# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Program</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>BREAKUP</td>
<td>1</td>
</tr>
<tr>
<td>General Description</td>
<td>1</td>
</tr>
<tr>
<td>Specific Input Instructions</td>
<td>2</td>
</tr>
<tr>
<td>Dimension Restrictions</td>
<td>2</td>
</tr>
<tr>
<td>Sample Network</td>
<td>3</td>
</tr>
<tr>
<td>Sample Input</td>
<td>4</td>
</tr>
<tr>
<td>Sample Output</td>
<td>5</td>
</tr>
<tr>
<td>Sample Problem: Schematic Representation</td>
<td>13</td>
</tr>
<tr>
<td>Program Listing</td>
<td>21</td>
</tr>
<tr>
<td>LOOP</td>
<td>33</td>
</tr>
<tr>
<td>Specific Input Instructions</td>
<td>33</td>
</tr>
<tr>
<td>Dimension Restrictions</td>
<td>34</td>
</tr>
<tr>
<td>Sample Problem</td>
<td>35</td>
</tr>
<tr>
<td>Program Listing</td>
<td>36</td>
</tr>
</tbody>
</table>
ABSTRACT

Statistical PERT is a new procedure for obtaining information about the distribution of a project's completion time when the project is comprised of a large number of activities and the time required to complete an individual activity once it can be begun is a random variable. The project is represented as an acyclic network whose arcs correspond to the project activities. This network is simplified by replacing various activity configurations by single equivalent activities and then decomposed into several subnetworks. The distribution and moments of each subnetwork's completion time are bounded and approximated on the basis of two points from each activity's completion time distribution by using some mathematical programming techniques and a new result in the theory of networks. The project's completion time distribution is then approximated by combining the approximate subnetwork distributions.

This report documents two computer programs. The first program BREAKUP decomposes a project network into several subnetworks which are connected in either series or parallel in the project network. The second program LOOP checks a given project network for loops (cycles) since any loop would contradict the assumed acyclic structure of the project network.
BREAKUP

General Description:

This program "breaks up" a network into a set of subnetworks which can be linked together either in series or parallel to yield the given network. This breakup is complete in the sense that none of the subnetworks in the set can be further broken up.

The basic breakup procedure involves two main subroutines, BUNDLE and CUT. BUNDLE partitions the activities in a given network or subnetwork into parallel subnetworks connecting the network's source and sink. CUT identifies the cut nodes in a given network or subnetwork and then identifies the sets of activities between each of the consecutive cut nodes. This series of activity sets represents a breakup of the given network or subnetwork into subnetworks in series. The complete breakup of the given network is the following sequential procedure:

(1) Use BUNDLE to identify the parallel subnetworks connecting the source and sink of the network.

(2) Use CUT separately on each parallel subnetwork identified in the previous step - (1) or (3) -, and breakup the parallel subnetwork into subnetworks in series. If no such breakup is possible for a parallel subnetwork, that subnetwork is not considered again. If no new series subnetwork is identified in this entire step, stop. Otherwise go to (3).

(3) Use BUNDLE separately on each series subnetwork identified in (2). If BUNDLE cannot breakup a series subnetwork, that subnetwork is not considered again. If no new parallel subnetwork is identified in this entire step,
A schematic example of this procedure accompanies the sample problem.

**Specific Input Instructions:**

Card 1. Col. 1-3: The number of arcs in the network, Format (I3).
   Col. 4-6: The number of the node which is the source node, Format (I3).
   Col. 7-9: The number of the node which is the sink node, Format (I3).
   Col. 10-12: The largest node number in the network, Format (I3).

For each activity one card with:

   Col. 1-3: The activity's number, Format (I3).
   Col. 4-6: The activity's origin node number, Format (I3).
   Col. 7-9: The activity's terminal node number, Format (I3).

The activities and nodes may be numbered in any way and may be read in any order.

**Dimension Restrictions:**

Currently this program will accommodate a network that has a maximum of 300 nodes and a maximum of 300 arcs. It can store a maximum of 100 subnetworks with each subnetwork having a maximum of 300 arcs.

This program is written in FORTRAN G.
SAMPLE INPUT

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>34</td>
<td>1</td>
<td>18</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>4</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>6</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1</td>
<td>13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>14</td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>16</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>15</td>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>17</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>19</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>19</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>20</td>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>13</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
INPUT STAGE

THE INITIAL NETWORK HAS 34 ARCS

THE SOURCE IS NODE NUMBER 1

THE SINK IS NODE NUMBER 19

THE LARGEST NODE IS NODE NUMBER 20

THE INITIAL NETWORK AS READ IN IS:

<table>
<thead>
<tr>
<th>ARC NUMBER</th>
<th>ORIGIN NODE</th>
<th>TERMINAL NODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>27</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>28</td>
<td>6</td>
<td>18</td>
</tr>
<tr>
<td>29</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>16</td>
<td>14</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>16</td>
<td>17</td>
</tr>
<tr>
<td>14</td>
<td>15</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>17</td>
</tr>
<tr>
<td>33</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>30</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td>19</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>33</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>22</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>32</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>33</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>34</td>
<td>20</td>
<td>18</td>
</tr>
<tr>
<td>24</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>25</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>26</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>13</td>
</tr>
</tbody>
</table>

SUBNET 1 IS COMPOSED OF SUBNETWORKS:

2, 3, 4, 5, 6,
STAGE 1 BREAKUP

SUBNETWORK 2 IS COMPOSED OF SUBNETWORKS:
7, 8, 9, 10,
IN SERIES

SUBNETWORK 3 IS COMPOSED OF SUBNETWORKS:
11, 12, 13,
IN SERIES

SUBNETWORK 4 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 1
SINK NODE = 18
ARC S(ARC) T(ARC)
29 1 18

SUBNETWORK 5 IS COMPOSED OF SUBNETWORKS:
14, 15, 16,
IN SERIES

SUBNETWORK 6 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 1
SINK NODE = 18
ARC S(ARC) T(ARC)
1 1 18
STAGE 2 BREAKUP

SUBNETWORK 7 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 1
SINK NODE = 7
ARC S(ARC) T(ARC)
 2   1   7

SUBNETWORK 8 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 7
SINK NODE = 11
ARC S(ARC) T(ARC)
 3   7   8
 4   7   9
 6   8   10
 8   8   11
 5   9   8
 7   9   10
 9   10  11

SUBNETWORK 9 IS COMPOSED OF SUBNETWORKS:
  17, 18,
IN PARALLEL

SUBNETWORK 10 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 17
SINK NODE = 18
ARC S(ARC) T(ARC)
 19  17  18

SUBNETWORK 11 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 1
SINK NODE = 4
<table>
<thead>
<tr>
<th>ARC</th>
<th>S(ARC)</th>
<th>T(ARC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>22</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>24</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

**SUBNETWORK 12 IS COMPOSED OF SUBNETWORKS:**

19, 20,

**IN PARALLEL**

**SUBNETWORK 13 IS A MINIMUM NETWORK**

IT IS COMPOSED OF:

<table>
<thead>
<tr>
<th>SOURCE NODE</th>
<th>= 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SINK NODE</td>
<td>= 18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ARC</th>
<th>S(ARC)</th>
<th>T(ARC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>6</td>
<td>18</td>
</tr>
</tbody>
</table>

**SUBNETWORK 14 IS A MINIMUM NETWORK**

IT IS COMPOSED OF:

<table>
<thead>
<tr>
<th>SOURCE NODE</th>
<th>= 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>SINK NODE</td>
<td>= 19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ARC</th>
<th>S(ARC)</th>
<th>T(ARC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>1</td>
<td>19</td>
</tr>
</tbody>
</table>

**SUBNETWORK 15 IS COMPOSED OF SUBNETWORKS:**

21, 22, 23,

**IN PARALLEL**

**SUBNETWORK 16 IS A MINIMUM NETWORK**

IT IS COMPOSED OF:

<table>
<thead>
<tr>
<th>SOURCE NODE</th>
<th>= 20</th>
</tr>
</thead>
<tbody>
<tr>
<td>SINK NODE</td>
<td>= 18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ARC</th>
<th>S(ARC)</th>
<th>T(ARC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>34</td>
<td>20</td>
<td>18</td>
</tr>
</tbody>
</table>
STAGE 3 BREAKUP

SUBNETWORK 17 IS COMPOSED OF SUBNETWORKS:
24, 29,
IN SERIES

SUBNETWORK 18 IS COMPOSED OF SUBNETWORKS:
26, 27, 28,
IN SERIES

SUBNETWORK 19 IS A MINIMUM NETWORK
IT IS COMPOSED OF:
SOURCE NODE = 4
SINK NODE = 6
ARC \((S(ARC), T(ARC))\)
27 4 6

SUBNETWORK 20 IS COMPOSED OF SUBNETWORKS:
29, 30,
IN SERIES

SUBNETWORK 21 IS A MINIMUM NETWORK
IT IS COMPOSED OF:
SOURCE NODE = 19
SINK NODE = 20
ARC \((S(ARC), T(ARC))\)
31 19 20

SUBNETWORK 22 IS A MINIMUM NETWORK
IT IS COMPOSED OF:
SOURCE NODE = 19
SINK NODE = 20
ARC \((S(ARC), T(ARC))\)
32 19 20
SUBNETWORK 23 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 19
SINK NODE = 20

ARC S(ARC) T(ARC)
33 19 20
STAGE 4 BREAKUP

SUBNETWORK 24 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 11
SINK NODE = 12
ARC S(ARC) T(ARC)
10 11 12

SUBNETWORK 25 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 12
SINK NODE = 17
ARC S(ARC) T(ARC)
18 12 17

SUBNETWORK 26 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 11
SINK NODE = 13
ARC S(ARC) T(ARC)
11 11 13

SUBNETWORK 27 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 13
SINK NODE = 16
ARC S(ARC) T(ARC)
12 13 14
13 13 15
16 14 16
15 15 16
14 15 14

SUBNETWORK 28 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 16
SINK NODE = 17

ARC S(ARC) T(ARC)
17 16 17

SUBNETWORK 29 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 4
SINK NODE = 5

ARC S(ARC) T(ARC)
25 4 5

SUBNETWORK 30 IS A MINIMUM NETWORK
IT IS COMPOSED OF:

SOURCE NODE = 5
SINK NODE = 6

ARC S(ARC) T(ARC)
26 5 6
SAMPLE PROBLEM: SCHEMATIC REPRESENTATION

BREAKUP

Arcs are designated as follows:

```
5  10  7
```

The arc number is written above or to the left of the arc activity line. The arrowhead indicates the direction of the activity. The circled numbers are node numbers. In this case the arc depicted is arc number 10, flowing from left to right with beginning node 5 and terminating node 7.

Subnetworks are designated as follows:

```
11
12  15
```

The subnetwork number is written above or to the left of the network activity direction line. The arrowhead indicates the direction of the subnetwork activity. The numbers enclosed in ovals are source and sink nodes. In this case the subnetwork depicted is subnetwork 11, flowing from left to right with source node 12 and sink node 15. A solid activity direction line indicates the subnetwork is a minimum network. A broken line indicates the subnetwork must be considered by at least one more subroutine of the BREAKUP program.

```
6  8  17
```

The above figure indicates subnetwork 8, flowing from left to right with source node 6 and sink node 17, is not yet a minimum network.

In the BREAKUP diagrams, the decomposition of the initial network is finished when all subnetworks are minimum networks, e.g. all lines are solid.
INITIAL NETWORK

1

Ψ

18
PROGRAM LISTING

PROGRAM BREAKUP

BREAKUP IDENTIFIES PARALLEL SUBNETWORKS AND SERIES SUBNETWORKS
MORE THAN ONE NETWORK MAY BE DECOMPOSED DURING A RUN.
SEPARATE THE DATA FOR EACH SUBNETWORK BY A BLANK CARD

THE FOLLOWING IS AN ALPHABETICAL LISTING OF THE VARIABLES AND
ARRAYS THAT ARE USED IN THIS MAIN PROGRAM AND ITS SUBROUTINES

ARCS = THE NUMBER OF ARCS IN THE SUBNETWORK
BDNUM(I) = THE BUNDLE NUMBER TO WHICH NODE I IS ASSIGNED
CHECK = ARRAY USED TO STORE ARCS HAVING THE SOURCE AND THE
SINK AS THEIR ONLY NODES
CTNSUB = THE TEMPORARY NUMBER OF SUBNETWORKS FOUND IN THE
PREVIOUS STEP
LNODEN = THE LARGEST NODE NUMBER BEING READ IN
MAXND = THE LARGEST NODE NUMBER THAT HAS ALREADY BEEN ASSIGNED
AT LEAST TEMPORARILY TO A BUNDLE
NARCSS(I) = THE NUMBER OF ARCS IN SUBNETWORK I
NSUB = THE TOTAL OF SUBNETWORKS THUS FAR
NTARC = THE NUMBER OF ARCS IN THIS SUBNETWORK
NUMBD = THE NUMBER OF BUNDLES CREATED
S(I) = THE STARTING NODE FOR ARC I
SINK = THE NODE NUMBER CORRESPONDING TO THE SINK
SINKS(I) = THE SINK IN SUBNETWORK I
SOURC(I) = SOURCE NODE IN SUBNETWORK I
SOURCE = THE NODE NUMBER CORRESPONDING TO THE SOURCE
STEP = STAGE NUMBER
SUBNET(I,J) = THE ITH ARC IN THE JTH SUBNETWORK
SUMARC = THE CURRENT NUMBER OF ARCS IN SUBNETWORK NSUB
T(I) = THE TERMINATING NODE FOR ARC I
TARC = THE SUBNETWORK WITHOUT THE ARCS INVOLVING NODE K
TLNODEN = TEMPORARY LARGEST NODE NUMBER
TNSUB = THE NUMBER OF SUBNETWORKS CREATED IN THE CURRENT STAGE
TSNSUB = THE TEMPORARY SUBNETWORK BEING USED IN STAGE
SUBROUTINE
TSUBN = THE NUMBER OF THE SUBNETWORK CURRENTLY BEING CONSIDERED
TYPESN = THE TYPE OF SUBNETWORK BEING CONSIDERED:
1 = BUNDLE SUBNETWORK  2 = CUT SUBNETWORK

A MINIMUM SUBNETWORK IS ONE THAT IS NOT COMPOSED OF SMALLER
SUBNETWORKS

IMPLICIT INTEGER*2 (A-Z)
COMMON NSUB,TNSUB,S,T,SUBNET,SOURC,SINKS,NARCSS
DIMENSION S(300),T(300),SUBNET(300,100),SOURC(100)
DIMENSION SINKS(100),NARCSS(100)

THIS PROGRAM WILL ACCOMMODATE A NETWORK THAT HAS A MAXIMUM OF
300 NODES, A MAXIMUM OF 300 ARCS AND CAN STORE A MAXIMUM OF
100 SUBNETWORKS IN THE BREAKUP PROCESS
NA = THE NUMBER OF ARCS IN THE NETWORK TO BE CONSIDERED
NS = THE TOTAL NUMBER OF POSSIBLE SUBNETWORKS IN THE BREAKUP
PROCESS
IN THE CASE WHERE NO GOOD ESTIMATE CAN BE MADE OF NS, USE NA
FOR ALL DIMENSIONS

THE ARRAY DIMENSIONS ARE:

S(NA),T(NA),SUBET(NA,NS),SOURC(NS),SINKS(NS),NARCSS(NS)

IN SUBROUTINE BUNDLE THE ARRAY DIMENSIONS ARE:

BDNUM(NS),CHECK(NS)

IN SUBROUTINE CUT THE ARRAY DIMENSIONS ARE:

ORIGIN(NS),POST(NS),TARC(NS),RCUT(NA)

READ THE NETWORK IN

CALL NETIN (LNODEN)

STEP=1

TYPESN=1

TSUBN=1

CALL BUNDLE (LNODEN,TSUBN)

PRINT OUT THIS STAGE OF THE BREAKUP

CALL STAGE (TYPESN,TSUBN)

CTNSUB=TNDSUB

GO TO 70

FIND THE NUMBER OF NEWLY CREATED SUBNETWORKS

CTNSUB=0

TSUBN=NSUB-LU0P

TYPESN=1

DO 20 I=1,LOOP

FIND THE NEXT SUBNETWORK TO BE FURTHER SUBDIVIDED

TSUBN=TSUBN+1

FIND THE LARGEST NODE NUMBER IN SUBNETWORK TSUBN

CALL NODER (TLNODEN,TSUBN)

FIND THE BUNDLE SUBNETWORKS

CALL BUNDLE (TLNODEN,TSUBN)

PRINT OUT THIS STAGE OF THE BREAKUP

CALL STAGE (TYPESN,TSUBN)

IF THERE IS ONLY ONE BUNDLE FOUND IN SUBNETWORK TSUBN, WE ARE

FINISHED. PRINT OUT ITS COMPONENT ARCS

IF (TNDSUB.FQ.1) CALL ENDNET (TSUBN)

COUNT THE NEW NUMBER OF SUBNETWORKS CREATED

CTNSUB=TNDSUB+CTNSUB

IF ALL SUBNETWORKS ARE IN THEIR SMALLEST FORM, WE ARE FINISHED

IF (CTNSUB.EQ.0) GO TO 90

GO TO 70
FIND THE NUMBER OF NEWLY CREATED SUBNETWORKS

DO 30 I=1,LOOP

FIND THE NEXT SUBNETWORK TO BE FURTHER SUBDIVIDED

IF (TSUBN.EQ.1) GO TO 55

FIND THE LARGEST NODE NUMBER IN SUBNETWORK TSUBN

CALL NODER (TLNDEN,TSUBN)

PRINT OUT THIS STAGE OF THE BREAKUP

CALL STAGE (TYPESN,TSUBN)

IF THERE ARE NO CUTS FOUND IN SUBNETWORK TSUBN, WE ARE FINISHED

PRINT OUT ITS COMPONENT ARCS

CALL ENDSNT (TSUBN)

COUNT THE NEW NUMBER OF SUBNETWORKS CREATED

CTNSUB=CTNSUB+CTNSUB

IF ALL SUBNETWORKS ARE IN THEIR SMALLEST FORM, WE ARE FINISHED

IF (CTNSUB.EQ.0) GO TO 90

GO TO 70

WRITE (6,900) STEP

FORMAT (1H1,5X,'STAGE',13,' BREAKUP')

LET'S GO BACK TO THE APPROPRIATE LOOP FOR THE NEXT STAGE

GO TO (80,85),TYPESN

CONTINUE

READ (5,100,END=666)

FORMAT (13)

GO TO 6000

WRITE (6,9000)

RETURN

END

SUBROUTINE NETIN (LNODEN)
IMPLICIT INTEGER*2 (A-Z)
COMMON NSUB, TNSUB, S, T, SUBNET, SOURC, SINKS, NARCSS
DIMENSION S(300), T(300), SUBNET(300,100), SOURC(100)
DIMENSION SINKS(100), NARCSS(100)
C
ZEROIZE SOURC ARRAY
C
DO 20 I=1,50
20 SOURC(I)=0
C
READ IN THE INITIAL NETWORK LIMITS
C
READ (5,100) ARCS, SOURCE, SINK, LNODEN
100 FORMAT (4(I3))
WRITE (6,200)
200 FORMAT (1H1,5X,'INPUT STAGE')
WRITE (6,210) ARCS, SOURCE, SINK, LNODEN
210 FORMAT (1H0,/6X, 'THE INITIAL NETWORK HAS',12X,14,' ARCS',//6X,
* 'THE SOURCE IS NODE NUMBER',11X,13, //6X, 'THE SINK IS NODE NUMBER',
*13X,13, //6X, 'THE LARGEST NODE IS NODE NUMBER',5X,I3)
WRITE (6,220)
220 FORMAT (1H0,5X, 'THE INITIAL NETWORK AS READ IN IS:', //6X,
* 'ARC NUMBER',5X,'ORIGIN NODE',5X,'TERMINAL NODE')
C
READ IN EACH ARC AND ITS STARTING AND TERMINATING NODES
C
THE ARCS AND NODES MAY BE NUMBERED ANY WAY AND READ IN IN ANY
C
ORDER
C
I = ARC NUMBER
C
S = THE NODE NUMBER FOR THE START OF AN ARC
C
T = THE TERMINAL NODE OF AN ARC
C
DO 10 J=1,ARCS
READ (5,100) I,S(I),T(I)
WRITE (6,240) I,S(I),T(I)
240 FORMAT (1H8X,13I3,13X,13I3,14X,13)
C
CREATE THE FIRST SUBNETWORK
C
10 SUBNET (J,1)=I
SOURC(I)=SOURCE
SINKS(I)=SINK
NARCSS(I)=ARCS
NSUB=1
RETURN
END

SUBROUTINE NOODE (TLNODEN, TSUBN)
C
FINDS LARGEST NODE NUMBER IN THE SUBNETWORK TSUBN
C
IMPLICIT INTEGER*2 (A-Z)
COMMON NSUB, TNSUB, S, T, SUBNET, SOURC, SINKS, NARCSS
DIMENSION S(300), T(300), SUBNET(300,100), SOURC(100)
DIMENSION SINKS(100), NARCSS(100)
ARCS=NARCSS(TSUBN)
TLNODEN=0
DO 20 J=1,ARCS
A=SUBNET(J, TSUBN)
M=S(A)
N=T(A)
20 CONTINUE
IF (M > N) THEN
...
MAXND=N
IF (M.GT.N) MAXND=M
IF (MAXND.GT.TLNDEN) TLNDEN=MAXND
20 CONTINUE
RETURN
END

SUBROUTINE ENOSNT (TSUBN)
C PRINTS SMALLEST BREAKDOWN OF SUBNETWORK TSUBN
C
IMPLICIT INTEGER*2 (A-Z)
COMMON NSUB,TNSUB,S,T,SUBNET,SOURC,SINKS,NARCSS
DIMENSION S(300),T(300),SUBNET(300,100),SOURC(100)
DIMENSION SINKS(100),NARCSS(100)
SOURCE=SOURC(TSUBN)
SINK=SINKS(TSUBN)
WRITE (6,100) TSUBN
100 FORMAT (1HO,6X,'SUBNETWORK ',I3,' IS A MINIMUM NETWORK',/16X, 
* 'IT IS COMPOSED OF:'))
WRITE (6,200) SOURCE,SINK
200 FORMAT (1HO,19X,'SOURCE NODE = ',I3,'//20X,'SINK NODE = ',I3)
M=NARCSS(TSUBN)
WRITE (6,400)
400 FORMAT (1HO,19X,'ARC',2X,'S(ARC) ',2X,'T(ARC)')
DO 11 M=1,NSUB
11 WRITE (6,300) I,S(I),T(I)
300 FORMAT (1H0,19X,I3,3X,I3,5X,I3)
C SET TNSUB=0 SO THAT THE REMAINING NUMBER OF SUBNETWORKS DOESN'T 
INCLUDE THIS MINIMUM SUBNETWORK
C
TNSUB=0
RETURN
END

SUBROUTINE STAGE (TYPESN,TSUBN)
C PRINTS OUT THE CURRENT STAGE OF BREAKUP
C
IMPLICIT INTEGER*2 (A-Z)
COMMON NSUB,TNSUB,S,T,SUBNET,SOURC,SINKS,NARCSS
DIMENSION S(300),T(300),SUBNET(300,100),SOURC(100)
DIMENSION SINKS(100),NARCSS(100)
C TNSUB=THE NUMBER OF NEW SUBNETWORKS RESULTING FROM THE BREAKUP 
IN THIS STAGE
C TNSUB=1 IMPLIES NO BREAKUP OCCURRED IN THIS STAGE
C
IF (TNSUB.EQ.1) GO TO 600
WRITE (6,100) TSUBN
100 FORMAT (1HO,6X,'SUBNETWORK ',I3,' IS COMPOSED OF SUBNETWORKS: ') 
TNSUB=NSUB-TNSUB
M=TNSUB+1
WRITE (6,300) I,1=M,NSUB
300 FORMAT (1HO,5X,20(I3,','))
IF (TYPESN.EQ.1) GO TO 60
WRITE (6,400)
COMMON NSUB, TNSUB, S, T, SUBNET, SOURC, SINKS, NARCSS
DIMENSION S(300), T(300), SUBNET(300,100), SOURC(100)
DIMENSION SINKS(100), NARCSS(100), BDNUM(300), CHECK(300)

GROUP NODES INTO BUNDLES

SOURCE=SOURC(TSUBN)
SINK=SINKS(TSUBN)
NUMBD=1

ZEROIZE THE BDNUM ARRAY

DO 10 I=1, LN0DEN
10 BDNUM(I)=0
K=SUBNET(I, TSUBN)
M=S(K)
N=T(K)
BDNUM(M)=1
BDNUM(N)=1
MAXND=N
IF (M.GT. N) MAXND=M
ARCS=NARCSS(TSUBN)
IF (ARCS.EQ.1) GO TO 515
DO 1 K=2, ARCS
I=SUBNET(K, TSUBN)
BDNUM(SOURCE)=0
BDNUM(SINK)=0
M=S(I)
N=T(I)
IF(M.GT.MAXND) MAXND=M
IF(N.GT.MAXND) MAXND=N

IF AT LEAST 1 NODE ON THE ARC HAS NOT BEEN ASSIGNED TO A BUNDLE
GO TO 2

IF(BDNUM(M).EQ.0) GO TO 2

IF ONLY THE TERMINAL NODE ON THE ARC HAS NOT BEEN ASSIGNED TO A BUNDLE, GO TO 3

IF(BDNUM(N).EQ.0) GO TO 3

IF BOTH NODES ON THE ARC HAVE BEEN ASSIGNED TO THE SAME BUNDLE
EVERYTHING IS OKAY, GO TRY ANOTHER ARC

IF(BDNUM(N).EQ.BDNUM(M)) GO TO 1
IF THE NODES ON THE ARC ARE ASSIGNED TO DIFFERENT BUNDLES, THEN THESE TWO BUNDLES SHOULD BE POOLED

IF(BDNUM(N).LT.BDNUM(M)) GO TO 6

POOL BUNDLES
THE BUNDLE WITH THE LARGER BUNDLE NUMBER IS POOLED INTO THE BUNDLE WITH THE SMALLER BUNDLE NUMBER
THE BUNDLE NUMBERS OF ALL BUNDLES ARE ALL ADJUSTED

MAXBD=BDNUM(N)
MINBD=BDNUM(M)
GO TO 7
6 MAXBD=BDNUM(M)
MINBD=BDNUM(N)
7 DO 5 J=1,MAXNO
B=BDNUM(J)
IF (B.EQ.MAXBD) BDNUM(J)=MINBD
IF (B.GT.MAXBD) BDNUM(J)=BDNUM(J)-1
5 CONTINUE
NUMB0=NUMBD
GO TO 1

IF BOTH NODES ON THE ARC ARE UNASSIGNED, GO TO 4 WHERE A NEW BUNDLE IS CREATED

2 IF(BDNUM(N).EQ.0) GO TO 4

ASSIGN THE ORIGIN NODE OF THE ARC TO THE BUNDLE CONTAINING THE TERMINAL NODE
BDNUM(M)=BDNUM(N)
GO TO 1

ASSIGN THE TERMINAL NODE OF THE ARC TO THE BUNDLE CONTAINING THE ORIGIN NODE OF THE ARC
3 BDNUM(N)=BDNUM(M)
GO TO 1

CREATE A NEW BUNDLE
4 NUMBD=NUMBD+1
BDNUM(M)=NUMBD
BDNUM(N)=NUMBD
1 CONTINUE
515 CONTINUE
BDNUM(SINK)=0

IF WE ONLY HAVE 1 BUNDLE FROM THE SUBNETWORK, WE ARE FINISHED

IF (NUMBD.EQ.1) GO TO 219

ZEROIZE CHECK ARRAY

DO 290 I=1,ARCS
 CHECK(I)=0
290 L=0

THE NODES ARE IN BUNDLES. PUT THE ASSOCIATED ARCS INTO
APPROPRIATE PARALLEL SUBNETWORKS

DO 33 I=1,NUMBD
    SUMARC=0
    NSUB=NSUB+1
    DO 34 K=1,ARCS
        M=SUBNET(K,TSUBN)
        N=S(M)
    S Remarks
SOURCE AND SINK HAVE BUNDLE NUMBER 0
IF (N.EQ.SOURCE) N=T(M)
IF (BDNUM(N).EQ.1) GO TO 239
IF (N.EQ.SINK) GO TO 229
GO TO 34

SPECIAL CASE: BUNDLE HAS ONLY 2 NODES: SOURCE, SINK.
PUT ALL ARCS THAT ARE PARALLEL SUBNETWORKS BY THEMSELVES INTO
THE CHECK ARRAY
229 DO 291 J=1,K
    W=CHECK(J)
    IF (W.EQ.0) GO TO 292
    IF (M.EQ.W) GO TO 34
291 CONTINUE
292 CHECK(J)=M
GO TO 34

THIS ARC IS IN THE BUNDLE I, HENCE IT IS IN THE ITH NEW
SUBNETWORK
239 SUMARC=SUMARC+1
    SUBNET(SUMARC,NSUB)=M
34 CONTINUE

CREATE NEW SUBNETWORKS

IF THIS BUNDLE HAS NO NODES, PUT AN ARC SUBNETWORK INTO
SUBNET (I,NSUB)
343 IF (SUMARC.EQ.0) GO TO 333

NARCSS(NSUB)=SUMARC
SOURC(NSUB)=SOURCE
SINKS(NSUB)=SINK
GO TO 33

STORE THE SUBNETWORKS THAT HAVE ONLY SOURCE AND SINK NODES
333 SUMARC=1
L=L+1
SUBNET(L,NSUB)=CHECK(L)
GO TO 343
33 CONTINUE
219 TNSUB=NUMBD
RETURN
END

SUBROUTINE CUT (TSUBN,LNODEN)
CUT IDENTIFIES CUT NODES EXCLUDING THE DESIGNATED SOURCE AND SINK.

CUT ALSO IDENTIFIES THE CUT GROUPS; THAT IS, THE SUBNETWORKS WHICH ARE-IN SERIES AND CONNECTED BY THE CUT NODES.

IMPLICIT INTEGER*2 (A-Z)
COMMON NNSUB,NSURC,S,T,SUBNET,SOURC,SINKS,NARCSS
DIMENSION S(300),T(300),SUBNET(300,100),SOURC(100),TARC(300)
DIMENSION SINKS(100),NARCSS(100),ORIGIN(300),POST(300),RCUT(100)

FIND THE CUT NODES
NCUT IS THE NUMBER OF CUT NODES FOUND THUS FAR.

NCUT=0
ARCS=NARCSS(TSUBN)
SOURCE=SOURC(TSUBN)
SINK=SINKS(TSUBN)

THE DO LOOP DOWN TO STATEMENT NUMBER 1 DETERMINES THE CUT NODES.

DO 1 K=1, LNODEN

CHECK TO SEE IF NODE K IS ACTUALLY IN THE SUBNETWORK.

DO 20 J=1,ARCS
Z=SUBNET(J,TSUBN)
IF(S(Z).EQ.K) GO TO 21
IF(T(Z).EQ.K) GO TO 21
20 CONTINUE

NODE K IS NOT IN THIS SUBNETWORK

GO TO 1

21 CONTINUE

NODE K IS IN THIS SUBNETWORK.

IF(K.EQ.SOURCE) GO TO 1
IF(K.EQ.SINK) GO TO 1
NTARC=0

DO 2 J=1,ARCS
Z=SUBNET(J,TSUBN)
IF(S(Z).EQ.K) GO TO 2
IF(T(Z).EQ.K) GO TO 2
NTARC=NTARC+1
TARC(NTARC)=SUBNET(J,TSUBN)
2 CONTINUE

TARC IS THE SUBNETWORK WITHOUT THE ARCS INVOLVING NODE K.

IF TARC CONTAINS A PATH FROM THE SOURCE TO THE SINK, THEN NODE K IS NOT A CUT NODE.

OTHERWISE, K IS A CUT NODE.

ORIGIN(1)=SOURCE
NORIG=1
11 CONTINUE
NPOST=0

IF THERE ARE NO ARCS IN THE TARC ARRAY, K IS A CUT NODE.
C IF (NTARC.EQ.0) GO TO 44
C FIND ALL NODES WHICH COME AFTER AN ORIGIN; PUT THEM IN POST
DO 4 I=1,NORIG
DO 5 J=1,NTARC
Y=ORIGIN(I)
Z=TARC(J)
U=S(Z)
V=T(Z)
 IF (U.NE.Y) GO TO 5
 IF (V.EQ.SINK) GO TO 1
C
C IF WE'VE REACHED THE SINK, NODE K IS NOT A CUT NODE
C
IF(NPOST.GE.1) GO TO 8
NPOST=NPOST+1
POST(NPOST)=V
GO TO 5
8 DO 9 L=1,NPOST
C IF THIS TERMINAL NODE IS ALREADY A POST, LET'S IGNORE IT
C
IF (POST(L).EQ.V) GO TO 5
CONTINUE
NPOST=NPOST+1
POST(NPOST)=V
CONTINUE
4 CONTINUE
C IF THERE ARE NOW NO POSTS, NODE K IS A CUT NODE
C
44 IF(NPOST.NE.0) GO TO 13
NCUT=NCUT+1
RCUT(NCUT)=K
GO TO 1
13 NORIG=NPOST
DO 14 L=1,NORIG
C THESE ARE NOW OUR NEW ORIGINS
C
14 ORIGIN(L)=POST(L)
C CHECK THE NEW ORIGINS FOR THEIR POSTS
C
GO TO 11
1 CONTINUE
IF (NCUT.EQ.0) GO TO 32
NSUB=NSUB+1
SOURCE(NSUB)=SOURCE
33 ORIGIN(1)=SOURCE
C NOW WE NEED TO FIND THE COMPONENTS OF THE SERIES SUBNETWORKS
C THAT ARE SEPARATED BY THE CUT NODES
C
39 SUMARC=0
NORIG=1
23 NPOST=0
C
C ZEROIZE POST ARRAY
C
DO 55 I=1,LNODEN
55 POST(I)=0
DO 24 I=1,NORIG
Y=ORIGIN(I)
DO 25 J=1,ARCS
Z=SUBNET(J,TSUBN)
C
ALL ARCS BEGINNING AT THIS ORIGIN GO INTO THE NEW SUBNETWORK
C
IF (S(Z).NE.Y) GO TO 25
SUMARC=SUMARC+1
SUBNET (SUMARC,NSUB)=Z
C
T(Z) WILL BE A NEW ORIGIN IF IT ISN'T A REPEAT OF A CURRENT ORIGIN
C
CHECK TO SEE IF IT IS A REPEAT
C
DO 30 K=1,NORIG
X=T(Z)
C
IF T(Z) IS A REPEAT OF A CURRENT ORIGIN, LET'S IGNORE IT
C
IF(X.EQ.ORIGIN(K)) GO TO 25
30 CONTINUE
NPOST=NPOST+1
POST(NPOST)=X
C
CHECK TO SEE IF ANY POST IS A CUT NOOE
IF IT IS, REPLACE IT WITH A ZERO
C
DO 52 W=1,NCUT
0=RCUT(W)
IF (X.NE.0) GO TO 52
POST(NPOST)=0
NPOST=NPOST-1
C
THIS CUT NOOE IS THE SINK OF THE SUBNETWORK UNDER CONSIDERATION AND THE SOURCE OF THE NEXT SUBNETWORK TO BE CONSIDERED.
C
SINKS(NSUB)=0
SOURCE(NSUB+1)=0
C
CHECK TO SEE IF ANY POSTS ARE REPEATED IN THE POST ARRAY. IF THEY ARE, REDUCE THE NUMBER OF POSTS TO WHERE THERE ARE NO REPEATS
C
52 CONTINUE
C
IF WE HAVE ONE OR LESS POSTS, THERE ARE NO ADJUSTMENTS OF THE POST ARRAY TO BE MADE; LET'S CONTINUE
C
53 IF (NPOST.LE.1) GO TO 25
POSTCK=NPOST-1
DO 7 K=1,POSTCK
IF (POST(K).EQ.POST(NPOST)) GO TO 61
7 CONTINUE
GO TO 25

61 POST(NPOST) = 0
NPOST = NPOST - 1
25 CONTINUE
24 CONTINUE
C
C IF WE HAVE NO POSTS LEFT, WE HAVE FOUND ALL OF THIS SUBNETWORK
C
C IF (NPOST.EQ.0) GO TO 34
NORIG = NPOST
DO 28 L = 1, NORIG
28 ORIGIN(L) = POST(L)
GO TO 23
34 NARCSS(NSUB) = SUMARC
NSUB = NSUB + 1
X = SOURC(NSUB)
C
C IF THE SOURCE OF NSUB IS NOT A CUT NODE, WE NEED TO ADJUST NSUB
C AND GO BACK TO THE MAIN PROGRAM FOR THE NEXT STAGE OF THE
C BREAKUP
C
C IF (X.EQ.0) GO TO 31
ORIGIN(1) = X
GO TO 39
31 NSUB = NSUB - 1
SINKS(NSUB) = SINK
32 TNSUB = NCUT + 1
RETURN
END
LOOP

This short program will determine whether a given network is acyclic or contains loops (cycles). The program examines each node and indicates whether or not the node is part of a loop. If a node is part of a loop, the number of activities in the loop is also indicated.

The basic steps in determining whether or not the INODE-th node is part of a loop are as follows:

1. Identify all activities whose terminal node is the INODE-th node. Let A be the set of all origin nodes for these activities.
2. If INODE is in A, the INODE-th node is part of a loop and stop.
3. Identify all activities whose terminal node is in A. Redefine A to be the set of origin nodes for these activities. If A is now empty, stop and the INODE-th node is not part of a loop. If A is not empty, return to step 2.

Specific Input Instructions:

Card 1: Col. 1-3: The number of activities in the network, Format I3). Col. 4-6: The largest node number in the network, Format (I3).

For each activity one card with:

Col. 1-3: The activity’s origin node number, Format (I3).
Col. 4-6: The activity’s terminal node number, Format (I3).

The activities may be inputted in any order. The nodes may be numbered in any manner; however, the program is more efficient if the set of node numbers contains only the numbers 1 through N where N is the number of nodes in the network.
Dimension Restrictions:

This program is written in FORTRAN G. The current dimensions will allow a network with 300 activities and 200 nodes to be considered.
SAMPLE NETWORK

![Network Diagram]

SAMPLE INPUT

6 5
1 2
1 3
4 3
3 2
2 4
4 5

SAMPLE OUTPUT

<table>
<thead>
<tr>
<th>Activity</th>
<th>Tail</th>
<th>Head</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Node 1 is not looped
Node 2 is looped. There are 3 activities in the loop.
Node 3 is looped. There are 3 activities in the loop.
Node 4 is looped. There are 3 activities in the loop.
PROGRAM LOOP

This program determines whether the inputted network is acyclic or contains nodes which are part of loops (cycles).

IMPLICIT INTEGER*2 (A-Z)
DIMENSION A(200), B(200), HEAD(300), TAIL(300)

The array dimensions are: A(N), B(N), HEAD(M), TAIL(M)
WHERE
N = the number of nodes in the network
M = the number of activities in the network

ILOOP = 0
READ (5,100) M, N
100 FORMAT (2I3)
READ (5,101) (TAIL(I), HEAD(I), I = 1, M)
101 FORMAT (2I3)
WRITE (6, 2001)
2001 FORMAT (IH1)  
WRITE (6, 2000) (I, TAIL(I), HEAD(I), I = 1, M)
2000 FORMAT (ACTIVITY TAIL HEAD, (4X, I3, 7X, I3, 6X, I3))
WRITE (6, 2001)

Form the 1st hierarchy
INODE = 0
80 HIER = 2
INODE = INODE + 1
J = 0
DO 1 I = 1, M
IF (HEAD(I) .NE. INODE) GO TO 1
J = J + 1
A(J) = TAIL(I)
IF (TAIL(I) .EQ. INODE) GO TO 998
1 CONTINUE
IF (J .EQ. 0) GO TO 997
IA = J
J = 0

Form the subsequent hierarchies
10 CONTINUE
DO 2 II = 1, IA
DO 3 I = 1, M
IF (HEAD(I) .NE. A(II)) GO TO 2
IF (TAIL(I) .EQ. INODE) GO TO 998
IF (J .EQ. 0) GO TO 40
DO 10 K = 1, J
IF (TAIL(I) .EQ. B(K)) GO TO 11
10 CONTINUE
40 CONTINUE
J = J + 1
B(J) = TAIL(I)
11 CONTINUE
3 CONTINUE
2 CONTINUE
IF (J .EQ. 0) GO TO 997
HIER = HIER + 1
IA = J
J = 0
DO 20 I = 1, IA
20 A(I) = B(I)
GO TO 102
997 CONTINUE
   WRITE(6,2002) INODE
   2002 FORMAT(' NODE',I5,' IS NOT LOOPED')
   IF (INODE.NE.NMM ) GO TO 80
   IF(ILOOP.EQ.1) GO TO 50
   WRITE(6,1000)
   1000 FORMAT(' THERE ARE NO LOOPS IN THIS NETWORK')
   GO TO 999
50 WRITE(6,51)
   51 FORMAT(' THERE ARE NO OTHER LOOPS IN THIS NETWORK')
   GO TO 999
998 WRITE (6,1001) INODE,HIER
   1001 FORMAT(' NODE ',I5,' IS LOOPED. THERE ARE ',I3,' ACTIVITIES IN THE LOOP')
   ILOOP=1
   INDD=INODE+1
   IF(INDD.NE.NMM) GO TO 80
999 CONTINUE
   WRITE(6,2001)
   STOP
END
STATISTICAL PERT: DECOMPOSING A PROJECT NETWORK

Progress Report

R. L. Sielken, Jr.  N. E. Fisher

Nov 1975
NG0014-68-A-0148
NR047-700

This document has been approved for public release and sale; its distribution is unlimited.

See page following Table of Contents

THEMIS: Optimization research program.
<table>
<thead>
<tr>
<th>KEY WORDS</th>
<th>LINK A</th>
<th>LINK B</th>
<th>LINK C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ROLE</td>
<td>WT</td>
<td>ROLE</td>
</tr>
<tr>
<td>Statistical PERT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project Scheduling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network Decomposition</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ATTACHMENT III (Continued)
BASIC DISTRIBUTION LIST FOR
UNCLASSIFIED TECHNICAL REPORTS

Operations Research
Office of Naval Research
(Code 434)
Arlington, Virginia 22217
3

Director, Naval Research Laboratory
Attn: Library, Code 2029 (ONRL)
Washington, D.C. 20390
2

Defense Documentation Center
Cameron Station
Alexandria, Virginia 22314
12

Defense Logistics Studies
Information Exchange
Army Logistics Management Center
Fort Lee, Virginia 23801
1

Technical Information Center
Naval Research Laboratory
Washington, D.C. 20390
6

Office of Naval Research
New York Area Office
207 West 24th Street
New York, New York 10011
Attn: Dr. J. Laderman
1

Director, Office of Naval Research
Branch Office
1030 East Green Street
Pasadena, California 91101
Attn: Dr. A.R. Laufer
1

Lt. Dennis Johnston - ED3
Manned Spacecraft Center
NASA
Houston, Texas 77058
1

Mr. William Dejka
Code 360
Naval Electronics Laboratory Center
San Diego, California 92132
1

Professor Lucien A. Schmit, Jr.
Room 6731, Boelter Hall
School of Engrng. & Applied Sci.
University of California
Los Angeles, California 90024
1

Director
Office of Naval Research
Branch Office
495 Summer Street
Boston, Massachusetts 02210
Attn: Dr. A.L. Powell
1

Director
Office of Naval Research
Branch Office
219 South Dearborn Street
Chicago, Illinois 60604
Attn: Dr. A.R. Dawe
1

Office of Naval Research
San Francisco Area Office
760 Market St. - Room 447
San Francisco, California 94103
1

Technical Library
Naval Ordnance Station
Indian Head, Maryland 20640
1

Bureau of Naval Personnel
Navy Department
Technical Library
Washington, D.C. 20370
STOP 82
2

Library, Code 0212
Naval Postgraduate School
Monterey, California 93940
1

Naval Ordnance Station
Louisville, Kentucky 40214
1

Library
Naval Electronics Laboratory Center
San Diego, California 92152
1

Naval Ship Engineering Center
Philadelphia
Division Technical Library
Philadelphia, Pennsylvania 19112
1

Dr. A.L. Slafkosky
Scientific Advisor
Commandant Marine Corps (Code AX)
Washington, D.C. 20380
1
Purdue University
Graduate School of Ind. Admin.
Lafayette, Indiana 47907
Attn: Prof. Andrew Whinston

Department of Economics
Tisch Hall - 5th Floor
Washington Square, NY Univ.
New York City, NY 10003
Attn: Prof. O. Morgenstern

Dept. of Statistics
Syntex Research
3401 Hillview
Palo Alto, CA 94304
Attn: Prof. Stuart Bessler

University of California
Department of Engineering
Los Angeles, California 90024
Attn: Prof. R.R. O'Neill

Maritime Transportation Research Board
National Academy of Sciences
2101 Constitution Avenue
Washington, D.C. 20418
Attn: RADM J.B. Oren USCG (RET)

Stanford University
Department of Operations Research
Stanford, California 94305
Attn: Prof. A.F. Veinott, Jr.

Texas A&M Foundation
College Station, Texas 77843
Attn: Prof. H.O. Hartley

Case Western Reserve University
Cleveland, Ohio 44106
Attn: Prof. B.V. Dean

University of California
Center for Research in
Management Science
Berkeley, California 94720
Attn: Prof. W.I. Zangvell

U.S. Naval Postgraduate School
Department of Operations
Research and Economics
Monterey, California 93940
Attn: Prof. C.R. Jones

Dr. Del Gilmer
Naval Weapons Center (Code 607)
China Lake, California 93555

Dr. Paul Murrill
Project Themis
Department of Chemical Engineering
Louisiana State University
Baton Rouge, Louisiana 70803

Director
Army Materials & Mechanics Research Center
Attn: Mr. J. Bluhm
Watertown, Massachusetts 02172

Commanding Officer
U.S. Army Ballistic Research Laboratories
Attn: Dr. John H. Giese
Aberdeen-Proving Ground, Maryland 21005

University of Iowa
Department of Mechanics and Hydraulics
Iowa City, Iowa 52240
Attn: Prof. W.F. Ames

Office of Naval Research
Resident Representative
1105 Guadalupe
Lowich Building
Austin, Texas 78701
Attn: Mr. Francis M. Lucas

Dr. Jerome Bracken
Institute of Defense Analyses
400 Army-Navy Drive
Arlington, Virginia 22202

Professor Richard L. Fox
School of Engineering
Case Western Reserve University
Cleveland, Ohio 44106

Director, National Security Agency
Ft. George G. Meade, Maryland 20755
Attn: Dr. Joseph Blum, R44
<table>
<thead>
<tr>
<th>Naval Applied Science Laboratory</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Library, Code 222</td>
<td></td>
</tr>
<tr>
<td>Flushing &amp; Washington Avenues</td>
<td></td>
</tr>
<tr>
<td>Brooklyn, New York 11251</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Naval Undersea Warfare Center</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>3202 E. Foothill Boulevard</td>
<td></td>
</tr>
<tr>
<td>Pasadena, California 91107</td>
<td></td>
</tr>
<tr>
<td>Technical Library</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University of Chicago</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statistical Research Center</td>
<td></td>
</tr>
<tr>
<td>Chicago, Illinois 60637</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. W. Kruskal</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stanford University</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department of Statistics</td>
<td></td>
</tr>
<tr>
<td>Stanford, California 94305</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. G.J. Lieberman</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Florida State University</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department of Statistics</td>
<td></td>
</tr>
<tr>
<td>Tallahassee, Florida 32306</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. I.R. Savage</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Princeton University</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department of Mathematics</td>
<td></td>
</tr>
<tr>
<td>Princeton, New Jersey 08540</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. J.W. Tukey</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stanford University</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department of Statistics</td>
<td></td>
</tr>
<tr>
<td>Stanford, California 94305</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. T.W. Anderson</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University of California</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department of Statistics</td>
<td></td>
</tr>
<tr>
<td>Berkeley, California 94720</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. P.J. Bickel</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University of Washington</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department of Mathematics</td>
<td></td>
</tr>
<tr>
<td>Seattle, Washington 98105</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. Z.W. Birnbaum</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cornell University</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department of Mathematics</td>
<td></td>
</tr>
<tr>
<td>White Hall</td>
<td></td>
</tr>
<tr>
<td>Ithaca, New York 14850</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. J. Kiefer</td>
<td>1</td>
</tr>
<tr>
<td>Logistics Research Project</td>
<td>Copies</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>The George Washington University</td>
<td>1</td>
</tr>
<tr>
<td>707 - 22nd Street, N.W.</td>
<td></td>
</tr>
<tr>
<td>Washington, D. C. 20037</td>
<td></td>
</tr>
<tr>
<td>Attn: Dr. W. H. Marlow</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>International Business Machines Corporation</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>P.O. Box 218, Lambe Estate</td>
<td>1</td>
</tr>
<tr>
<td>Yorktown Heights, New York 10598</td>
<td></td>
</tr>
<tr>
<td>Attn: Dr. Alan Hoffman</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University of California Management Sciences Research Project</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Los Angeles, California 90024</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Dr. J.R. Jackson</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Harvard University Department of Economics</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cambridge, Massachusetts 02138</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. K.J. Arrow</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cowles Commission for Research in Economics</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yale University</td>
<td>1</td>
</tr>
<tr>
<td>New Haven, Connecticut 06520</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. Martin Shubik</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Carnegie-Mellon University Graduate School of Industrial Administration</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pittsburgh, Pennsylvania 15213</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. G. Thompson</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University of California Department of Economics</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berkeley, California 94720</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. R. Radner</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University of California Operations Research Center Institute of Engineering Research</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berkeley, California 94720</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. D. Gale</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University of California Graduate School of Business Administration</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Los Angeles, California 90024</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. J. Marschak</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>The University of Michigan Department of Mathematics, W.E.</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ann Arbor, Michigan 48104</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. R.M. Thrall</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Princeton University Department of Mathematics</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Princeton, New Jersey 08540</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. A.W. Tucker</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case Western Reserve University Systems Research Center</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cleveland, Ohio 44106</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. M. Mesarovic</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University of Texas Department of Mathematics</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Austin, Texas 78712</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Dr. A. Charnes</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stanford University Department of Operations Research</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stanford, California 94305</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. F.S. Hillier</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dr. Claude-Alain Burdet</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asst. Prof. Industrial Admin.</td>
<td>1</td>
</tr>
<tr>
<td>Carnegie-Mellon University</td>
<td></td>
</tr>
<tr>
<td>Pittsburgh, Pennsylvania 15213</td>
<td></td>
</tr>
<tr>
<td>Attn: Prof. R. Remage, Jr.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stanford University Department of Operations Research</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stanford, California 94305</td>
<td>1</td>
</tr>
<tr>
<td>Attn: Prof. F.S. Hillier</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chief of Naval Research (Code 436) Department of the Navy</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arlington, Va. 22217</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Science Librarian Kresge Library Oakland University Rochester, Michigan 48063</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Office of Gifts &amp; Exchanges Library Texas A&amp;M University College Station, Texas 77843</td>
<td>Copies</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

| Archives Texas A&M University College Station, TX 77843 | 1       |