of each card describes the event. An end-of-file card is expected as a terminator for the ACORN cards. ='NO'; no descriptor cards are expected. and no descriptions are supplied to the output. Preset READ='NO'
LIMIT=	Integer	Maximum Number of dominant sub-cut sets allowed from each cut set in terms of risk measure for the line printer. $(1 \leq \text{LIMIT} \leq 37)$ Preset LIMIT=25
MAXWRT=	Integer	Maximum number of dominant (largest risk measure) cut sets allowed for the print file. $(1 \leq \text{MAXWRT} \leq 2000)$ Preset MAXWRT=2000
PRINT=	Alpha-numeric	Option flag for specifying ACORN descriptors to be printed. ='YES'; all events will be defined on output ='NO'; no events will be defined on output Preset PRINT='NO'
FIELD=	Integer	Active column field definition. Identifies last column of NAMELIST input cards which is to remain active. Preset FIELD=80
$		Case terminator

The format of the input cut set file (on tape or disk or cards) is given under the NAMELIST parameter MEDIA discussed above. The maximum number of events per cut set is currently 10. The events in a cut set are identified by an integer value presently not to exceed 500. The sample case given in Section 2.4 illustrates the input to RAFT for the case where the cut sets are input by tape or disk. The structure of the RAFT input with the cut sets supplied by cards is given below. The event description cards are also shown in the input stream.

```
$ RAFT     MEDIA='CARDS'     READ='YES'

  ⋮ other NAMELIST parameters

$
```

Event description cards. One card for each event. (2X, I3, 7A10, A5)

column	
1-2	not used
3-5	event number
6-80	Hollerith description of the event

END-OF-RECORD CARD to terminate event description cards.

Cut set identification cards. One card per cut set. Cut sets must be partitioned (i.e., all one term, then all two term, etc.)
(I2, E10.0, 10I5)

column	
1-2	number of events per cut set
3-12	cut set probability
13-17	event index, 1st event
18-22	event index, 2nd event
⋮	⋮
58-62	event index, 10th event

END-OF-RECORD CARD to terminate cut set identification cards.

2.3 DESCRIPTION OF OUTPUT

Cut sets are numbered and listed according to descending values of risk measure. Each cut set printed contains the probability and the cumulative risk measure fraction of the total tree. In addition, the numerical indices of the events within each cut set are printed along with verbal descriptors of those events and the release fractions associated with each of the dominant sub-cut sets. Further, the accumulated product of the risk measures and the sum of the cut set probabilities are given for the tree. These quantities represent contributions from all cut sets surviving the probability threshold.

RAFT places no limit on the number of cut sets from a tree. However, if the hashing scheme is used for redundancy checking, the output file could become restrictive in core size, causing the remaining information to be stored on scratch disk. When this occurs, the code's efficiency is severely reduced, causing excessive execution times.

2.4 SAMPLE CASE

A sample case is given in this section to illustrate the input information RAFT requires and the resulting output. The list of cut sets used for this sample case was obtained from processing the fault tree given in Reference 3 by the MFAULT computer program.

```
RAFT,T100,CM140000.                                   SEYBOLD               3000 AREA
ACOUNT CARD
FTN(OPT=2,L=0,B=RAFT)
ATTACH(TAPE7,RAFTS,ID=PELTO)
REQUEST(TAPE1,*PF,EC)
REQUEST(TAPE11,*PF)
RFL(140000)
RAFT.
'
C
C        PROGRAM RAFT SOURCE DECK GOES HERE...
C
'
 $RAFT TIA=1.0 CUTOFA=0. CUTOFF=0. FIELD=80 MEDIA='DISK' HSHING='YES'
 MAXWRT=50 LIMIT=10
 TITLE='EXAMPLE FAULT TREE GIVEN IN BNWL-2145'
 RF(1,1)=.1 CP(1,1)=1.
 RF(1,2)=1. CP(1,2)=1.
 RF(1,3)=1. CP(1,3)=1.
 RF(1,4)=.1 CP(1,4)=1.
 RF(1,5)=1. CP(1,5)=1.
 RF(1,6)=1. CP(1,6)=1.
 RF(1,7)=.1 CP(1,7)=1.
  RF(1,8)=1. CP(1,8)=1.
 RF(1,9)=1. CP(1,9)=1.
 RF(1,10)=.1 CP(1,10)=1.
 RF(1,11)=.01 .1 CP(1,11)=.9 .1
 RF(1,12)=1. CP(1,12)=1.
 RF(1,13)=1. CP(1,13)=1.
 RF(1,14)=.1 CP(1,14)=1.
 RF(1,15)=1. CP(1,15)=1.
 RF(1,16)=1. CP(1,16)=1.
 RF(1,17)=.1 CP(1,17)=1.
 RF(1,18)=.1 CP(1,18)=1.
 RF(1,19)=1. CP(1,19)=1.
 RF(1,20)=1. CP(1,20)=1.
 RF(1,21)=1. CP(1,21)=1.
 RF(1,22)=1. CP(1,22)=1.
 RF(1,23)=1. CP(1,23)=1.
 RF(1,24)=1. CP(1,24)=1.
 RF(1,25)=.1 .5 CP(1,25)=.9 .1
 RF(1,26)=1. CP(1,26)=1.
 RF(1,27)=1. CP(1,27)=1.
 RF(1,28)=.1 CP(1,28)=1.
 RF(1,29)=1. CP(1,29)=1.
 RF(1,30)=1. CP(1,30)=1.
 RF(1,31)=1. CP(1,31)=1.
 RF(1,32)=1. CP(1,32)=1. $
'
```

```
EVENT   RF(1,EVENT)     CP(1,EVENT)     RF(2,EVENT)     CP(2,EVENT)     RF(3,EVENT)     CP(3,EVENT)     RF(4,EVENT)     CP(4,EVENT)
    1   1.000000E-01    1.000000E+00
    2   1.000000E+00    1.000000E+00
    3   1.000000E+00    1.000000E+00
    4   1.000000E-01    1.000000E+00
    5   1.000000E+00    1.000000E+00
    6   1.000000E+00    1.000000E+00
    7   1.000000E-01    1.000000E+00
    8   1.000000E+00    1.000000E+00
    9   1.000000E+00    1.000000E+00
   10   1.000000E-01    1.000000E+00
   11   1.000000E-02    9.000000E-01    1.000000E-01    1.000000E-01
   12   1.000000E+00    1.000000E+00
   13   1.000000E+00    1.000000E+00
   14   1.000000E-01    1.000000E+00
   15   1.000000E+00    1.000000E+00
   16   1.000000E+00    1.000000E+00
   17   1.000000E-01    1.000000E+00
   18   1.000000E-01    1.000000E+00
   19   1.000000E+00    1.000000E+00
   20   1.000000E+00    1.000000E+00
   21   1.000000E+00    1.000000E+00
   22   1.000000E+00    1.000000E+00
   23   1.000000E+00    1.000000E+00
   24   1.000000E+00    1.000000E+00
   25   1.000000E-01    9.000000E-01    5.000000E-01    1.000000E-01
   26   1.000000E+00    1.000000E+00
   27   1.000000E+00    1.000000E+00
   28   1.000000E-01    1.000000E+00
   29   1.000000E+00    1.000000E+00
   30   1.000000E+00    1.000000E+00
   31   1.000000E+00    1.000000E+00
   32   1.000000E+00    1.000000E+00
```

```
EXAMPLE FAULT TREE GIVEN IN BNWL-2145                                                                    PAGE    1

                                     B N W   P R O G R A M   R A F T
                                     RISK ASSESSMENT FOR FAULT TREES

                                              I N P U T   D A T A
                                              ---------   -------

TOTAL INVENTORY       PROBABILITY          RISK            FIELD INPUT         INITIAL
AVAILABLE             THRESHOLD          THRESHOLD         DEFINITION          RISK SUM
---------------       -----------        ---------         ----------          --------
     TIA                CUTOFA            CUTOFF             FIELD              TREES
 1.000000E+00          0.                0.                   80              0.

 CUTSET MEDIA         DISC DEVICE        HASH TABLE        PROBABILITY         INITIAL
                      FOR RANKING        LIST OPTION       CALCULATION       CUTSET NUMBER
 ------------         -----------        -----------       -----------       -------------
 MEDIA= DISK          LUDISK=   1        DEBUG=  NO        MODIFY= NO        CUTSET=     1

 PRINT LIMIT            HASHING
  PER CUTSET           INITIATED
 -----------           -----------
      10                  YES
```

```
CUTSET PROBABILITY  =    4.600000E-04   ACCUMMULATIVE RISK FRACTION      =   9.297413E-01              CUTSET RANK=    1
UNAVAILABILITY TERM =    0.             TOTAL RISK FOR FAILURE SEQUENCE=   4.600000E-04
CUTSET FAILURE SEQUENCE      =      29         30

LARGEST  1    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 4.600E-04 1.000E+00 4.600E-04 1.000E+00 1.000E+00

CUTSET PROBABILITY  =    3.000000E-04   ACCUMMULATIVE RISK FRACTION      =   9.903766E-01              CUTSET RANK=    2
UNAVAILABILITY TERM =    0.             TOTAL RISK FOR FAILURE SEQUENCE=   3.000000E-05
CUTSET FAILURE SEQUENCE      =      10

LARGEST  1    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 3.000E-05 1.000E-01 3.000E-04 1.000E-01

CUTSET PROBABILITY  =    3.000000E-05   ACCUMMULATIVE RISK FRACTION      =   9.964401E-01              CUTSET RANK=    3
UNAVAILABILITY TERM =    0.             TOTAL RISK FOR FAILURE SEQUENCE=   3.000000E-06
CUTSET FAILURE SEQUENCE      =       1          2          3

LARGEST  1    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 3.000E-06 1.000E-01 3.000E-05 1.000E-01 1.000E+00 1.000E+00

CUTSET PROBABILITY  =    8.539600E-06   ACCUMMULATIVE RISK FRACTION      =   9.981661E-01              CUTSET RANK=    4
UNAVAILABILITY TERM =    0.             TOTAL RISK FOR FAILURE SEQUENCE=   8.539600E-07
CUTSET FAILURE SEQUENCE      =       4          5          6

LARGEST  1    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 8.540E-07 1.000E-01 8.540E-06 1.000E-01 1.000E+00 1.000E+00

CUTSET PROBABILITY  =    2.070000E-04   ACCUMMULATIVE RISK FRACTION      =   9.992790E-01              CUTSET RANK=    5
UNAVAILABILITY TERM =    0.             TOTAL RISK FOR FAILURE SEQUENCE=   5.506200E-07
CUTSET FAILURE SEQUENCE      =      11         25

LARGEST  4    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 1.863E-07 1.000E-02 1.863E-05 1.000E-01 1.000E-01
 1.677E-07 1.000E-03 1.677E-04 1.000E-02 1.000E-01
 1.035E-07 5.000E-02 2.070E-06 1.000E-01 5.000E-01
 9.315E-08 5.000E-03 1.863E-05 1.000E-02 5.000E-01

CUTSET PROBABILITY  =    2.613333E-06   ACCUMMULATIVE RISK FRACTION      =   9.998072E-01              CUTSET RANK=    6
UNAVAILABILITY TERM =    0.             TOTAL RISK FOR FAILURE SEQUENCE=   2.613333E-07
CUTSET FAILURE SEQUENCE      =       7          8          9

LARGEST  1    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 2.613E-07 1.000E-01 2.613E-06 1.000E-01 1.000E+00 1.000E+00
```

```
                    EXAMPLE FAULT TREE GIVEN IN BNWL-2145                                                      PAGE    5

CUTSET PROBABILITY  =   3.078000E-06   ACCUMMULATIVE RISK FRACTION     =   9.999254E-01                     CUTSET RANK=    7
UNAVAILABILITY TERM =   0.             TOTAL RISK FOR FAILURE SEQUENCE=   5.848200E-08
CUTSET FAILURE SEQUENCE     =     11        12        13

LARGEST  2    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 3.078E-08 1.000E-01 3.078E-07 1.000E-01 1.000E+00 1.000E+00
 2.770E-08 1.000E-02 2.770E-06 1.000E-02 1.000E+00 1.000E+00


CUTSET PROBABILITY  =   3.690000E-07   ACCUMMULATIVE RISK FRACTION     =   1.000000E+00                     CUTSET RANK=    8
UNAVAILABILITY TERM =   0.             TOTAL RISK FOR FAILURE SEQUENCE=   3.690000E-08
CUTSET FAILURE SEQUENCE     =     14        15        16

LARGEST  1    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 3.690E-08 1.000E-01 3.690E-07 1.000E-01 1.000E+00 1.000E+00


CUTSET PROBABILITY  =   5.838336E-18   ACCUMMULATIVE RISK FRACTION     =   1.000000E+00                     CUTSET RANK=    9
UNAVAILABILITY TERM =   0.             TOTAL RISK FOR FAILURE SEQUENCE=   5.838336E-21
CUTSET FAILURE SEQUENCE     =     17        18        19        20        21        22        23        24        27        28

LARGEST  1    TIA*     CUTSET P
  RISK(S)    RF(SEQ)   *CP(SEQ)  CONTRIBUTING RELEASE FRACTIONS FOR ABOVE FAILURE SEQUENCE
 5.838E-21 1.000E-03 5.838E-19 1.000E-01 1.000E-01 1.000E+00 1.000E+00 1.000E+00 1.000E+00 1.000E+00 1.000E+00 1.000E+00 1.000E-01
```

```
                         EXAMPLE FAULT TREE GIVEN IN BNWL-2145

TOTAL A*B PRODUCT FOR THE TREE=       4.947613E-04
TOTAL A*B PRODUCT BRANCH RESIDUAL=  0.

PROCESSING COMPLETE FOR              9 CUTSETS.
      0 CUTSETS WERE DISCARDED AS REDUNDANT.

SUM OF CUTSET PROBABILITIES =      1.011600E-03

SUM(PROBABILITIES)/(NUMBER OF UNIQUE CUTSETS)=      1.124000E-04
```

3.0 PROCEDURE FOR SOLUTION

3.1 CALCULATION OF THE RISK MEASURE:

A cut set risk can be defined as the product of five terms:

RISK= A*B*C*D*E[1] (1)

where

A= probability of the release sequence;

B= release quantity;

C= relevant measures of the radioactive material characteristics (isotopic composition, size distribution, solubility, volatility, etc.);

D= a measure of environmental transport path efficiency;

E= a measure of population distribution, characteristics, and habits.

The risk measure, R, used in the RAFT code represents a simplification of Equation (1). Using only the probability and the release quantity, the risk measure is defined as:

R= A*B (2)

Reference 1 gives a detailed discussion of the use of this simplified risk expression.

The fault tree cut sets, such as X_1, X_3X_5, $X_9X_{11}X_{42}$, etc., define the chain of events with which the risk measure is evaluated. Each X_j represents a basic event, an inhibit condition, or an on/off switch.

To calculate the risk measure of a cut set, the RAFT code uses the release fractions and their associated conditional probabilities of each basic event. The release fraction term (RF) is defined as the amount of radioactive material available at the output or completion of the event divided by the amount available at the input or beginning of the event. Some events do not have a physically associated release fraction, while others can be assigned a single-valued release fraction whose value is less than or equal to 1.0. Still others might be assigned distributed release fractions which must also contain a corresponding set of conditional probabilities, CP_{ji}. In these cases, the condition $\sum_{i=1}^{L} CP_{ji} = 1.0$ must be maintained. The value $L \leq 4$

represents a code limitation, while the subscript j indicates the event for which the RF_{ji} and CP_{ji} values are applied.

The other input quantity for RAFT is the total inventory available (TIA) of radioactive material. Typically a fault tree models one potential source of radioactive material so only one number is required. If several potential sources are modeled, the TIA value may differ for different cut sets. This situation is treated by modifying the release fracitons for the initial containment barriers and assigning the largest potential source as the input TIA to RAFT.

The procedure for calculating the cut set risk measure, A X B, (A is probability and B is release quantity) is given in the following examples taken from Reference 1. A cut set consisting of basic events X_1, X_2, ...X_n and having probability A is assumed.

Case 1 There are no distributed release fractions for events in the cut set. The release fractions are RF_1 for X_1, RF_2 for X_2, ..., RF_n for X_n.

$B = (TIA)\ RF_1 RF_2 \ldots RF_n$

$A \times B = A\ (TIA)\ RF_1 RF_2 \ldots RF_n$

For each cut set of this type, only one additive A x B term arises.

Case 2 Basic event x_j has a distributed release fraction. All remaining basic events in the cut set have single release fractions as before.

If the release fraction distribution for X_j has L pairs of (RF_{ji}, CP_{ji}) values, then A x B for the cut set consists of L additive terms (sub-cut sets):

Term 1 yields $(A)(CP_{j1})(TIA)RF_1 RF_2 \ldots RF_{j_1} \ldots RF_n$

Term 2 yields $(A)(CP_{j2})(TIA)RF_1 RF_2 \ldots RF_{j_2} \ldots RF_n$

:

Term L yields $(A)(CP_{jL})(TIA)RF_1 RF_2 \ldots RF_{j_L} \ldots RF_n$.

These L contributors constitute the total A x B product for the cut set. Code RAFT prints a record of (up to 37 per cut set) the contributors of highest magnitude, including the values of probability and release quantity as calculated by the above equations. This information indicates which part of the release fraction distribution contributes most to the calculated risk measure. Further analysis or testing in that region may be advisable.

Case 3 Several basic events in the cut set have distributed release fractions. This is the most general case.

Assume m basic events have distributed release fractions, with one distribution having L_1 values, another having L_2 values, and the m^{th} such basic event having L_m values. Then the A x B product for this cut set consists of $L_1L_2 \ldots L_m$ additive contributions (sub-cut sets). The technique in Case 2 is readily extended to accommodate Case 3. The technique includes treatment of the cross-product terms resulting from multiple distributions.

3.2 REDUNDANCY CHECKING

Cut set files obtained from fault tree codes may contain repeated cut set sequences. The impact on computer time and storage of processing large cut set files to assure uniqueness could be prohibitive. Therefore, linear programming techniques were replaced by hash addressing (scatter storage) in order to alleviate excessive running times.

The hash addressing scheme employed is actually a modification of hashing addresses, in that overflow tables are used. This eliminates the need to expand or contract the size of the hash table. In addition, the method demands that the input file be partitioned on boundaries separating cut set groups by their event sequence length. This technique automatically implies uniqueness across partitions and allows the hash table and overflow tables to be re-initialized at partition boundaries. The overall reduction in computer time is realized by reducing the load factor (collision rate) of the hashing tables.

Since the intent is to guarantee uniqueness of cut sets for subsequent processing, the hash address was defined simply as:

$$\text{HASH} = \sum_{N=1}^{\text{EVENTS}} \text{INDICE}_N$$

where EVENTS represents the number of basic events in the cut set, and INDICE identifies each event as a uniquely defined integer.

The entire scheme remains as an optional portion of the code. For extremely large files or for cases where the effect of redundant cut sets may be insignificant, the user may find it advantageous to omit the hashing option.

A flowchart illustrating the structure of the hash table and accompanying overflow table is shown in Figure 3 in Section 4.0. The routine, WEED, utilizes some of the features of Record Manager[a] in order to facilitate the maneuvering of cut set records on mass storage devices.

[a] Record Manager is a series of CDC supplied software routines which are used for storage and retrieval of data records on mass storage devices.

4.0 ADDITIONAL PROGRAMMING INFORMATION

4.1 GENERAL INFORMATION

Except for the exclusive use of Record Manager, the RAFT code is written entirely in FORTRAN language. It has been successfully used at PNL for the past 2 years without any apparent problems. It was compiled and executed under the SCOPE 3.4.2 operating system. The package is made up of 1493 source cards of which NAMELIST constitutes 631.

The computational body of the code is made up of 18 program modules, while the NAMELIST loading package includes an additional 16 subroutines. They are listed here as two independent groups.

Group 1/RAFT

RAFT	RECORD	PRINT	UNPACK	DSKCRD
DATA	SPEED	ALGOR	REPAIR	STACK
LOAD	WEED	PAGE	DISC	
PUTIN	ACCEPT	TABLES	PUTOUT	

Group 2/NAMELIST

NEWTAB	SCAN	SIGN2	SIGN8
NAMTAB	LOCATR	SIGN34	MOVEAB
READ	LOADER	SIGN5	CHECK
LOCATE	ERRORS	SIGN7	CHAIN

A cross-reference map (Appendix B) is included which relates the coupling of each program variable, subroutine, and system library function to each of its associated counterparts. The table is paged and alphabetized to allow immediate reference.

4.2 SUBROUTINE DESCRIPTIONS

PROGRAM RAFT is the main program. Its principal function is to establish the Record Manager files and control the primary flow for processing cases. Some default values for maintaining the overhead logic are also defined. A simplified flowchart is given in Figure 2.

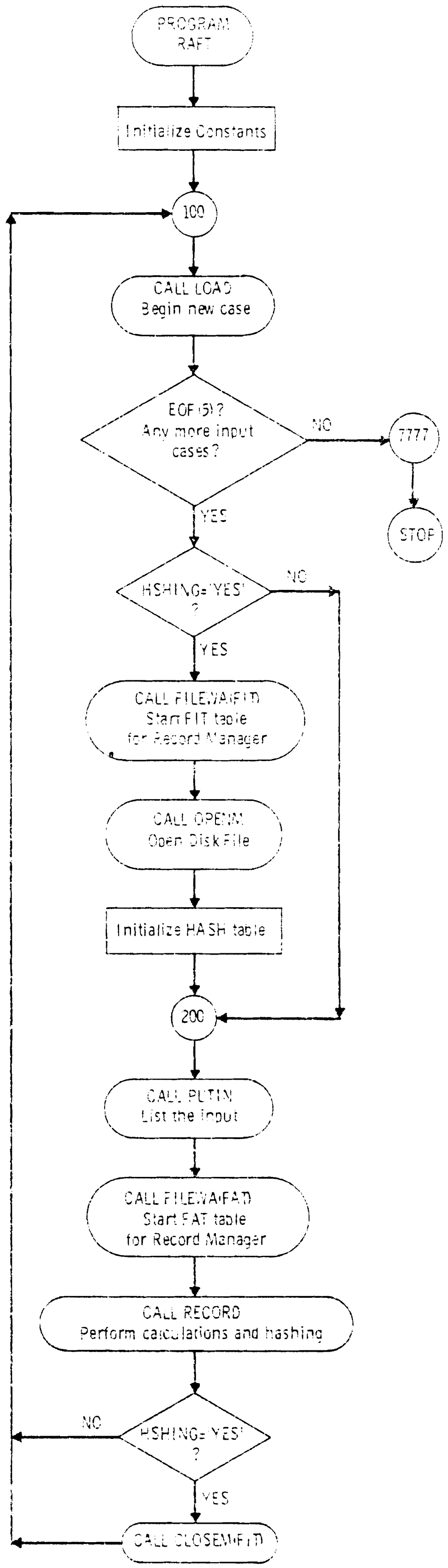
PROGRAM RAFT
Initialize Constants
100
CALL LOAD Begin new case
EOF(5)? Any more input cases?
NO
7777
STOP
YES
HSHING='YES' ?
NO
YES
CALL FILEWA(FIT) Start FIT table for Record Manager
CALL OPENM Open Disk File
Initialize HASH table
200
CALL PUTIN List the input
CALL FILEWA(FAT) Start FAT table for Record Manager
CALL RECORD Perform calculations and hashing
NO
HSHING='YES' ?
YES
CALL CLOSEM(FIT)

FIGURE 2. Flow Chart for RAFT

BLOCK DATA is used strictly to default the input parameters. The defaulted values are listed in the Input Instructions, where appropriate.

LOAD's primary function is to allow loading of any input data, except cut set sequences, which are to be defined by cards. The NAMELIST table is established, input parameters defining limits or dimensions are protected, and restart capabilities are initiated.

The PUTIN routine is provided in order to list the input parameters on the output file.

RECORD controls the calculational procedures for the fault tree data. All subroutines which are used to compute the sub-cut set risks, total cut set risks, total tree risk, ranking, and output file managing are controlled by RECORD. In addition, all cut set event sequences, whether input by cards or mass storage, are loaded within this subroutine. Furthermore, the cut set probability screening is performed in RECORD in order to eliminate cut sets which are destined to have very low risk measure contributions. A flowchart is given in Figure 3.

A word of caution is in line here with respect to the cut set input sequences. Simply stated, any sequence possessing an event subscript which is outside the range 1 to 500, inclusive, will be eliminated from the processing with no diagnostic messages supplied to the user. The resulting implication is that cut sets generated by codes other than MFAULT(3) must guarantee this property in order to use RAFT successfully in any meaningful sense.

Subroutine SPEED is a special purpose routine designed specifically for computer speed in order to compute all the existing permutations of risk measure derived from a cut set sequence. Both the "total tree sum" and the "branch residual sum" are accumulated in this routine depending upon their surviving the risk measure threshold.

ACCEPT is used to provide some preliminary screening of each cut set. However, the screening here does not involve elimination of the cut set; instead, it is intended to determine the dominant sub-cut set risk measure so

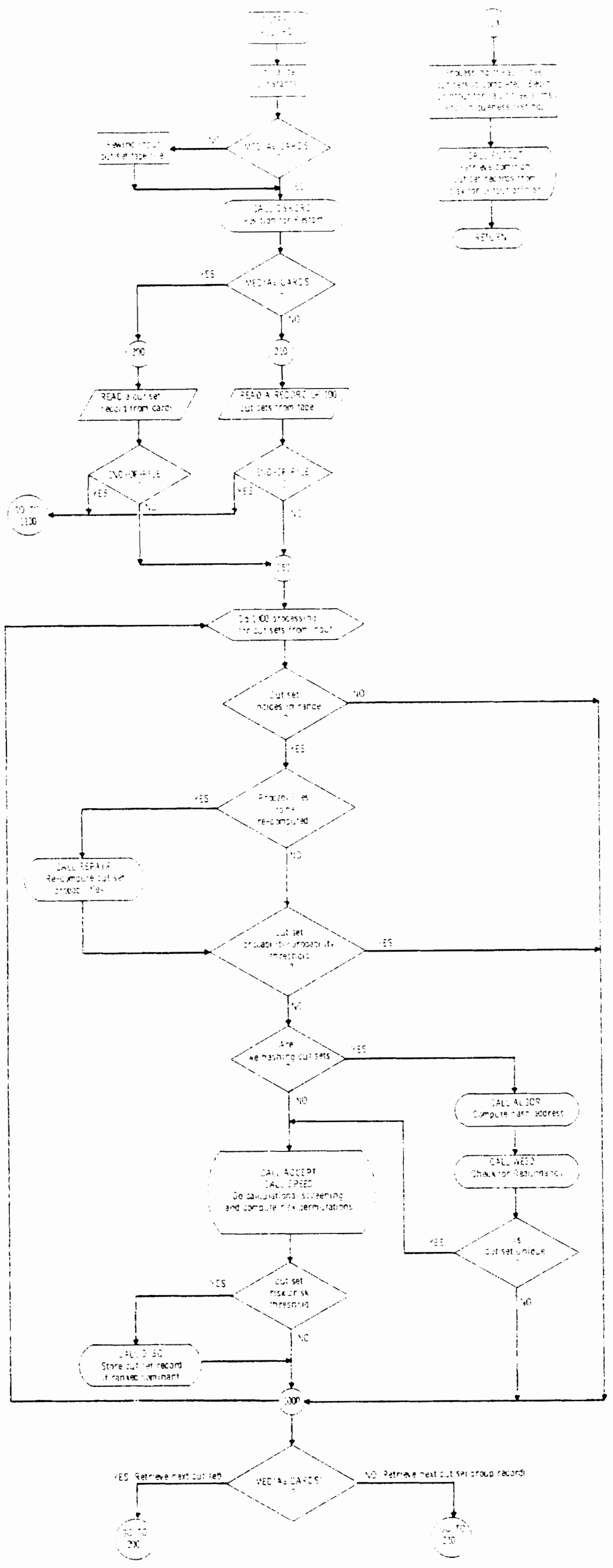
Rewind input cut set tape file
MEDIA = CARDS ?
NO
YES
CALL DSKORD Position for restart
MEDIA = CARDS ?
200
210
READ a cut set record from cards
READ A RECORD OF 100 cut sets from tape
END-OF-FILE ?
GO TO 1100
150
Do 1100 processing for cut sets from input
Cut set indices in range ?
Probabilities to be re-computed
CALL REPAIR Re-compute cut set probabilities
Cut set probability < probability threshold ?
Are we hashing cut sets ?
CALL ACCEPT
CALL SPEED
Do calculational screening and compute risk permutations
CALL ADDR Compute hash address
CALL WEED Check for Redundancy
Is cut set unique ?
Cut set risk < risk threshold
CALL DISO Store cut set record if ranked dominant
1000
YES Retrieve next cut set
NO Retrieve next cut set group record
MEDIA = CARDS ?
GO TO 200
GO TO 210
RETURN

FIGURE 3. Flow Chart for RECORD

that the cut set can be flagged for possible inclusion in the "total tree sum" or within the "branch residual sum". The underlying purpose behind this scheme is to minimize the number of calls to subroutine STACK. In other words, limit the cut set ranking to only those which contain sub-cut sets above the risk measure threshold. This masking scheme was proven to be beneficial for both small and large fault trees and particularly true when the risk measure threshold is set to eliminate the bulk of the surviving cut sets from the output file.

A risk analysis could be made without the masking provided by subroutine ACCEPT, but only at the expense of increased computer time.

Subroutine STACK has the task of retaining an ordered list of the sub-cut set release fractions obtained from each cut set. The routine utilizes a linear scheme for ordering the release fraction sequences with respect to descending risk measure. This type of programming was chosen because of the limited length of items to be ordered. Therefore, if the value of the input parameter, LIMIT, is substantially revised upward, another searching scheme, such as a binary search, should be employed.

DISC is called from subroutine RECORD in order to rank the surviving cut sets for the output file. Only cut sets which contain sub-cut set risk measures above the risk measure threshold are sent to this subroutine. In each instance the total risk measure of the cut set is used to rank all cut sets destined for output. The CYBER Record Manager is used to store the records on disc in word addressable form. Each record is assigned a KEY which is used to retain the storage location on disc. The KEY is kept in memory so as to alleviate the necessity of ordering the records.

Subroutine PUTOUT is initiated upon completion of the entire fault tree problem. Its function is to retrieve output records for printing by computing their location on disc, as a function of the KEY, and using Record Manager to access the record. Once accessed, subroutine PRINT is called to provide the printed record of permeated release fractions and their associated risk measures of the ranked cut sets. A utility routine, PAGE, is included for the explicit purpose of paging the output in a neat and presentable manner.

Subroutine DSKCRD has the singular purpose of allowing the code to read past cut set sequences from the input data file in order to position the data file for restart problems.

REPAIR is included to provide the user the option of recomputing or reassigning probabilities to the basic events before the cut set sequence is evaluated in terms of risk measure. The definition of these probabilities is contained in Reference 1. Input quantities associated with this option are defined in the descriptions of the input parameters MODIFY, MODE, and PQ.

ALGOR is the first subroutine called which is associated with the hashing scheme. The subroutine has four functions:

- Rearrange the cut set indices in ascending order.
- Eliminate cut sets which contain repeated indices.
- Compute the hash address as a function of the cut set indices.
- Pack cut set indices into three computer words so as to reduce storage requirements.

The subroutine is called for every cut set sequence when the hashing option, HSHING, is activated.

Subroutine WEED, Figure 4, performs the redundancy tests on the cut-set sequences and provides for their storage if they are determined unique. Three distinctly different tables are utilized for this purpose. First, the hash table is used for scatter storage. It is dimensioned so as not to allow storage into an illegal address from the hash addressing formula. Because scatter storage can be wasteful, a second table, the overflow table, was set up for in-core use. The overflow table is actually the primary recipient for storage of the cut sets when the load factor of the hash table has been reached. Its size can be easily modified to adjust for computer installation resources.

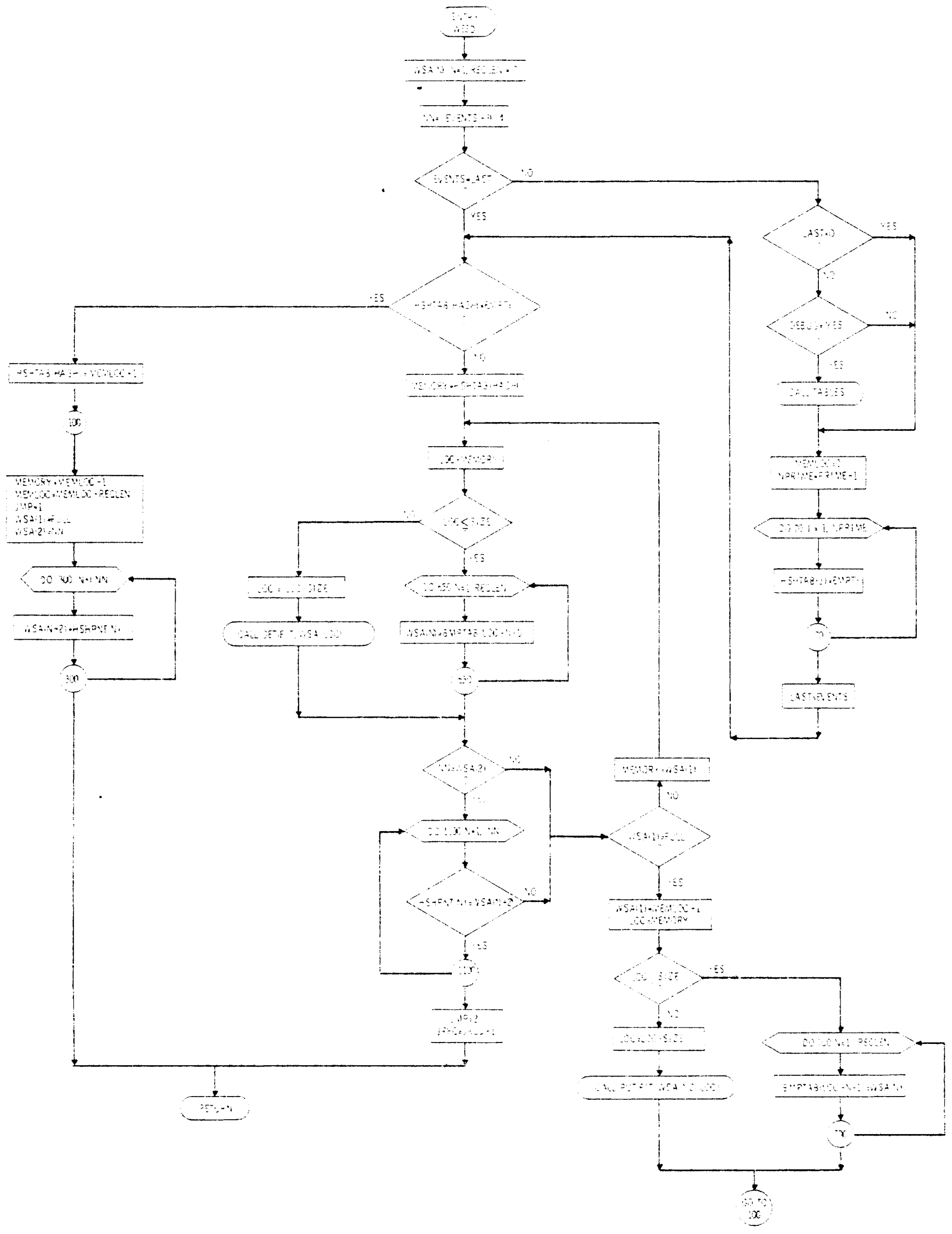

FIGURE 4. Flow Chart for WEED

In the event that the overflow table becomes full, a third table is established on disk. This table is maneuvered by the use of Record Manager word addressable files and can be as large as the computer system mass storage media allows. The obvious advantage of using disk is the tremendous increase in storage capacity. The only disadvantage is the penalty paid for access times involved. Problems associated with disk access timing were substantially reduced when the input data file was partitioned. This technique allowed the fault tree problem to be reduced by at least one level of magnitude, so far as storage requirements were concerned, because the hash table and the overflow table could then be reinitialized at the changing partitions.

The structure of the overflow table and the disc table is such that a linear search is never performed. Instead, each stored record contains a pointer to another location in the event that there is a collision and the stored record does not match the new record. Two benefits are derived from this technique:

- No storage locations in either table are wasted due to a scatter storage algorithm.
- Linear searching techniques are abolished and replaced by keyed addresses from record to record.

Another feature used in the storage scheme was that of word packing. The benefits are obvious and the computing time used to pack or unpack words is insignificant. The subroutine UNPACK is used by WEED to unpack the cut set sequences.

Subroutine TABLES is the last to be used by the hashing options available. In fact, it is used as a separate option to list the contents of the hash table. The routine is included so that possibly more efficient algorithms could be explored in the event that the one in RAFT is deemed inefficient for a large class of fault tree problems.

4.3 CALCULATIONAL DEPENDENCY

a) Number of events per cut set sequence, $EVENTS \leq 10$.

b) Number of events per fault tree, $INPUT \leq 500$.

c) Output line printer limit per cut set, $LIMIT \leq 37$.

d) Number of release fractions per event, $ITEMS \leq 4$.

e) Size of the hash table must be 5000 words.

f) The length of COMMON/FIXED/ must not deviate from 512 words.

g) The input data file must be partitioned when hashing options are employed.

h) The "DO LOOPS" in subroutine SPEED should always be equal to the maximum allowable number of events per cut set.

i) Input cut set data files are limited by mass storage resources when hashing. No limit when not hashing.

j) Input cut set data records are read in binary form in blocks of 100 cut set sequences when the input media is magnetic tape or disk.

k) The maximum integer word size for packed storage is computed as follows:

Packed word = $\{[INDICE(4)*512+INDICE(3)]*512+INDICE(2)\}*512+INDICE(1)$ $\approx 6.724*10^{10}$ where, INDICE represents the subscripts for events identifying the sequence.

l) The size of the overflow table, BMPTAB(SIZE), can be adjusted by the user. Two changes need to be made if a modification is required.

- The dimension, SIZE, of the variable BMPTAB needs to be updated.
- The variable SIZE in the main program must be equal to the dimension of BMPTAB.

Currently SIZE=40000 in the code, however, a nominal SIZE can be computed by the following algorithm:

SIZE$\simeq$5*N_{cs}, where N_{cs} = maximum number of expected unique cut sets from the largest partition.

4.4 MACHINE DEPENDENCY

a) Record Manager is used throughout for word addressable storage and retrieval from mass storage devices.

b) End-of-file routines are used to detect status conditions.

Remaining items concerning machine dependency are imbedded within the NAMELIST package. If they are too restrictive, the NAMELIST package could be easily removed and replaced by an installation supplied version of NAMELIST.

c) Program variables are limited to a maximum of three subscripts.

d) Dynamic dimensioning is used extensively.

e) ENCODE/DECODE are used throughout NAMELIST.

f) NAMELIST Parameter names can use up to seven characters.

g) Machine words are designed to admit 10 characters or 60 bits.

h) Input is limited to 80 character records.

i) Hollerith input must be delimited by apostrophe marks.

j) Variable data FORMATS are defined for CYBER systems.

k) The LOCF system routine is used to access relative locations of program variables.

l) Full word boundaries are assumed for all implicit data types.

4.5 EXTERNAL FILES

The RAFT code uses the following files when executed on the CYBER 74 computer:

Logical Unit	Use
5	Input data records
6	Output line printer
7	Magnetic tape or disk for maintaining cut set sequences for input to RAFT
1	Disk file used for ranking output cut sets
11	Disk file used for storing cut set records when hashing

4.6 CASE REQUIREMENTS

Blank COMMON consumes 51,118 words of memory.

Named COMMON consumes 10,183 words of memory.

Named COMMON within NAMELIST consumes 502 words of memory.

With current dimensions, the code requires 240K octal words to load. This includes the code, systems routines and functions, and a nominal 2,000 octal word internal buffer for each of the logical devices.

4.7 TIMING

Execution times are as varied as the problem and options specified. Some examples are as follows:

Case A) Assume the following dimensions and input conditions:

1. BMPTAB(SIZE), SIZE=12,500 words.
2. HSHING-'YES'; hashing scheme is activated.
3. The input data file contains 300 to 500 cut set sequences.
4. LIMIT=25; sub-cut set print limit.

5. Number of release fractions per event averages about two.

6. CUTOFA=0., and CUTOFF=0.; no existing threshold definition.

7. MODIFY='YES'; cut set probabilities are recomputed.

8. MAXWRT=2000; all unique cut sets will be disposed to the output file.

9. READ='YES', PRINT='YES'; event descriptions will be included.

These input conditions are representative of a small fault tree where the analyst might be interested in all the release fraction sequences. This is a typical problem for a sensitivity study in which the cut set probabilities are being varied to define a range, or envelope, of the projected consequences.

One case of this type would use approximately 30 seconds of computing time.

Case B) Assume the following dimensions and input conditions:

1. BMPTAB(SIZE), SIZE=40,000 words.

2. HSHING='YES'; hashing scheme is activated.

3. The input data file contains approximately 10,000 cut set sequences.

4. Limit=10; sub-cut set print limit.

5. Number of release fractions per event averages between 2 and 3.

6. CUTOFA=0., and CUTOFF>0 in order to minimize stacking.

7. MODIFY='NO'; cut set probabilities are considered sufficient.

8. MAXWRT=100; the output print file is limited to the most dominant cut sets.

9. READ='YES', PRINT='YES'; event descriptions will be included.

The above conditions are representative of a moderately sized fault tree with a sufficiently well defined set of probabilities. The overflow table dimension is set to allow for an efficient storage of cut set records and the output print file is limited to 100 of the most dominant cut sets. For well-constructed fault trees, MAXWRT=100 is usually sufficient in order to have selected cut sets contributing over 90% of the total risk measure.

A problem of this type can be executed within two to four minutes machine time depending heavily upon the collision rate within the hashing scheme.

REFERENCES

1. T. H. Smith, P. J. Pelto, D. L. Stevens, G. D. Seybold, W. L. Purcell, and L. V. Kimmel, A Risk-Based Fault Tree Analysis Method for Identification, Preliminary Evaluation, and Screening of Potential Accidental Release Sequences in Nuclear Fuel Cycle Operations, BNWL-1959, Battelle, Pacific Northwest Laboratories, Richland, WA, January 1976.

2. J. L. Carter, ACORN: A Computer Program for Plotting Fault Trees, BNWL-2144, Battelle, Pacific Northwest Laboratories, Richland, WA, October 1977.

3. P. J. Pelto and W. L. Purcell, MFAULT: A Computer Program for Analyzing Fault Trees, BNWL-2145, Battelle, Pacific Northwest Laboratories, Richland, WA, October 1977.

4. USAEC, Reactor Safety Study, An Assessment of Accident Risks in Commercial Nuclear Power Plants, WASH-1400, October 1975.

5. H. E. Lambert, Systems Safety Analysis and Fault Tree Analysis, UCID-16238, Lawrence Livermore Laboratory, University of CA, May 1973.

APPENDIX A

NAMELIST

This version of the NAMELIST input package was developed specifically for RAFT. Certain qualities were built into it which allow for some degree of quality assurance. The package was designed to be used as an independent entity within other codes with as little effort as possible for inclusion or omission.

Because of the voluminous aspects of fault tree problems, a loader similar to NAMELIST is desirable, provided it can be tailored to satisfy the problem. Therefore, the package was written as a set of FORTRAN subroutines with no reliance upon the compiler NAMELIST statement. This means that the NAMELIST table is constructed at execution time rather than at compilation time.

It would not have been difficult to create the package using the NAMELIST command statement; however, each computer's operating system contains different anomalies and modification is normally unavailable to the programmer.

The omission of the NAMELIST statement does, however, cause some inconvenience for the programmer. It now becomes necessary to link the package by the use of a LOAD subroutine in order to construct the NAMELIST table. This inconvenience is not so severe as it may suggest. Once some control parameters are supplied via the main program, the versatility of the package becomes quite obvious. Furthermore, construction of the table at execution time allows the programmer to detect and trap error conditions under program control. This is not always available with systems supplied NAMELIST loaders.

Comprehensive error diagnostics are built into the package, the purpose of which is twofold:

1. If possible, supply the user with diagnostics indicating the trouble area.
2. Except for extreme conditions, allow for the continuance of editing, rather than relying on the ill-fated classical dump.

The construction of the NAMELIST table at execution time allows another very important feature. Namely, the table can be expanded or contracted with the use of program control parameters rather than the necessary recompilation of the code. This feature is well suited to accommodate problems where it becomes necessary to change loading patterns in the same execution, change variable dimensions in subsequent runs, or completely alter the input parameters in chained programs.

The package, although developed on a CYBER, is actually general purpose. There are only five items inherent to it which would require change in order to convert to another computer:

1. The LOCF system function for retrieving program variable addresses must be available in some alternative form.
2. ENCODE/DECODE statements would need altering.
3. Variable FORMAT data statements need to be adjusted to the host machine.
4. The number of characters per machine word (currently 10) would need to be taken into account.
5. Word boundaries must be considered in order to guarantee proper word storage in the event that variables defined as bytes were used.

LINKAGE

Only two program modules are used in order to provide the NAMELIST linkage. The main program and an associated loading subroutine. If we assume the two modules are represented by MAIN and LOAD, we can illustrate a typical linkage pattern by the following:

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      COMMON NEWOLD(2),IN,OUT,STATUS,TABLE(42),FIELD
      INTEGER OUT,STATUS,TABLE,FIELD
      FIELD=80
      NEWOLD(1)="NEW"
      NEWOLD(2)="OLD"
      IN=5
      OUT=6
      STATUS=NEWOLD(1)
C     BEGINNING OF NEW CASE.
  100 CALL LOAD,RETURNS(7777)
C     "
C     "
C     "
C     REMAINING HOST PROGRAM LOGIC.
C     "
C     "
C     "
      GO TO 100
 7777 STOP
      END
      SUBROUTINE LOAD,RETURNS(IQUIT)
      COMMON NEWOLD(2),IN,OUT,STATUS,TABLE(42),FIELD
      INTEGER OUT,STATUS,TABLE,FIELD
      COMMON TYPE,TITLE(8,2),LOT(100,10),ZDZSDS(6,5,25),POWER(3,3),Q(8)
      LOGICAL TYPE
      COMPLEX POWER
      DOUBLE PRECISION Q
C     TABLE(DIM),DIM.GE.(7*(NUMBER OF NAMTAB ENTRIES))
C     CALL NEWTAB("NEW") MEANS INITIALIZE START OF NEW NAMELIST TABLE.
C     CALL NEWTAB("OLD") MEANS RETAIN OLD TABLE AND/OR
C                        ADD NEW NAMELIST PARAMETERS AND/OR
C                        MODIFY DIMENSIONS OF EXISTING NAMELIST PARAMETERS.
      EQUIVALENCE (DOUBLE,Q(1)),(COMPLX,POWER(1,1))
C     IMPLICIT TYPE=1=LOGICAL OR ALPHANUMERIC
C     IMPLICIT TYPE=2=INTEGER
C     IMPLICIT TYPE=3=REAL
C     IMPLICIT TYPE=4=DOUBLE PRECISION
C     IMPLICIT TYPE=5=COMPLEX
      IF(STATUS.NE.NEWOLD(1)) GO TO 100
      CALL NEWTAB(STATUS)
      CALL NAMTAB(TYPE,"TYPE",1,0,0,0,0,TABLE)
      CALL NAMTAB(TITLE,"TITLE",1,2,8,2,0,TABLE)
      CALL NAMTAB(LOT,"LOT",2,2,100,10,0,TABLE)
      CALL NAMTAB(ZDZSDS,"ZDZSDS",3,3,6,5,25,TABLE)
      CALL NAMTAB(COMPLX,"POWER",5,2,3,3,0,TABLE)
      CALL NAMTAB(DOUBLE,"Q",4,1,8,0,0,TABLE)
  100 CALL READ(IN,OUT,FIELD,"RAFT",TABLE),RETURNS(200)
      RETURN
  200 RETURN IQUIT
      END
```

The MAIN program is set up to illustrate the simplest of structure. The NAMELIST control parameters (NEWOLD, IN, OUT, STATUS, FIELD) have been identified prior to case execution logic. They are described as follows:

NEWOLD(I),I=1,2 must be preset to the words 'NEW' and 'OLD', respectively. They represent the two meanings that STATUS can assume.

STATUS is the decision flag which signals the initialization or continuance of the NAMELIST table. When STATUS=NEWOLD(1), a new table is assumed, and the appropriate calls to NEWTAB and NAMTAB must be made to construct the table. If STATUS=NEWOLD(2), the code assumes the table has been constructed and there is no further necessity to reconstruct or modify its parameters for subsequent case execution.

IN and OUT define the input and output logical units, respectively.

FIELD defines the highest numbered column on the input record which is assumed active. Any remaining columns are ignored by NAMELIST and may be used at the discretion of the analyst.

Three subroutines are critical for the use of the package: NEWTAB, NAMTAB, and READ. The use and purpose of each is discussed in the following paragraphs.

NEWTAB sets up internal NAMELIST commands which signal the initialization of a new table. It must be called at least once with STATUS=NEWOLD(1) if NAMELIST is to be used. The argument, STATUS, is returned as STATUS=NEWOLD(2) each time NEWTAB is called.

NAMTAB is the subroutine which constructs the NAMELIST table. This routine must be called for each variable which is to be used as a NAMELIST input parameter. It performs the same function as the FORTRAN NAMELIST statement, except that the table is constructed at execution time. NAMTAB uses eight arguments in order to construct the table.

Argument 1/This is the host programs' mneumonic name (or similarly EQUIVALENCED variable) which is used for input storage.

Argument 2/The second argument defines (in hollerith notation) the name selected by the analyst which is to be punched on the input card to identify data. The argument need not be the same name as the first argument but simply identifies a cross-reference of input card name to program variable.

Argument 3/This argument identifies the implicit type of the parameter. It can assume values which range from one to five, inclusive. The integer value chosen assumes the following definitions:

1. The parameter is either type LOGICAL or hollerith.
2. The parameter is type INTEGER.
3. The parameter is type REAL.
4. The parameter is type DOUBLE PRECISION.
5. The parameter is type COMPLEX.

Argument 4/Identifies the number of subscripted indices the variable contains. Currently the value chosen must be in the range zero to three, inclusive.

Arguments 5, 6, 7/These arguments specify the first, second, and third dimensions of subscripted parameters, respectively. Zero is used if no subscript is implied.

Argument 8/The last argument, TABLE, represents an array which contains the NAMELIST table. It must be contained by the host program and is dimensioned by at least seven times the number of calls to NAMTAB.

READ is the subroutine which loads data from the input records. It is called to load data for each new case. It contains five arguments and a RETURNS statement. Arguments 1, 2, 3, and 5 (IN, OUT, FIELD, and TABLE, respectively) have already been discussed. The fourth argument is the NAMELIST group name (in hollerith form). The RETURNS statement is used to trap end-of-file status when reading input records. It is shown in subroutine LOAD as a means of returning control to the MAIN program in the event no further processing is required.

Once the linkage has been programmed, the NAMELIST package may be used freely as if a systems installation package was supplied. However, this package contains a few added features which are not always available. They are listed here to illustrate some additional flexibility:

a) NAMELIST group names and input parameter names may contain up to seven characters.

b) Blanks may be used as data delimiters; however, they may not be inbedded in names in violation of NAMELIST rules.

c) Hollerith input is allowed so long as it is delimited by apostrophes.

d) String loading is allowed on all input forms.

e) Comments are allowed in the input stream by using either a "C" or an "*" in column one. The entire input record is considered a nonloading field and is virtually ignored.

f) The active record field length is completely flexible and adjustable through input.

g) Error diagnostics are supplied to the user in the event of any anomaly.

h) Program dimensions can never be exceeded via input demands.

i) Dynamic table length and variable dimensions are allowed through subsequent case execution.

j) Any illegal load or invalid parameter name is nullified.

k) Error status is trapped for host program control.

l) All formal NAMELIST rules are recognized so that data sets never need to be updated for use with this package.

The NAMELIST package will load into approximately 4,300 octal words of memory. This excludes the LOAD routine and the size of the variable TABLE used to contain the NAMELIST table. Loading is accomplished in approximately the same speed as would be required through a supplied systems package.

Except for the required linkage pattern, the package is written in closed form so that the user need not be concerned with its interacting functions.

APPENDIX B

RAFT CROSS-REFERENCE TABLE

This cross-reference table provides a paged and alphabetized listing of the entire content of the RAFT code. The left most column represents a program variable, subroutine name, or system routine name or library. The remaining columns represent the subroutines with which they reside, similar to a compiler reference map.

```
A
ABS                                                      DSKCRD                          LOAD
ABSUM      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
ACCEPT
ALGOR
AMAX1      ACCEPT
B
BLANK      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
BMPTAB     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
BRANCH     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
CARD                                                                   ERRORS
CARDS      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
CHAIN
CHECK                                                                             LOADER
CLOSEM
COLUMN                           CHAIN      CHECK                                 LOADER
COMMA
CP         ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
CUTOFA     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
CUTOFF     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
CUTSET     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
DEBUG      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
DIGITS     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
DIM1
DIM2
DIM3
DISC
DOLLAR
DSKCRD
DUMPF      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
EMPTY      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
EOF                                                         DSKCRD                          LOAD
EQUAL
ERRORS                                      CHECK
EVENTS     ACCEPT     ALGOR                     .DISC       DSKCRD                          LOAD
EXTRA      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
FALSE                            CHAIN      CHECK                                 LOADER
FAT        ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
FIELD      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
FILENA
FIT        ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
FLAG       ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
FLOAT
FORMAT                           CHAIN      CHECK                                 LOADER
FULL       ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
GET
GNAME      ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
HASH       ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
HSHING     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
HSHPNT     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
HSHTAB     ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
IABS                                                                                                   LOCATE
ICOL                             CHAIN                                 ERRORS
IMPTYP                                                                            LOADER
IN         ACCEPT     ALGOR                      DISC       DSKCRD                          LOAD
INC                              CHAIN      CHECK                                 LOADER               LOCATE
INDIC
```

INDICE	ACCEPT	ALGOR	CHAIN	CHECK	DISC	DSKCRD		LOADER	LOAD	
IC	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
IST								LOADER		
ISTORE					DISC					
ITEMS	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
J1										
J10										
J2										
J3										
J4										
J5										
J6										
J7										
J8										
J9										
JCOL							ERRORS			
JMP	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
JREC	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
JUMP							ERRORS			
K1			CHAIN							
K2			CHAIN							
K3			CHAIN							
KEY	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
KEYS			CHAIN	CHECK				LOADER		
KOUNT	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
KREC	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
KTR	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
LEFT										
LENGTH			CHAIN	CHECK				LOADER		
LETS			CHAIN	CHECK				LOADER		
LIMIT	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
LIMITS			CHAIN	CHECK				LOADER		
LINES										
LIST			CHAIN	CHECK				LOADER		LOCATE
LOAD										
LOADER										
LOCATE										
LOCATR										
LOCF								LOADER		
LOCPOS			CHAIN	CHECK				LOADER		
LOCUS			CHAIN	CHECK				LOADER		
LOG			CHAIN	CHECK				LOADER		
LOGIC			CHAIN	CHECK				LOADER		LOCATE
LOGICS			CHAIN	CHECK				LOADER		LOCATE
LU						DSKCRD				
LUDISK	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
MAXG				CHECK				LOADER	LOAD	
MAXWRT	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
MEDIA	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
MEMLOC	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
MEMORY	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
MINO		ALGOR			DISC			LOADER	LOAD	
MINOR			CHAIN	CHECK				LOADER		
MOD										
MODE	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
MODIFY	ACCEPT	ALGOR			DISC	DSKCRD			LOAD	
MOVEAB								LOADER		
N										
NAME										LOCATE
NAMES										
NAMTAB									LOAD	
NAP1			CHAIN	CHECK				LOADER		
NAP2			CHAIN	CHECK				LOADER		
NPLNKS										
NPELIM			CHAIN	CHECK				LOADER		
NPR								LOADER		

```
NET
NEWOLD        ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
NEWTAB                                                                                                      LOAD
NMLIST
NN                                                                                              LOADER
NN1
NN10
NN2
NN3
NN4
NN5
NN6
NN7
NN8
NN9
NNN           ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
NWLOTE                                CHAIN       CHECK                                         LOADER
NULL                                  CHAIN       CHECK                                         LOADER
NUM           ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
NUMAST                                CHAIN       CHECK                                         LOADER
NUMBER
OPENM
OUT           ACCEPT      ALGOR       CHAIN       CHECK       DISC       DSKCRD      ERRORS     LOADER      LOAD
OUTS
PACKED
PAGE          ACCEPT      ALGOR
PAREN1                                CHAIN       CHECK                                         LOADER
PAREN2                                CHAIN       CHECK                                         LOADER
PC            ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
PRIME         ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
PRINT
PRNT          ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
PROB34        ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
PRODCT        ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
PUT                                                           DISC
PUTIN
PUTOUT
QUOTE
RANK          ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
READ                                                                                                        LOAD
RECNO         ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
RECORD
REPAIR
RF            ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
RFCP          ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
RIGHT
ROW           ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
SCAN
SIGN2
SIGN34
SIGN5
SIGN7
SIGN8
SIZE          ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
SPEED
STACK
STAR
START                                 CHAIN       CHECK                  DSKCRD                 LOADER
STATE
STATUS        ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
STOP                                  CHAIN       CHECK                                         LOADER
STORE                                                         DISC
SUMAR         ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
TABLE         ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
TABLES
TTA           ACCEPT      ALGOR                               DISC       DSKCRD                             LOAD
```

```
[illegible]    ACCEPT     ALLOC                                 DISC      DSKCRD                        LOAD
[illegible]    ACCEPT     ALLOC                                 DISC      DSKCRD                        LOAD
[illegible]    ACCEPT     ALLOC                                 DISC      DSKCRD                        LOAD
[illegible]    [illegible] [illegible]                          DISC      DSKCRD                        LOAD
[illegible]                                                               DSKCRD
[illegible]                          [illegible]  [illegible]                               LOADER
[illegible]
[illegible]    ACCEPT     ALLOC                                 DISC      DSKCRD                        LOAD
[illegible]
[illegible]
[illegible]
[illegible]
```

```
A                   MOVEAB
ABS                                                                                 READ
ABSUM                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT
ACCEPT
ALGOR
AMAX1
B                   MOVEAB
BLANK                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT    READ
BMPTAB                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
BRANCH                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
CARD
CARDS                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT
CHAIN
CHECK
CLOSEM                                                                      RAFT
COLUMN                      NAMTAB  NEWTAB                                          READ
COMMA                                                                               READ
CP                                          PAGE    PRINT   PUTIN   PUTOUT  RAFT
CUTOFA                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
CUTOFF                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
CUTSET                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
DEBUG                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT
DIGITS                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
DIM1                        NAMTAB
DIM2                        NAMTAB
DIM3                        NAMTAB
DISC
DOLLAR                                                                              READ
DSKCRD
DUMPF                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT
EMPTY                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT
EOF                                                                                 READ
EQUAL                                                                               READ
ERRORS                                                                              READ
EVENTS                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
EXTRA                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT
FALSE                       NAMTAB  NEWTAB                                          READ
FAT                                         PAGE    PRINT   PUTIN   PUTOUT  RAFT
FIELD                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT    READ
FILEWA                                                                      RAFT
FIT                                         PAGE    PRINT   PUTIN   PUTOUT  RAFT
FLAG                                        PAGE    PRINT   PUTIN   PUTOUT  RAFT
FLOAT
FORMAT                      NAMTAB  NEWTAB                                          READ
FULL                                        PAGE    PRINT   PUTIN   PUTOUT  RAFT
GET                                                                 PUTOUT
GNAME                                       PAGE    PRINT   PUTIN   PUTOUT  RAFT
HASH                                        PAGE    PRINT   PUTIN   PUTOUT  RAFT
HSHING                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
HSHCNT                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
HSHTAB                                      PAGE    PRINT   PUTIN   PUTOUT  RAFT
IABS                        NAMTAB          PAGE
ICOL
IMPTYP
IN                                          PAGE    PRINT   PUTIN   PUTOUT  RAFT    READ
INC                         NAMTAB  NEWTAB                                          READ
INDIC           LOCATR
```

```
INDICE                        NAMTAB    NEWTAB    PAGE      PRINT     PUTIN     PUTOUT    RAFT      READ
IO                                                PAGE      PRINT     PUTIN     PUTOUT    RAFT
IST
ISTORE                                                                          PUTOUT
ITEMS                                             PAGE      PRINT     PUTIN     PUTOUT    RAFT
J1
J10
J2
J3
J4
J5
J6
J7
J8
J9
JCON
JMP                                               PAGE      PRINT     PUTIN     PUTOUT    RAFT
JREC                                              PAGE      PRINT     PUTIN     PUTOUT    RAFT
JUMP
K1
K2
K3
KEY                                               PAGE      PRINT     PUTIN     PUTOUT    RAFT
KEYS                          NAMTAB    NEWTAB                                                      READ
KOUNT                                             PAGE      PRINT     PUTIN     PUTOUT    RAFT
KREC                                              PAGE      PRINT     PUTIN     PUTOUT    RAFT
KTR                                               PAGE      PRINT     PUTIN     PUTOUT    RAFT
LEFT                                                                                                READ
LENGTH                        NAMTAB    NEWTAB                                                      READ
LETS                          NAMTAB    NEWTAB                                                      READ
LIMIT                                             PAGE      PRINT     PUTIN     PUTOUT    RAFT
LIMITS                        NAMTAB    NEWTAB                                                      READ
LINES                                             PAGE
LIST        LOCATR            NAMTAB    NEWTAB                                                      READ
LOAD                                                                                      RAFT
LOADER
LOCATE
LOCATR
LOCF                          NAMTAB
LOCPOS                        NAMTAB    NEWTAB                                                      READ
LOCUS                         NAMTAB    NEWTAB                                                      READ
LOG                           NAMTAB    NEWTAB                                                      READ
LOGIC                         NAMTAB    NEWTAB                                                      READ
LOGICS                        NAMTAB    NEWTAB                                                      READ
LU
LUDISK                                            PAGE      PRINT     PUTIN     PUTOUT    RAFT
MAXC        LOCATR
MAXVRT                                            PAGE      PRINT     PUTIN     PUTOUT    RAFT
MEDIA                                             PAGE      PRINT     PUTIN     PUTOUT    RAFT
MEMLOC                                            PAGE      PRINT     PUTIN     PUTOUT    RAFT
MEMORY                                            PAGE      PRINT     PUTIN     PUTOUT    RAFT
MINO
MINOR       LOCATR            NAMTAB    NEWTAB                                                      READ
MOD
MODE                                              PAGE      PRINT     PUTIN     PUTOUT    RAFT
MODIFY                                            PAGE      PRINT     PUTIN     PUTOUT    RAFT
MOVEAR
N                     MOVEAR
NAME                          NAMTAB                                                                READ
NAMES                         NAMTAB
NAMTAB
NAP1                          NAMTAB    NEWTAB                                                      READ
NAP2                          NAMTAB    NEWTAB                                                      READ
NBLNKS
NDELIM                        NAMTAB    NEWTAB                                                      READ
NOP
```

```
NET         LOCATR
NEWOLD                                PAGE    PRINT   PUTIN   PUTOUT  RAFT
NEWTAB
NMLIST                                                                        READ
NN
NN1
NN10
NN2
NN3
NN4
NN5
NN6
NN7
NN8
NN9
[illegible]                           PAGE    PRINT   PUTIN   PUTOUT  RAFT
NOTE                    NAMTAB    NEWTAB                                      READ
NULL                    NAMTAB    NEWTAB                                      READ
[illegible]                           PAGE    PRINT   PUTIN   PUTOUT  RAFT
NUMBRT                  NAMTAB    NEWTAB                                      READ
NUMBER                  NAMTAB
OPEN                                                                  RAFT
OUT                     NAMTAB    NEWTAB  PAGE    PRINT   PUTIN   PUTOUT  RAFT    READ
OUTS                                                                          READ
PACKED
PAGE                                          PRINT   PUTIN   PUTOUT  RAFT
PAREN1                  NAMTAB    NEWTAB                                      READ
PAREND                  NAMTAB    NEWTAB                                      READ
PL                                    PAGE    PRINT   PUTIN   PUTOUT  RAFT
PRIME                                 PAGE    PRINT   PUTIN   PUTOUT  RAFT
PRINT                                                         PUTOUT
PRNT                                  PAGE    PRINT   PUTIN   PUTOUT  RAFT
PROGRAM                               PAGE    PRINT   PUTIN   PUTOUT  RAFT
PROMPT                                PAGE    PRINT   PUTIN   PUTOUT  RAFT
PUT
PUTIN                                                                 RAFT
PUTOUT
QUOTE                                                                         READ
RANK                                  PAGE    PRINT   PUTIN   PUTOUT  RAFT
READ
RECUR                                 PAGE    PRINT   PUTIN   PUTOUT  RAFT
RECORD                                                                RAFT
REPORT
RF                                    PAGE    PRINT   PUTIN   PUTOUT  RAFT
RECR                                  PAGE    PRINT   PUTIN   PUTOUT  RAFT
RIGHT                                                                         READ
ROW                                   PAGE    PRINT   PUTIN   PUTOUT  RAFT
SCAN                                                                          READ
SIGN2                                                                         READ
SIGN34                                                                        READ
SIGN5                                                                         READ
SIGNIT                                                                        READ
[illegible]                                                                   READ
SIZE                                  PAGE    PRINT   PUTIN   PUTOUT  RAFT
SPEED
STACK
STAR                                                                          READ
START                   NAMTAB    NEWTAB                                      READ
STATE                             NEWTAB
STATUS                                PAGE    PRINT   PUTIN   PUTOUT  RAFT
STOP                    NAMTAB    NEWTAB                                      READ
STORE                                                         PUTOUT
SUMAR                                 PAGE    PRINT   PUTIN   PUTOUT  RAFT
TABLE                                 PAGE    PRINT   PUTIN   PUTOUT  RAFT
TABLES
TTA                                   PAGE    PRINT   PUTIN   PUTOUT  RAFT
```

```
[illegible]
TITLE                                         PAGE      PRINT     PUTIN     PUTOUT    RAFT
TITLES                                        PAGE      PRINT     PUTIN     PUTOUT    RAFT
TREE                                          PAGE      PRINT     PUTIN     PUTOUT    RAFT
TREES                                         PAGE      PRINT     PUTIN     PUTOUT    RAFT
TRIP
TRUE                      NAMTAB    NEWTAB                                                    READ
TYPE                      NAMTAB
UNAVL                                         PAGE      PRINT     PUTIN     PUTOUT    RAFT
UNPICK
[illegible]
[illegible]               NAMTAB
[illegible]
```

```
A
ABS         RECORD
ABSUM       RECORD    REPAIR                                                          SPEED     STACK
ACCEPT      RECORD
ALGOR       RECORD
AMAX1
B
BLANK       RECORD    REPAIR              SIGN2     SIGN34                            SPEED     STACK
BMPTAB      RECORD    REPAIR                                                          SPEED     STACK
BRANCH      RECORD    REPAIR                                                          SPEED     STACK
CARD
CARDS       RECORD    REPAIR                                                          SPEED     STACK
CHAIN                                             SIGN34    SIGN5     SIGN7
CHECK
CLOSEM      RECORD
COLUMN                          SCAN      SIGN2   SIGN34    SIGN5     SIGN7     SIGN8
COMMA                           SCAN              SIGN34    SIGN5
CP          RECORD    REPAIR                                                          SPEED     STACK
CUTOFA      RECORD    REPAIR                                                          SPEED     STACK
CUTOFF      RECORD    REPAIR                                                          SPEED     STACK
CUTSET      RECORD    REPAIR                                                          SPEED     STACK
DEBUG       RECORD    REPAIR                                                          SPEED     STACK
DIGITS      RECORD    REPAIR                                                          SPEED     STACK
DIM1
DIM2
DIM3
DISC        RECORD
DOLLAR                                            SIGN34
DSKCRD      RECORD
DUMPF       RECORD    REPAIR                                                          SPEED     STACK
EMPTY       RECORD    REPAIR                                                          SPEED     STACK
EOF         RECORD
EQUAL
ERRORS                          SCAN      SIGN2   SIGN34    SIGN5     SIGN7     SIGN8
EVENTS      RECORD    REPAIR                                                          SPEED     STACK
EXTRA       RECORD    REPAIR                                                          SPEED     STACK
FALSE                           SCAN      SIGN2   SIGN34    SIGN5     SIGN7     SIGN8
FAT         RECORD    REPAIR                                                          SPEED     STACK
FIELD       RECORD    REPAIR                                                          SPEED     STACK
FILEWA
FIT         RECORD    REPAIR                                                          SPEED     STACK
FLAG        RECORD    REPAIR                                                          SPEED     STACK
FLOAT       RECORD    REPAIR
FORMAT                          SCAN      SIGN2   SIGN34    SIGN5     SIGN7     SIGN8
FULL        RECORD    REPAIR                                                          SPEED     STACK
GET
GNAME       RECORD    REPAIR                                                          SPEED     STACK
HASH        RECORD    REPAIR                                                          SPEED     STACK
HSHING      RECORD    REPAIR                                                          SPEED     STACK
HSHPNT      RECORD    REPAIR                                                          SPEED     STACK
HSHTAB      RECORD    REPAIR                                                          SPEED     STACK
IABS                                              SIGN34
ICOL
IMPTYP
IN          RECORD    REPAIR                                                          SPEED     STACK
INC                             SCAN      SIGN2   SIGN34    SIGN5     SIGN7     SIGN8
INDIC
```

```
INDICE    RECORD    REPAIR    SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8     SPEED     STACK
IO        RECORD    REPAIR                                                                SPEED     STACK
IST
ISTORE
ITEMS     RECORD    REPAIR                                                                SPEED     STACK
J1                                                                                        SPEED
J10                                                                                       SPEED
J2                                                                                        SPEED
J3                                                                                        SPEED
J4                                                                                        SPEED
J5                                                                                        SPEED
J6                                                                                        SPEED
J7                                                                                        SPEED
J8                                                                                        SPEED
J9                                                                                        SPEED
JCOL
JMP       RECORD    REPAIR                                                                SPEED     STACK
JREC      RECORD    REPAIR                                                                SPEED     STACK
JUMP
K1
K2
K3
KEY       RECORD    REPAIR                                                                SPEED     STACK
KEYS                          SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
KOUNT     RECORD    REPAIR                                                                SPEED     STACK
KREC      RECORD    REPAIR                                                                SPEED     STACK
KTS       RECORD    REPAIR                                                                SPEED     STACK
LEFT
LENGTH                        SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LETS                          SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LIMIT     RECORD    REPAIR                                                                SPEED     STACK
LIMITS                        SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LINES
LIST                          SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LOAD
LOADER                                            SIGN34                        SIGN8
LOCATE                                  SIGN2
LOCATR                                  SIGN2
LOOP
LOOPS                         SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LOCUS                         SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LOG                           SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LOGIC                         SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LOGICS                        SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
LU
LUDISK    RECORD    REPAIR                                                                SPEED     STACK
MAXC                                    SIGN2     SIGN34
MAXART    RECORD    REPAIR                                                                SPEED     STACK
MEDIA     RECORD    REPAIR                                                                SPEED     STACK
MEMLOC    RECORD    REPAIR                                                                SPEED     STACK
MEMORY    RECORD    REPAIR                                                                SPEED     STACK
MENU                                                                                                STACK
MINOR                         SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
MOD
MODE      RECORD    REPAIR                                                                SPEED     STACK
MODIFY    RECORD    REPAIR                                                                SPEED     STACK
MOVEAR
N                                       SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
NAME                                    SIGN2     SIGN34
NAMES
NAMTAB
NAP1                          SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
NAP2                          SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
NBLNKS                        SCAN
NDELIM                        SCAN      SIGN2     SIGN34    SIGN5     SIGN7     SIGN8
NOR
```

NET										
NEWOLD	RECORD	REPAIR							SPEED	STACK
NEWTAB										
NMLIST										
NM										
NM1									SPEED	
NM10									SPEED	
NM2									SPEED	
NM3									SPEED	
NM4									SPEED	
NM5									SPEED	
NM6									SPEED	
NM7									SPEED	
NM8									SPEED	
NM9									SPEED	
NMN	RECORD	REPAIR							SPEED	STACK
NOQUOTE			SCAN	SIGN2	SIGN34	SIGN5	SIGN7	SIGN8		
NULL			SCAN	SIGN2	SIGN34	SIGN5	SIGN7	SIGN8		
NUM	RECORD	REPAIR							SPEED	STACK
NUMAST			SCAN	SIGN2	SIGN34	SIGN5	SIGN7	SIGN8		
NUMBER										
OPEN	RECORD									
OUT	RECORD	REPAIR	SCAN	SIGN2	SIGN34	SIGN5	SIGN7	SIGN8	SPEED	STACK
OUTS										
PACKED										
PAGE	RECORD									
PAREN1			SCAN	SIGN2	SIGN34	SIGN5	SIGN7	SIGN8		
PAREN2			SCAN	SIGN2	SIGN34	SIGN5	SIGN7	SIGN8		
PP	RECORD	REPAIR							SPEED	STACK
PRIME	RECORD	REPAIR							SPEED	STACK
POINT										
PRNT	RECORD	REPAIR							SPEED	STACK
PROBAB	RECORD	REPAIR							SPEED	STACK
PRODCT	RECORD	REPAIR							SPEED	STACK
PUT										
PUTIN										
PUTOUT	RECORD									
QUOTE			SCAN		SIGN34					
RANK	RECORD	REPAIR							SPEED	STACK
READ										
RECNO	RECORD	REPAIR							SPEED	STACK
RECORD										
REPAIR	RECORD									
RE	RECORD	REPAIR							SPEED	STACK
RECP	RECORD	REPAIR							SPEED	STACK
RIGHT										
ROW	RECORD	REPAIR							SPEED	STACK
SCAN										
SIGN2										
SIGN34										
SIGN5										
SIGN7										
SIGN8										
SIZE	RECORD	REPAIR							SPEED	STACK
SPEED	RECORD									
STACK									SPEED	
STAR					SIGN34					
START			SCAN	SIGN2	SIGN34	SIGN5	SIGN7	SIGN8		
STATE										
STATUS	RECORD	REPAIR							SPEED	STACK
STOP			SCAN	SIGN2	SIGN34	SIGN5	SIGN7	SIGN8		
STORE										
SUMAR	RECORD	REPAIR							SPEED	STACK
TABLE	RECORD	REPAIR							SPEED	STACK
TABLES										
TIA	RECORD	REPAIR							SPEED	STACK

```
TITLE      RECORD   REPAIR                                                            SPEED    STACK
TITLES     RECORD   REPAIR                                                            SPEED    STACK
TREE       RECORD   REPAIR                                                            SPEED    STACK
TREES      RECORD   REPAIR                                                            SPEED    STACK
TRIP
TRUE                         SCAN     SIGN2    SIGN34   SIGN5    SIGN7    SIGN8
TYPE
WAVE       RECORD   REPAIR                                                            SPEED    STACK
UNPACK
UNPACKR
VARS
WRED       RECORD
```

[illegible]		
[illegible]		
[illegible]	TABLES	NEED
ACCEPT		
[illegible]		
[illegible]		
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
BRANCH	TABLES	NEED
[illegible]		
[illegible]	TABLES	NEED
[illegible]		
[illegible]		
[illegible]		
[illegible]		
[illegible]		
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
OUTSET	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]		
[illegible]		
[illegible]		
[illegible]		
[illegible]		
[illegible]		
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]		
[illegible]		
[illegible]		
[illegible]	TABLES	NEED
EXTRA	TABLES	NEED
FALSE		
[illegible]	TABLES	NEED
FIELD	TABLES	NEED
[illegible]		
[illegible]	TABLES	NEED
FLAG	TABLES	NEED
FLOAT		
FORMAT		
[illegible]	TABLES	NEED
GET	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]	TABLES	NEED
[illegible]		
[illegible]		
[illegible]		
[illegible]	TABLES	NEED
[illegible]		
[illegible]		

[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]			
[illegible]			
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED
[illegible]		[illegible]	
[illegible]		[illegible]	
[illegible]	TABLES		NEED
[illegible]	TABLES		NEED

```
[illegible]
[illegible]        TABLES                        NEED
[illegible]
[illegible]
[illegible]                         [illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]        TABLES                        NEED
[illegible]
[illegible]
[illegible]        TABLES                        NEED
[illegible]
[illegible]
[illegible]
[illegible]        TABLES                        NEED
[illegible]
[illegible]                         [illegible]
[illegible]        TABLES
[illegible]
[illegible]
[illegible]        TABLES                        NEED
[illegible]        TABLES                        NEED
[illegible]
[illegible]        TABLES                        NEED
[illegible]        TABLES                        NEED
[illegible]        TABLES                        NEED
[illegible]                                      NEED
[illegible]
[illegible]
[illegible]
[illegible]        TABLES                        NEED
[illegible]
[illegible]        TABLES                        NEED
[illegible]
[illegible]
[illegible]        TABLES                        NEED
[illegible]        TABLES                        NEED
[illegible]
[illegible]        TABLES                        NEED
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
SIZE               TABLES                        NEED
SPEED
STACK
[illegible]
START
STATE
STATUS             TABLES                        NEED
STOP
STORE
SYMAB              TABLES                        NEED
TABLE              TABLES                        NEED
TABLES                                           NEED
TTA                TABLES                        NEED

[illegible]        TABLES                        NEED
[illegible]        TABLES                        NEED
[illegible]        TABLES                        NEED
[illegible]        TABLES                        NEED
[illegible]
[illegible]
[illegible]
[illegible]        TABLES                        NEED
[illegible]        TABLES
[illegible]
[illegible]
[illegible]
```

DISTRIBUTION

No. of
Copies

OFFSITE

UNITED STATES

A. A. Churm
U.S. DOE Chicago Patent Group
9800 South Cass Avenue
Argonne, IL 60439

27 U.S. DOE Technical Information
Center

W. Brobst
U.S. DOE Division of
Environmental Control Technology
Transportation Branch
U.S. DOE Headquarters
Washington, DC 20545

J. A. Sisler
U.S. DOE Division of
Environmental Control Technology
Transportation Branch
U.S. DOE Headquarters
Washington, DC 20545

W. K. Eister
U.S. DOE Division of Nuclear
Fuel Cycle and Production
Washington, DC 20545

R. D. Walton
U.S. DOE Division of Nuclear
Fuel Cycle and Production
Washington, DC 20545

G. E. Apostolakis
UCLA
5532 Boelter Hall
Los Angeles, CA 90024

No. of
Copies

R. E. Barlow
UCLA, Berkeley
Berkeley, CA 94720

G. R. Burdick
Aerojet Nuclear Company
550 Second Street
Idaho Falls, ID 83401

H. C. Claiborne
Union Carbide Corporation (HNL)
P. O. Box Y
Oak Ridge, TN 37830

J. Curtis
Department of Transportation
Materials Transportation Bureau
2100 Second St. S.W.
Washington, DC 20590

W. S. Durant
duPont Company, Aiken (U.S. DOE)
E. I. duPont DeNemours and Co.
Savannah River Laboratory
Aiken, SC 29801

D. Egan
Environmental Protection Agency
5207 Longsdale Drive
Springfield, VA 22151

R. C. Erdmann
Science Applications, Inc.
2680 Hanover Street
Palo Alto, CA 94303

R. R. Fullwood
Science Applications, Inc.
2680 Hanover Street
Palo Alto, CA 94303

No. of
Copies

J. B. Fussell
University of Tennessee
Department of Nuclear Engineering
Knoxville, TN 37916

D. F. Haasl
Institute of System Sciences
3085 North Bar Drive
North Bend, OR 97459

J. D. Hill
Battelle Memorial Institute
505 King Avenue
Columbus, OH 43201

R. Howard
Stanford University
Stanford, CA 93045

B. R. Judd
Management Analysis Co.
Palo Alto, CA 94303

H. E. Lambert
Lawrence Livermore Laboratory
Livermore, CA 94550

G. S. Lellouche
Electric Power Research Institute
3412 Hillview Avenue
P.O. Box 10412
Palo Alto, CA 94303

S. E. Logan
University of New Mexico
Albuquerque, NM 87131

D. Okrent
UCLA
5532 Boelter Hall
Los Angeles, CA 90024

T. G. Priddy
Sandia Laboratories
P.O. Box 5800
Albuquerque, NM 87115

No. of
Copies

N. Rasmussen
Massachusetts Institute of
Technology
Cambridge, MA 02139

W. Rowe
Environmental Protection Agency
401 M. Street
Washington, DC 20460

P. Rubel
Union Carbide Corporation (HNL)
P.O. Box Y
Oak Ridge, TN 37830

T. H. Smith
EEG Company
Idaho Falls, ID 83401

D. Turner
Union Carbide Corporation
Office of Waste Isolation
P.O. Box Y
Oak Ridge, TN 37830

W. E. Vesely
NRC Office of Nuclear Regulatory
Research
Washington, DC 20545

I. B. Wall
NRC Office of Nuclear Regulatory
Research
Washington, DC 20545

R. Williams
Electric Power Research Institute
3412 Hillview Avenue
P.O. Box 10412
Palo Alto, CA 94304

FOREIGN

G. D. Bell
United Kingdom Atomic Energy
Authority
Safety and Reliability
Directorate
Warrington WA3 4NE
UNITED KINGDOM

No. of
Copies

H. W. M. Braun
C.C.R. EURATOM
21020 Ispra (Va)
C.P. 5
ITALIA

H. Dyroff
NUKEM GmbH
645 Hanau 11
Postfach 110080
WEST GERMANY

Ferruccio Gera
Laboratorio Rifiuti
  Radioattivi
C.N.E.N. - C.S.N. Casaccia
Casella Postale 2400
00100 Rome, ITALY

S. Hartwig
Battelle Institute, e.v.
Am Romerhof 35
600 Frankfurt Main 90
GERMANY

International Atomic Energy
  Agency
Kärtner Ring 11
P.O. Box 590
A-1011, Vienna, AUSTRIA

H. Krause
Nuclear Research Center
Waste Management Department
D75 Karlsruhe
Weberstr. 5
GERMANY

M. Stammler
Battelle-Institute, e.v.
6000 Frankfurt am Main 90
Am Romerhuf 35
Frankfurt, GERMANY

No. of
Copies

H. J. Wingender
NUKEM GmbH
645 Hanau 11
Postfach 110080
WEST GERMANY

H. Witte
NUKEM GmbH
645 Hanau 11
Postfach 110080
WEST GERMANY

ONSITE

3 US DOE Richland Operations Office

H. E. Ransom
R. B. Goranson
M. J. Zamorski

4 Rockwell Hanford Operations

R. A. Deju
R. E. Isaacson
D. D. Wodrich
File Copy

5 Exxon

J. L. Carter

1 Joint Center for Graduate Study

J. Cooper

1 United Nuclear Industries, Inc.

P. A. Crosetti

3 HEDL

J. E. Ecker
J. W. Hagen
D. E. Simpson

No. of
Copies

64 Battelle-Northwest

N. E. Carter
J. C. Fox
R. J. Hall
S. W. Heaberlin
J. W. Lichfield
T. I. McSweeney
E. S. Murphy
P. J. Pelto (10)
A. M. Platt
W. L. Purcell
R. E. Rhoads
G. D. Seybold (10)
D. L. Stevens
J. W. Voss
L. D. Williams
W. K. Winegardner (25)
Technical Information (5)
Technical Publications