I29A
424
cy 3

UILU-ENG-75-2030

CIVIL ENGINEERING STUDIES

STRUCTURAL RESEARCH SERIES NO. 424

TECHNOLOGY FOR THE FORMULATION AND EXPRESSION OF SPECIFICATIONS

VOLUME II: PROGRAM USER'S MANUAL

Metz Reference Room
Civil Engineering Department
B106 C. E. Building
University of Illinois
Urbana, Illinois 61801

By

J. R. HARRIS

J. W. MELIN

C. ALBARRAN

A Report on a Research Project
Sponsored by
THE NATIONAL BUREAU OF STANDARDS
Contract 5-35844

UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN
URBANA, ILLINOIS
DECEMBER 1975

TECHNOLOGY FOR THE FORMULATION AND EXPRESSION OF SPECIFICATIONS

VOLUME II: PROGRAM USER'S MANUAL

by

J.R. Harris, J.W. Melin, and C. Albarran

A report on a research project sponsored by:

THE NATIONAL BUREAU OF STANDARDS

Contract No. 5-35844

DEPARTMENT OF CIVIL ENGINEERING

UNIVERSITY OF ILLINOIS

URBANA, ILLINOIS

December 1975

INTRODUCTION

This report describes the use of three computer programs (DECISION TABLE, NETWORK, AND OUTLINE) produced as a part of the study, "Technology for the Formulation and Expression of Specifications." The programs are designed for interactive use from a remote console, and the communication with the programs is in free format, making the programs easy to use for those with limited experience in working with computers. The programs are operational on the Burroughs B6700 computer at the Civil Engineering Systems Laboratory (CESL) at the Department of Civil Engineering, University of Illinois, Urbana, Illinois.

The principles on which the programs are based and examples of the application of these principles are contained in volume I of this report while a more technical description of the three programs, including logic diagrams, data structure, and program listings, is contained in volume III. This volume contains a description of the use of each program preceded by a chapter describing access to and use of the computer facility at which the programs are operational.

PROGRAM USER'S MANUAL

TABLE OF CONTENTS

PROGRAM USERS MANUAL

LIST OF FIGURES AND TABLES

Chapter One

GENERAL INFORMATION

1.1 Computer Facility

The facility at which the computer programs are operational is the Civil Engineering Systems Laboratory (CESL), located in the Department of Civil Engineering, University of Illinois, Urbana, Illinois. CESL operates a Burroughs B6700 computer on a time sharing basis for use from remote terminals. The programs described in this manual were written in FORTRAN IV and are operational on the B6700 installed at CESL.

1.2 Access

Remote terminals (teletype machines or cathode ray tubes) may be connected to the B6700 via conventional telephone lines. The personnel at CESL will assign appropriate telephone numbers to users. Before dialing the telephone number, switch the terminal to full duplex. If the terminal does not have a duplex switch, check the duplex after making the phone connection by typing any message and pressing the carriage return. If the message is repeated by the computer, the terminal is operating at half duplex. In that case enter the command ↑ HALF and press the carriage return.

Once the proper phone connection is made, the session is initiated by pressing the carriage return. The receiving computer will respond with the message;

CENTS OF: date, time of message, informative messages for users

Make the connection to the B6700 computer by typing ↑ CON B6700 and pressing carriage return. This may be abbreviated as ↑ CON B6. The computer will respond with a greeting, such as:

CESL TIMESHARING B6700 CANDE... YOU ARE STATION 36

ENTER USERCODE PLEASE

The usercode and password assigned to the user by personnel at CESL should then be entered even if the second message above was not received). If the computer accepts the entry, it will send a message like:

SESSION 7054 ...

At this point it is possible to run any of the programs described in the following chapters or perform any of the operations listed in sections 1.3 and 1.4.

When the user is finished, he may terminate the session by typing the word BYE and pressing the carriage return. The session will also be terminated if the telephone connection with CESL is broken.

1.3 Machine Communication

Communication with the computer is accomplished with the Command AND Edit language (CANDE). Most users will not need to learn the syntax of this language, because their communication with the computer will be directly controlled by the program being executed. That is, all programs are designed to be operated in an interactive mode, with the user simply responding to instructions from the program. Chapters 2, 3 and 4 describe this interaction in detail. If more infor-

mation is desired on the syntax and use of CANDE beyond what is presented herein, it may be obtained from CESL or the Burroughs Corp. (ref. 1.1)

1.3.1 Symbolism

The following conventions are used in the description of the commands:

1) Upper case letters are commands recognized by CANDE. If a portion of the word is underlined, it is permissible to use only that portion.
2) Lower case letters enclosed by the symbols < > are variables which represent information to be supplied by the user.
3) "delim" represents any arbitrary delimiter which may generally be any non-alphanumeric character except one that occurs in the variables being separated. The slash, /, is a particularly convenient delimiter.
4) A sequence number refers to the number associated with a particular line of information.
5) A sequence range list is an inclusive range of sequence numbers, such as 500-1000 or 100-END.

1.3.2 Control Commands

Backspace: depress the keys "CTRL" and "H" simultaneously.

Delete the line: depress the keys "CTRL" and "D" simultaneously.

Terminate the execution of a program (this "kills" the program while it is running): type ?DS and press the carriage return.

Terminate the listing of output on the remote terminal: type ?BRK and press the carriage return, or press the key marked "BREAK".

Question the status of a program while it is running: type ?STATUS and press the carriage return.

Program operation: type RUN <program name> and press carriage return.

1.4 Files

Each timesharing user of the B6700 operates on and with files that are stored on magnetic disc. A file may be the FORTRAN source code for a program, the compiled object version of the program, a set of input or output data, or any other collection of records. The name "file" is probably derived from the use of the word file to describe a group of punched computer cards. The files of interest to the reader include the three programs described in this manual and the files of data stored from previous problems run on these programs. Files may be manipulated and edited by the user from his terminal with CANDE.

The programs will automatically create a data file for each problem the user runs, assigning a name given by the user. The data in these files will allow the user to rerun the problem in the future without re-entering the input data. Although each program contains a routine for modifying data, a few users will find it convenient to examine the data file and modify it directly. It is also possible for the user to create new files if he wishes. A feature that will be incorporated into the programs in the future will allow a user to create a file that may be used as original input to the programs. At present, the original input must be made with the

program operating in interactive communication with the user.

1.4.1 CANDE Commands

In the following commands, a carriage return is required at the end of the line in order to dispatch the command to the computer. The term work file means the file that is actively being examined or modified from the terminal. A filename consists of 1 to 12 words separated by slashes. Each word may contain up to 17 characters, but the total name is limited to 136 characters. Generally 4 to 8 characters are sufficient to describe files without ambiguity. Note that the sequence numbers are of importance for editing, but they are otherwise ignored by the computer.

1.4.2 File Manipulation Commands

MAKE < filename > - creates a new workfile

GET < filename > - recalls a file as the workfile

SAVE < filename > - saves the file in the user's library

REMOVE < filename > - removes the file from the user's library

LIST < filename > - displays the contents of the file on the user's terminal.

TITLE < filename > TO < filename > - changes the title of a file.

FILES - causes a list of all the names of the user's files to be printed on the terminal.

Note that the variables < filename > may be omitted from the commands SAVE, REMOVE, LIST, and TITLE (the first filename only for TITLE) if the file in question is the current workfile.

1.4.3 File Editing Commands

SEQUENCE - invokes the automatic sequence mode, causing the computer to provide the sequence number for each new line. The automatic sequence may be stopped by pressing the carriage return directly after the sequence number is printed. The sequence numbers generally start with 100 and have increments of 100, so it is possible to enter new lines between old ones. It is possible to override these values by entering the desired base number and increment thus:

SEQUENCE < base > ± < increment >

When a sequence has been stopped, it may be resumed by invoking the sequence command with the new base desired. If the workfile has not changed between stopping and restarting a sequence, the computer will automatically begin with the correct new base.

RESEQUENCE < base > ± < increment > - assigns new sequence numbers without changing the order or content of the lines of the workfile. If the base and increment are not specified, 100 will be used for both.

DELETE < sequence range list > - discards the specified lines from the workfile

FIX < sequence number >< delim >< old text >< delim >< new text > replaces the specified item in a line by new material. Some examples of this useful procedure are shown below:

original line:	700 NUMBER AF NODES 267
FIX command:	FIX 700 /AF/OF
new line:	700 NUMBER OF NODES 267

original line	1150 GO TO 733
FIX command:	FIX 1150 /3/4
new line:	1150 GO TO 743

Metz Reference Room
Civil Engineering Department
B106 C. E. Building
University of Illinois
Urbana, Illinois 61801

Note that only the first occurance of the old text is replaced by the new text.

INSERT (< filename >) < sequence range list > AT < base > (± < increment >) - causes the specified lines to be copied in the workfile beginning at the specified base. The items in parentheses are optional, so that the workfile is assumed and an increment of 100 is assumed.

MOVE < sequence range list > TO < base > (± < increment >) moves lines from one point to another in the workfile.

FIND < delim >< text >< delim > - searches the workfile for the specified text.

REPLACE < delim >< text >< delim >< delim >< new text >< delim > (< sequence range list >) - replaces the specified text with the new text at all occurrences within the sequence range list. If no range is given, the command affects the entire workfile.

1.4.4 File Maintenance

It may be necessary for the personnel at CESL to occasionally remove a user's files from the disc and store them on magnetic tape. When this occurs, the computer will give a message similar to:

FILES NOT PRESENT

When this occurs, a phone request to the personnel at CESL to place the user's files on the disc will be necessary.

Chapter Two

DECISION TABLE PROGRAM

2.1 General Description

This program is designed to accept input describing a limited entry decision table and perform operations upon it to aid the user in checking the logical formulation of the decision table and in preparing textual expression of the content of the decision table. It is assumed that the user is familiar with decision tables and has a rudimentary understanding of what a network is, the second in order to interpret a portion of the output of the program. Decision tables and networks are described in Volume I of this report. The text by Pollack (ref. 2.1) is also a good reference for decision tables.

2.1.1 Formulation and Decomposition of Logic

The logical formulation of the decision table is checked by decomposing it into a decision tree, which is simply a graphical representation of the logical content in the form of a network rather than in a tabular display. The process of decomposition will identify all redundant or contradictory rules (that is, all rules that are not unique). Non-unique rules will prevent the synthesis of a unique path in the network for each rule, and the program will call this to the user's attention. The decision tree also identifies any combinations of condition entries that are not given as a rule by showing a branch in the decision tree labeled as an "else" rule. Detection of the else rules allows the user to study the situations that his decision table has not covered and make modifications if it is desired to have a complete table.

2.1.2 Guide to Expression

Textual expression of the logical content of a decision table is generally done in a rulewise manner. That is, each rule (that requires expression) can be associated with a phrase, clause, or sentence. It is expected that the decision network will aid the ordering of the rules, but no specific principles are yet formulated.

2.2 Definitions and Conventions

2.2.1 Terminology

It is not the purpose of this manual to introduce the user to decision tables. The user should understand what is meant by decision table, condition, condition stub, condition entry, action, action stub, action entry and rule (see figure 2.1). This program deals only with limited entry tables, meaning that the condition and action entries must all be logical values. However, the program will accept a somewhat wider range of values than what has been conventional for limited entry tables in the past. The acceptable condition entries are:

T	True
F	False
.	Immaterial
+	Implicitly true
-	Implicitly false

The implicit entries are for use in a rule where the value of a condition may be predetermined by the values of the other conditions. It is not necessary to test an immaterial entry in order to verify a rule. Implicit entries are useful when the conditions are not independent.

The acceptable action entry for input is a number indicating which action is to be executed for each rule. The output displays an "X" in the action entry to indicate which action is to be executed.

The decision tree is a special kind of network. The name tree implies a network that has one root and has no closed loops. The decision tree is constructed by showing one condition at each node. Each node has one branch entering it and two branches leaving it, one representing a true value, the other a false value. At the end of each path is a terminal node representing a rule. There is no unique decision tree for any one decision table, each tree depends on the order that the conditions are used in constructing it. However, each of the possible trees does represent the same decision logic. It is worthwhile to note that the same condition can appear at more than one node, and that the same rule can appear on more than one path, although the latter only occurs when there is an immaterial entry in the rule and the condition which corresponds to it appears in the network where the two paths diverge. See figure 2.2 for an example.

2.2.2 Operating Mode Philosophy

The program is designed to be operated in an interactive mode from a teletypewriter or cathode ray tube terminal. The output may be displayed on the remote terminal or on the line printer at the computer installation. It will be possible to run the program in a batch mode, in the future. The interactive mode seems to be advantageous because of the short response time and the relatively small amount of input data required.

2.2.3 Data Storage

The program creates a data file in permanent memory (magnetic disc) for each decision table with the name supplied by the user. All of the necessary information for recreating the decision table and network is stored in this data file so that it may be re-used at any time. The data file is automatically updated as modifications to the decision table are made. It is possible to gain access to these data files when operating outside of this program, since they are stored in a formatted form. This feature allows use of the more sophisticated editing capabilities of the Command and Edit Language (CANDE) that is a part of the Burroughs B6700 system, as described in Chapter One.

2.2.4 Restrictions

The program is dimensioned to accept tables with up to 27 rules, 27 conditions and 27 actions. Modifying this limitation would require changes in the source code and permanent data file structure of the program. Because of the width of paper on remote terminals, only tables with twelve or fewer rules can be printed in one unit. Larger tables are printed in two portions when the output is to be sent to a teletype terminal.

The condition entries and action entries must correspond to the definition of limited entries discussed previously. The descriptive title of the table is limited to 60 characters in length. The expressions in the condition and action stubs are limited to 10 lines of 30 characters each (total of 300 characters per stub). In addition, the total number of characters in all the condition stubs or action stubs is limited to 1200.

2.2.5 Decomposition Algorithm

The program automatically decomposes the decision table into a decision tree using a procedure similar to the "quick" rule of M. Montalbano (ref. 2.2). It tends to produce a network with some short and some long branches, i.e., a skewed network that isolates a few rules quickly. This is advantageous if those rules are the ones that occur most frequently. The program will also produce a decision tree using a procedure similar to Montalbano's "delayed" rule if the proper command is entered (see Section 2.3.3). This tends to produce a network with branches of relatively equal length. In decision tables with only explicit entries, this network tends to minimize the number of conditions tested, on the average, in order to isolate a rule. Both of these algorithms are described in detail in the technical reference manual.

2.3 Input and Operating Instructions

2.3.1 Starting the Program

The first step is to connect the remote terminal to the Buroughs B6700 computer and to enter the usercode and password. These steps are described in detail in the initial chapter of this manual. The program is initiated by the command:

```
RUN TABLE
```

Once the program has begun, most of the communication will be a two-way interchange between the program and the user. In the following list, the upper case letters without underlining are the program responses and the underlined lines are the user input.

```
ENTER P FOR OUTPUT ON THE ONSITE PRINTER,
OTHERWISE THE OUTPUT WILL BE ON THE REMOTE TERMINAL
```

Any character other than P, or a blank will cause the output to be sent to the user's terminal. If the user's response to this was a P, the program will respond

DO YOU WANT THE INPUT ECHOED ON THE OUTPUT

Simply answer YES or NO. The yes answer will cause each line of input from the terminal to be reproduced on the output. The next response from the program will be

ENTER THE DATA FILE NAME

The user should respond with the particular name he wishes to use, for example:

ABCXYZ

The name consists of a list of one to twelve identifiers separated by slashes. Each identifier may contain up to seventeen characters, but the total must not exceed 136 characters. Generally, it will only be necessary to use one identifier with five to ten characters. If more than one identifier is used, enclose the entire name in quotation marks.

If the program does not find any file with the name given, it will assume that the file will contain new data and will proceed to the data input routine (see section 2.3.2). If a file does exist with the name given, the program will respond:

FILE EXISTS WITH THIS NAME.

DO YOU WANT TO USE IT?

If the user does not intend to use data from an existing file, the correct response is NO. The program will return to request for a file name and the user should give a new name. This is important because only the most recent data is retained in a file. Valuable data may be

lost by inadvertently using the name of an existing file when entering new data.

If the user does intend to use data from an existing file, the correct response to the question above is YES. The program will then ask:

DO YOU WANT TO MODIFY THE EXISTING DATA?

Simply answer YES or NO. A YES sends the program to the input routine. A NO initiates the automatic decomposition of the table into a tree. See section 2.3.3 for the subsequent interaction. The sequence of commands described in this section is shown in the flow chart in Fig. 2.3.

2.3.2 Data Input Routine

The program will give the following message when it enters the input routine:

BEGIN INPUT INSTRUCTIONS. ENTER THE WORD END WHEN FINISHED.

The input of the table is keyed to a series of headings which are nearly self explanatory. The headings may be entered in any order, except as noted. The headings are shown below with the following conventions: The underlined letters are the key letters that the machine looks for, the remainder are used only to increase the readibility; the symbol i means that an integer number must be entered at this point; the symbols < rule > means that the condition entries must be entered using T, F, +, - and . as described previously; <"..."> means that any descriptive phrase for use as a title, condition stub, or action stub may be entered between the quotation marks--note that these titles only serve to make the output more readable, they are not involved in the logic of the program.

NUMBER of RULES i

NUMBER OF CONDITIONS i

NUMBER OF ACTIONS i

TITLE < "title of the decision table" >

RULES - begins the sequence of inputing the rules; the program will print the following:

ENTER THE RULE NUMBER, THE CONDITION ENTRIES, AND THE ACTION ENTRY, ONE RULE TO A LINE. ENTER THE WORD LAST WHEN FINISHED.

i < rule > i - the first i is the rule number followed by condition entries; the last i is the action number.

LAST terminates the sequence of rules

CONDITIONS begins the sequence of input for the condition stubs; the program prints out:

ENTER THE CONDITION NUMBER AND THE STRINGS, ONE 30 CHARACTER STRING TO A LINE. ENTER THE WORD LAST WHEN FINISHED.

i < "..." > i is the condition number.
< "..." > It is not to be repeated if more
< "..." > than one string is necessary to
i < "..." > express the stub. The sequence
i < "..." > of strings is terminated when a new
< "..." > condition number is encountered.

LAST terminates the sequence of conditions

ACTIONS begins the sequence if input for the action stubs. The messages and sequence of commands are exactly analogous to those for the condition stubs.

END terminates the input routine.

The only headings that must follow a specific order are LAST and END. This same routine is used for both new data and for modification of old data. For example, if a mistake is discovered in a previous entry, such as a condition stub, simply re-enter the condition heading, give the condition number, the new stub, and the LAST command. Any item of data may be changed in this manner. The syntax diagram for the input routine commands is shown in Fig. 2.4.

2.3.3 Operational Commands

Once the decision table data is entered, whether from the input routine or from retrieval of an old data file, the original table is printed out and the decomposition into a decision tree proceeds automatically. When it is complete the program prints out the decision tree as described in section 2.4 and issues the following statement, if the table does not contain redundant or contradictory rules:

DECISION NETWORK SUCCESSFULLY COMPLETED

ENTER A PROGRAM COMMAND

At this point the program will accept any of the following commands:

WRITE - the program will print out a version of the table with the rules and conditions ordered as they are encountered in the decision tree.

DELAY - this creates a new decision tree using a modified algorithm. See section 2.2.5 for a brief discussion. The new tree is printed out, and control is returned to the statement ENTER A PROGRAM COMMAND.

Metz Reference Room
Civil Engineering Department
B106 C. E. Building
University of Illinois
Urbana, Illinois 61801

SORT - This reorders the branches of the decision tree so that shorter branches are first. The reordered tree is printed out, and control is returned to the statement, ENTER A PROGRAM COMMAND.

MODIFY - This returns control to the input routine.

NEXT - This allows a new problem to begin. The program will respond ENTER THE DATA FILE NAME.

STOP - This stops the program.

If the decomposition of the table into a decision tree finds any redundant or contradictory rules, the following message will be printed:

THE FOLLOWING RULES ARE REDUNDANT OR CONTRADICTORY (rule number)

YOU MUST MODIFY THE DATA. ENTER THE MODIFY OR STOP COMMAND

The flow chart for the operational commands is shown in Fig. 2.5.

2.4 Output

The original version of the table is printed out once all of the data is in the program. If the decomposition of the table is successful. then the decision tree is displayed with the following conventions:

1) Each condition node is displayed with the letter C followed by the number of the condition;
2) The true branch emanating from each condition node is shown as a series of + symbols;
3) The false branch emanating from each condition node is shown as a series of - symbols;
4) Each branch terminates at a rule node displayed with the letter R followed by the rule number, unless it is an else rule, in which case it is shown as ELSE;

5) Unless the network is sorted, the true branch is shown above the false branch.

If the WRITE command is issued, the table is reprinted after the decision network with the conditions and rules ordered as they are encountered in the network. This ordering corresponds to pre-order as defined by Knuth (ref. 2.3).

2.5 Errors and Error Messages

2.5.1 Incorrect Input or Command

Any line of input that the program cannot interpret will cause the message

INCORRECT INPUT --- RE-ENTER ON A NEW LINE

to be printed. If the program is being used in a batch mode, such an error will terminate the program.

2.5.2 Non-Unique Rules

Redundant or contradictory rules will cause the program to suspend decomposition of the table. The incorrect rules will be identified to the user, so that the data may be modified. Rules are redundant if they have no logical difference in the condition entry and have the same action entry. Rules are contradictory if they have no logical difference in the condition entry and have different action entries.

2.5.3 Else Rules

The program will identify all the possible else rules in the decision tree. The user can examine the else rule simply by traversing the path back to the start of the network, noting the appropriate condition entries as he goes. Any conditions that do not appear on the branch have immaterial entries.

The user must be cautioned that spurious else rules may be generated from tables with implicit entries. A spurious else rule is defined as a rule that is not included in the original table because some implicit entry prohibits it. It can be identified because it will have the same condition entries as the rule with the implicit entry, except at that entry it will have the opposite logical value. Not all tables with implicit entries will develop these spurious else rules; in fact, most will not. The program will not automatically label them as spurious.

2.6 Example Problem

The following example is self-explanatory, and illustrates most of the features of the program. The lines with a "u" marked on the left are user input.

```
u RUN TABLE
  #RUNNING 1278

  ENTER P FOR OUTPUT ON THE ON-SITE PRINTER,
  OTHERWISE THE OUTPUT WILL BE ON THE REMOTE TERMINAL
u

  ENTER THE DATA FILE NAME
u "AISI/COMPR"

  BEGIN INPUT INSTRUCTIONS.  ENTER THE WORD END WHEN FINISHED.
u TITLE  "COMPRESSION ON UNSTIFFENED ELEMENTS"
u NUMBER OF RULES 6
u NUMBER OF CONDITIONS 6
u NUMBER OF ACTIONS 5
u RULES

  ENTER THE RULE NUMBER, THE CONDITION ENTRIES, AND THE ACTION ENTRY,
  ONE RULE TO A LINE.    ENTER THE WORD LAST WHEN FINISHED.
u 1 T T . . . . 1
u 2 F T T T . F 2
u 3 F F T T . F 3
u 4 . . F T T . 3
u 5 . . F T F . 4
u 6 F . T T . T 5
u LAST
u CONDITIONS

  ENTER THE CONDITION NUMBER AND THE STRINGS,
  30 CHARACTERS TO A LINE.  ENTER THE WORD LAST WHEN FINISHED.
u 1 " W/T < 63.3/SRFY"
u 2 " W/T/ <-
u 2 " W/T < 144/SRFY"
u 3 " W/T < 25"
u 4 " W/T < 60"
u 5 " MEMBER TYPE = ANGLE STRUT"
u 6 " FY < 33"
u LAST
u ACTIONS

  ENTER THE ACTION NUMBER AND THE STRING, ONE ACTION TO A LINE.
  ENTER THE WORD LAST WHEN FINISHED.
u 1 " FC = 0.60 FY"
u 2 " FC = 0.767 FY - "
u   "    0.00264 W/T (FY)**1.5"
u 3 " FC = 8000/(W/T)**2"
u 4 " FC = 19.8 - 0. 28 W/T
u 5 " FC = 0.60 FY - "
u   "    (W/T - 63.3/SRFY)*
u   "    (0.60 FY - 12.8)/
u  "   25(1 - 2.53/SRFY")"
u LAST
u END
```

ORIGINAL DECISION TABLE

COMPRESSION ON UNSTIFFENED ELEMENTS

```
                                          1  2  3  4  5  6
 1  W/T < 63.3/SRFY                   *   T  F  F  .  .  F
 2  W/T < 144/SRFY                    *   T  T  F  .  .  .
 3  W/T < 25                          *   .  T  T  F  F  T
 4  W/T < 60                          *   .  T  T  T  T  T
 5  MEMBER TYPE = ANGLE STRUT         *   .  .  .  T  F  .
 6  FY < 33                           *   .  F  F  .  .  T
*************************************************************
 1  FC = 0.60 FY                      *   X
 2  FC = 0.767 FY -                   *      X
       0.00264 W/T (FY)**1.5          *
 3  FC = 8000/(W/T)**2                *         X  X
 4  FC = 19.8 - 0.28 W/T              *               X
 5  FC = 0.60 FY -                    *                  X
       (W/T - 63.3/SRFY)*             *
       (0.60 FY - 12.8)/              *
      25(1 - 2.53/SRFY)               *
```

THE FOLLOWING RULES ARE REDUNDANT OR CONTRADICTORY: 1 4
YOUR MUST MODIFY THE DATA. ENTER THE MODIFY OR STOP COMMAND.<-

u MODIFY

BEGIN INPUT INSTRUCTIONS. ENTER THE WORD END WHEN FINISHED.

u RULES

ENTER THE RULE NUMBER, THE CONDITION ENTRIES, AND THE ACTION ENTRY,
ONE RULE TO A LINE. ENTER THE WORD LAST WHEN FINISHED.

u 4 F . F T T . 3
u 5 F . F T F . 4
u LAST
u END

ORIGINAL DECISION TABLE

COMPRESSION ON UNSTIFFENED ELEMENTS

```
                                          1  2  3  4  5  6
 1  W/T < 63.3/SRFY                   *   T  F  F  F  F  F
 2  W/T < 144/SRFY                    *   T  T  F  .  .  .
 3  W/T < 25                          *   .  T  T  F  F  T
 4  W/T < 60                          *   .  T  T  T  T  T
 5  MEMBER TYPE = ANGLE STRUT         *   .  .  .  T  F  .
 6  FY < 33                           *   .  F  F  .  .  T
*************************************************************
 1  FC = 0.60 FY                      *   X
 2  FC = 0.767 FY -                   *      X
       0.00264 W/T (FY)**1.5          *
 3  FC = 8000/(W/T)**2                *         X  X
 4  FC = 19.8 - 0.28 W/T              *               X
 5  FC = 0.60 FY -                    *                  X
       (W/T - 63.3/SRFY)*             *
       (0.60 FY - 12.8)/              *
      25(1 - 2.53/SRFY)               *
```

```
DECISION NETWORK SUCCESSFULLY COMPLETED


DERIVED DECISION NETWORK

C1  + + + C2  + + + R1
 -          -
 -          -
 -          -  - - ELSE
 -
 -
 -  - - - C4  + + + C3  + + + C6  + + + R6
            -          -          -
            -          -          -
            -          -          -  - - - C2  + + + R2
            -          -                     -
            -          -                     -
            -          -                     -  - - - R3
            -          -
            -          -
            -          -  - - - C5  + + + R4
            -                     -
            -                     -
            -                     -  - - - R5
            -
            -
            -  - - ELSE


  ENTER A PROGRAM COMMAND
u MODIFY

  BEGIN INPUT INSTRUCTIONS.  ENTER THE WORD END WHEN FINISHED.
u RULES

  ENTER THE RULE NUMBER, THE CONDITION ENTRIES, AND THE ACTION ENTRY,
  ONE RULE TO A LINE.   ENTER THE WORD LAST WHEN FINISHED.
u 1 T + + + . . 1
u 2 F T + + . F 2
u 3 - F T + . F 3
u 4 - . F T T . 3
u 5 - . F T F . 4
u 6 F . T + . T 5
u LAT

  INCORRECT INPUT---REENTER ON A NEW LINE  u  LAST
u END
```

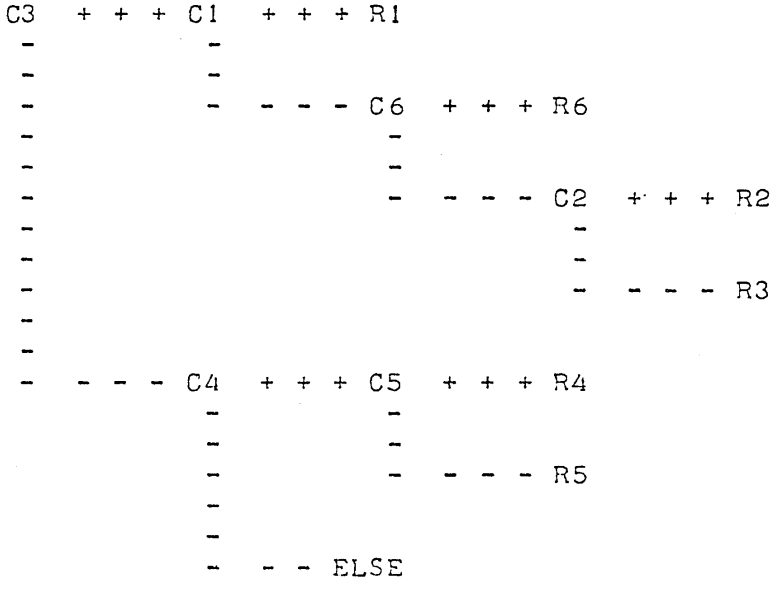
ORIGINAL DECISION TABLE

COMPRESSION ON UNSTIFFENED ELEMENTS

			1	2	3	4	5	6
1	W/T < 63.3/SRFY	*	T	F	-	-	-	F
2	W/T < 144/SRFY	*	+	T	F	.	.	.
3	W/T < 25	*	+	+	T	F	F	T
4	W/T < 60	*	+	+	+	T	T	+
5	MEMBER TYPE = ANGLE STRUT	*	.	.	.	T	F	.
6	FY < 33	*	.	F	F	.	.	T
		*						
1	FC = 0.60 FY	*	X					
2	FC = 0.767 FY - 0.00264 W/T (FY)**1.5	*		X				
3	FC = 8000/(W/T)**2	*			X	X		
4	FC = 19.8 - 0.28 W/T	*					X	
5	FC = 0.60 FY - (W/T - 63.3/SRFY)* (0.60 FY - 12.8)/ 25(1 - 2.53/SRFY)	*						X

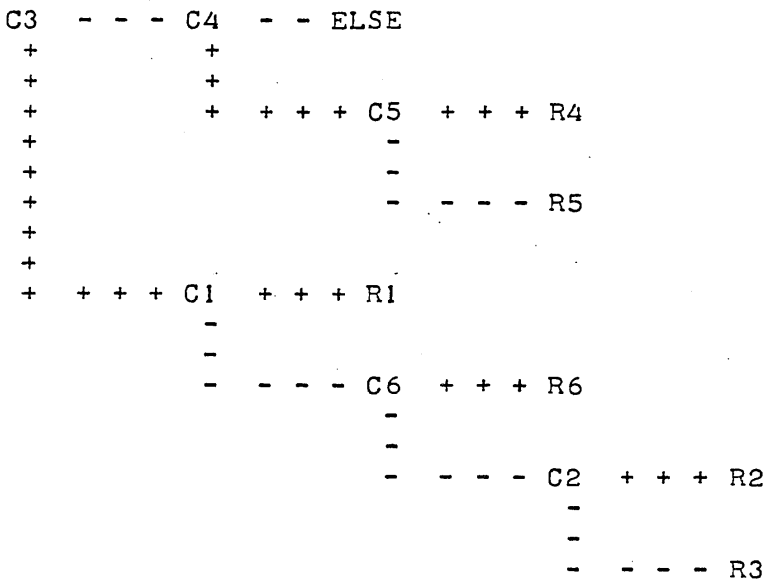
DECISION NETWORK SUCCESSFULLY COMPLETED

DERIVED DECISION NETWORK



```
  ENTER A PROGRAM COMMAND
u SORT
```

DECISION NETWORK SORTED BY BRANCH LENGTH




```
  ENTER A PROGRAM COMMAND
u WRITE
```

DECISION TABLE WITH CONDITIONS AND RULES IN PREORDER

COMPRESSION ON UNSTIFFENED ELEMENTS

			4	5	1	6	2	3
3	W/T < 25	*	+	+	T	F	F	T
4	W/T < 60	*	+	+	+	T	T	+
5	MEMBER TYPE = ANGLE STRUT	*	.	.	.	T	F	.
1	W/T < 63.3/SRFY	*	T	F	-	-	-	F
6	FY < 33	*	.	F	F	.	.	T
2	W/T < 144/SRFY	*	+	T	F	.	.	.
**********		**********						
1	FC = 0.60 FY	*			X			
2	FC = 0.767 FY - 0.00264 W/T (FY)**1.5	*					X	
3	FC = 8000/(W/T)**2	*	X					X
4	FC = 19.8 - 0.28 W/T	*		X				
5	FC = 0.60 FY - (W/T - 63.3/SRFY)* (0.60 FY - 12.8)/ 25(1 - 2.53/SRFY)	*				X		

```
   ENTER A PROGRAM COMMAND
u  DELAY

   DECISION NETWORK SUCCESSFULLY COMPLETED


   DECISION NETWORK DERIVED WITH THE  DELAYED DECISION RULE

   C3  + + + C1  + + + R1
    -         -
    -         -
    -         -  - - - C6  + + + R6
    -                    -
    -                    -
    -                    -  - - - C2  + + + R2
    -                                 -
    -                                 -
    -                                 -  - - - R3
    -
    -
    -  - - - C5  + + +  C4  + + + R4
               -         -
               -         -
               -         -  - - ELSE
               -
               -
               -  - - - C4  + + + R5
                         -
                         -
                         -  - - ELSE


   ENTER A PROGRAM COMMAND
u  STOP
   #ET=33:20.3 PT=15.2 IO=1.3
```

Chapter Three

INFORMATION NETWORK PROGRAM

3.1 General Description

This program accepts data describing the nodes in a network and their connections to adjacent nodes. The network is assembled by the program, and the information is displayed for the user as a modified tree in a variety of different ordering schemes. The user should be familiar with the concept of a network, especially as it relates to the of information in a specification. Volume I of this report describes information networks and their place in the organization of specifications. Knuth (ref. 2.3) contains a discussion of operations on information structures that is the basis for many of the operations in this program.

3.2 Definitions and Concepts

The operating mode philosophy and the method of data storage for this program are essentially the same as described previously for the decision table program (sections 2.2.2 and 2.2.3). It should be noted that the amount of data required to define an information network for any portion of a specification larger than a page or so is considerably larger than the amount of data for an average decision table.

3.2.1 Terminology

There are several words that are important for the proper use of the program:

NODE - any item of information in the specification, such as an input parameter, a criterion, or a value defined by functional or logical operation.

INGREDIENT - any node that may be directly required to establish the value of a second node is said to be an ingredient of the second node.

DEPENDENT - any node whose value may be directly affected by the value of a second node is said to be a dependent of the second node.

INGREDIENCE - (of a node) is the network beginning at the node and including all of its ingredient nodes , then all of their ingredients and so on, the process being repeated until those with no ingredients are reached.

DEPENDENCE - (of a node) is the network beginning at the node and including all of its dependent nodes, then all of their dependents, and so on until the nodes with no dependents are reached.

INPUT NODES - those nodes that have no ingredients.

OUTPUT NODES - those nodes that have no dependents.

INPUT LEVEL OR OUTPUT LEVEL - the number of steps from the node in question to the input (or output) nodes along the longest path that goes through the node in question.

FLOAT - the numerical difference between the longest path from input to output through a given node and the longest such path in the entire network.

TREE - a network that has one root and has no closed loops.

3.2.2 Restrictions

The maximum size of network that the program will accept is 500 nodes and 1000 branches. The output will not display more than 23 levels in a network, however, this has not proved to be a limitation since the largest number of levels encountered in the information network for any specification that has been analyzed at the University of Illinois is ten. The textual descriptions of the nodes are limited to 60 characters in length.

3.2.3 Operations Performed

The required input to the program consists of the number of each node and the numbers of each of its ingredients. Once this data is entered, the program calculates the dependents, the levels from input and output, and the float for each node. After these calculations are complete the user may request to see the ingredience or dependence network for any node. If the user wishes to see the entire network he may request it as the "complete" ingredience or dependence. The user may also request that the order in which the ingredients and dependents of a node are listed be altered by sorting them with respect to their levels or floats. The sorting algorithm can order the ingredients (or dependents) so that those with the largest (or smallest) level (or float) occur first in the network.

As was pointed out in volume I of this report, an ingredience network can be used to order the textual definition of a specification according to "conditional" ordering; that is, the items are only defined once they are used. The dependence network is used in a similar fashion to order the text according to "direct" ordering; each item being defined

before it is used. The sorting algorithm does not change this property; it merely refines it. An ingredience network sorted so that those ingredients with the largest float or level are first corresponds to conditional ordering with those items that have the smallest number of levels of precedence involved in their definition being defined first. There is generally some correlation between the depth of precedence and the complexity of definition, so that this particular ordering generally places the simpler definition first. It is useful to experiment with the various possible sorts to gain experience in their possible benefits.

3.3 Input and Operating Instructions

3.3.1 Starting the Program

The steps in starting the information network program are exactly the same as those used to start the decision table program, with one exception: the name of the program to be used in the RUN command is NETWORK. Following the command RUN NETWORK the procedure is just as described in section 2.3.1 and figure 2.3

3.3.2 Data Input Routine

The following message is printed when the program enters the data input routine:

> ENTER THE NODE NUMBERS AND ASSOCIATED DATA, ONE NODE TO A LINE. IF IT IS NECESSARY TO USE TWO LINES FOR THE INGREDIENTS, ENTER A COMMA AT THE BEGINNING OF THE SECOND LINE. ENTER 'END' TO SIGNIFY THE END OF THE DATA.

Each node is identified by a cardinal number. The program requires the number of each node and the number of its dependents. The program will also accept a label of six characters or less and a descriptive title of sixty characters or less. The labels and titles are useful as aids in interpreting the output, but they are not used in the logic of the program.

The set of numbers used to identify the nodes must be inclusive; that is, the largest number used must be the total number of nodes in the network, and no smaller number may be left unused. It is not necessary to enter the nodes in order, however.

The first entry on each line of input should be a node number, with one exception. Following the node number, the label, description, or ingredients may be entered in any order.

<node> <label> <"title"> <ingredients>

The first character of each label must be a letter, and the description must be enclosed in quotation marks. The exception will be discussed following this example.

Example: the following lines are all acceptable ways of entering information about node 176; which has ingredient nodes 75, 150, and 201, a label of FAX, and description of "allowable axial compression stress".

Method 1:

176 75 150 201 FAX

176 "ALLOWABLE AXIAL COMPRESSION STRESS"

Method 2:

176 FAX "ALLOWABLE AXIAL COMPRESSION STRESS"

176 75 150 201

Method 3:

176 "ALLOWABLE AXIAL COMPRESSION STRESS"

176 FAX 75 150 201

Not all possible ways of entering the data have been shown. The important point is that the first number on a line is the node number about which the information is going to be entered. The exception to this occurs when there are too many ingredients to fit on one line of input. In this case a comma is entered at the beginning of the second line to indicate that the list of ingredients is being continued. For example:

177 101 119 212 203 331 222 262

, 275 309

all of the numbers represent ingredients of node 177.

It is possible to change any of the information associated with a node simply by entering the new data. For example if it were desired to change the label of node 176 as entered above to AXSTRS and to include node 119 among its ingredients, the following instruction would suffice:

176 AXSTRS 75 119 150 201

If any one of the ingredients is to be changed, the entire list of ingredients must be re-entered. In the special case that a node with no ingredients had been incorrectly given some ingredients, the following command is to be used:

<node number> 0

The zero (the number, not the letter) is the key entry that allows a node to be disassociated from its old ingredients without entering new ingredients for it.

When data is being entered in the input routine it is not stored on the disc until a SAVE or END command is issued. When entering networks with a large amount of data, it is prudent to protect against a machine failure by issuing a SAVE command periodically (every 20 lines or so). Once the SAVE has been issued, simply continue entering the data. For example:

```
:
74 TIME "LENGTH OF TIME ERECTED"
75 CONDSS "CONDITION OF SIGN AND SUPPORT"
76 SIZACC "SIZE ACCEPTABLE" 73 7 48 69
SAVE
77 FMACC "FRAME AND MATERIALS ACCEPTABLE" 8 73 7
:
```

The command END is used to terminate the input routine and return control to the main program. The input routine is used for both entry of new data and correction of old data. The flow diagram for use input is shown in figure 3.1.

3.3.3 Operational Commands

When the data entry is complete, the program calculates the dependents, levels from input and output, and the float for each node. If this work is successfully completed, the program will issue the statement:

ENTER A PROGRAM COMMAND

At this point, the program will accept any of the following commands: (the symbolism used herein is the same as described in sections 1.3.1 and 2.3.1).

INGREDIENCE (TITLE) - the program will prepare to print out an ingredience network. If the word TITLE is included, the descriptive titles will be used in the output; if not, the labels will be used. The program will respond with the following instruction:

ENTER THE ROOT NODE NUMBER--OR THE WORK 'COMPLETE

The word complete causes the entire network to be displayed as an ingredience network of a ficticious node assumed to be a dependent of all those nodes in the network with no dependents.

DEPENDENCE (TITLE) - the program will prepare to print out a dependence network. The format of the command and the program response are identical to that described above for INGREDIENCE.

SORT - the program will prepare to reorder the ingredients and dependents of each node. The program will make three requests to obtain parameters for the sorting process.

ENTER THE VALUE FOR FIRST PRIORITY SORTING

Either FLOAT or LEVEL should be entered.

ENTER THE MODE FOR FIRST PRIORITY SORTING.

Either SMALL OR LARGE should be entered.

ENTER THE MODE FOR SECOND PRIORITY SORTING

Either SMALL or LARGE should be entered.

When describing the mode for sorting, SMALL is taken to mean the selection of the ingredients or dependents with the smallest float or level first. This corresponds to the most densely populated and longest branches of the network being placed first. LARGE causes the opposite process to occur.

Once the network has been sorted, it will remain so until a new sort is ordered or the problem is ended, so the subsequent use of INGREDIENCE or DEPENDENCE will display the sorted network.

MODIFY - the program will return to the input routine so that the data may be changed.

WRITE - the program will print a tabular display of all the information entered and derived for each node: the label, description, ingredients, dependents, input level, output level, and float. When the output is being printed on a remote terminal, the table will be printed in two portions.

NEXT - the program will accept a new problem, the first response of the program will be ENTER THE DATA FILE NAME.

STOP - this stops the program.

The flow chart for the operational commands is shown in fig. 3.2.

3.4 Output

The output of the program is a graphical representation of the network. The form of the network is modified to that of a tree. That is, all the closed loops (meshes) in the network are broken. The break is shown by repeating the node with a negative sign in front of it. The negative sign indicates that the node has appeared previously (above) in the network, so that the branch actually would be directed upward to that previous occurrence. An example of a network and its modified computer printed version is shown in figure 3.3. The asterisk after a node with a negative sign indicates that the network continues on past the node and that this continuation is only shown at the first occurrence of the node. Note that the tracing of nodes is always done vertically since each node is always printed at the same level from input or output.

3.5 Errors and Error Messages

As with the program TABLE, any line of input that the program cannot interpret will cause the message

INCORRECT INPUT -- RE-ENTER ON A NEW LINE

to be printed. If the program is being used in a batch node, the error will terminate the program.

The only other error messages generated by the program are caused by defects in the definition of the network. The message:

PROGRAM STOPPED -- NO STARTING NODE IN THE NETWORK

indicates that no node exists for which there are no ingredients. The message:

PROGRAM STOPPED -- NETWORK CONTAINS A CIRCULAR LOOP

indicates that some node appears in its own ingredience network. The program accepts networks with closed loops if it is not possible to travel around the loop without changing the direction of the branches at least once. If such a circuit is possible, it means that the value of some node depends on a prior definition of its value. This type of iterative procedure is rare in design specifications. If such a case is encountered, it will be necessary to break the loop by defining two nodes, an initial value and a final value, in order to use this program.

3.6 Example Problem

The following example illustrates many of the features of the programs. The lines with a "u" marked on the left are user input.

```
u  RUN  NETWORK
   #RUNNING 1334

   ENTER P FOR OUTPUT ON THE ON-SITE PRINTER,
   OTHERWISE THE OUTPUT WILL BE ON THE REMOTE TERMINAL
u  REMOTE

   ENTER THE DATA FILE NAME
u  "BOCA/623"

   ENTER THE NODE NUMBERS AND ASSOCIATED DATA, ONE NODE  TO A LINE.
   IF IT IS NECESSARY TO USE TWO LINES FOR THE INGREDIENTS,
   ENTER A COMMA AT THE BEGINNING OF THE SECOND LINE.
   ENTER 'END' TO SIGNIFY THE END OF THE DATA.

   INPUT ERROR---RE-ENTER ON A  NEW LINE
u  1 ACCEPT "FIRE ESCAPE ACCEPTABLE 2 3 4 6 7 8 9
u  2 GROUP "USE GROUP"
u  3 SPORD "SPECIAL ORDER OF BUILDING OFFICIAL"
u  4 EXBLDG "EXISTING BUILDING"
u  5 HTLMT "HEIGHT LIMEIT"
u  5 "HEIGHT LIMIT"
u  56 NSTOR "NUMBER OF STORIES"
u  7 HIEEIGHT "HEIGHT, FEET"
u  8 CONSWR "CONSTRUCTION IN ACCORD WITH RULES "
uu 8 2 16 27 29 30 31
u  9 NOALT "MORE ADEQUATE EXITWAY IMPOSSIBLE"
u  10 FRONT "FRONT OF BUILDING"
u  11 PROJ "PROJECTING BEYOND FUILCIN"
u  11 "PROJECTION BEYOND BUILDING LINE"
u  SAVE

   INPUT ERROR---RE-ENTER ON A  NEW LINE
u  12 HTLL " HEIGHT LOWEST LANDING ABOVE GRADE"
u  13 CBALST "COUNTER BALANCED STAIR TO STREET"
u  14 FLROOF "FIXED LADDER TO ROOF"
u  15 ALLEY "ALLEY OR THOROUGHFARE LEESS THAN 30 FEET WIDE"
u  16 LIVELD "DESINGGN LIVE LOAD"
u  17 NONCOM "STEEL OR OTHER NONCOMBUSTIBLE MATERIAL"
u  18 WOOD "WOOD NOT LESS THAN TWO  INCHES THICK"
u  19 TYPE "TYPE OF CONSTRUCTION"
u  20 FDIST "FIRE DISTRICT"
u  21 SWIDTH "STAIR WIDTH"
u  22 RISER "RISER HEIGHT"
u  SAVE

   INPUT ERROR---RE-ENTER ON A  NEW LINE
u  23 TREAD "TREAD DEPTH"
u  24 LWIDTH  "LANDING WIDTH"
u  25 LLENG "LANDING LENGTH"
u  26 LBLW "LANDING BELOW ACCESS"
u  27 OPPROT "HOUR OPENING PROTECTIVE"
u  28 WFL "WITHIN FIRE LIMITS"
u  29 LCLEAR "PROPER LANDING CLEARANCE"
u  30 ACAMAT "ACCEPTABLE MATERIAL" 6 7 17 18 19 20 28 32 33
u  29 10 11 12 13 14 15
u  31 ACDIM "ACCEPTALBLE DIMENSIONS" 21 22 23 2 4 25 26
u  32 NOCC "NUMBER OF OCCUPANTS"
u  33 COMBUS "WOOD OR SIMILARILY CONMBUSTIBLE"
u  END
```

```
   ENTER A PROGRAM COMMAND
u  INGREDIENCE

   ENTER THE ROOT NODE NUMBER---OR THE WORD 'COMPLETE'
u  COMPLETE

   UNSORTED

   GLOBAL INGREDIENCE OF COMPLETE NETWORK

   EXTREME LEVEL FROM OUTPUT

      0         1         2
       1  ACCEPT
       3  SPORD
       4  EXBLDG
       5  HTLMT
       8  CONSWR
       :........31  ACDIM
       :         :........26  LBLW
       :         :........25  LLENG
       :         :........24  LWIDTH
       :         :........23  TREAD
       :         :........22  RISER
       :         :........21  SWIDTH
       :........30  ACMAT
       :         :........33  COMBUS
       :         :........32  NOCC
       :         :........28  WFL
       :         :........20  FDIST
       :         :........19  TYPE
       :         :........18  WOOD
       :         :........17  NONCOM
       :         :.........7  HEIGHT
       :         :.........6  NSTOR
       :........29  LCLEAR
       :         :........15  ALLEY
       :         :........14  FLROOF
       :         :........13  CBALST
       :         :........12  HTLL
       :         :........11  PROJ
       :         :........10  FRONT
       :........27  OPPROT
       :........16  LIVELD
       :.........2  GROUP
       9  NOALT
   ENTER A PROGRAM COMMAND
u  MODIFY

   ENTER THE NODE NUMBERS AND ASSOCIATED DATA, ONE NODE  TO A LINE.
   IF IT IS NECESSARY TO USE TWO LINES FOR THE INGREDIENTS,
   ENTER A COMMA AT THE BEGINNING OF THE SECOND LINE.
   ENTER 'END' TO SIGNIFY THE END OF THE DATA.
u  1 2 3 4 6 7 8 9
u  END

   ENTER A PROGRAM COMMAND
u  INGREDIENCE

   ENTER THE ROOT NODE NUMBER---OR THE WORD 'COMPLETE'
u  COMPLETE
```

```
UNSORTED

GLOBAL INGREDIENCE OF COMPLETE NETWORK

EXTREME LEVEL FROM OUTPUT

   0         1          2          3
    1  ACCEPT
    :.........9  NOALT
    :.........8  CONSWR
    :          :.........31  ACDIM
    :          :          :.........26  LBLW
    :          :          :.........25  LLENG
    :          :          :.........24  LWIDTH
    :          :          :.........23  TREAD
    :          :          :.........22  RISER
    :          :          :.........21  SWIDTH
    :          :.........30  ACMAT
    :          :          :.........33  COMBUS
    :          :          :.........32  NOCC
    :          :          :.........28  WFL
    :          :          :.........20  FDIST
    :          :          :.........19  TYPE
    :          :          :.........18  WOOD
    :          :          :.........17  NONCOM
    :          :          :.........7   HEIGHT
    :          :          :.........6   NSTOR
    :          :.........29  LCLEAR
    :          :          :.........15  ALLEY
    :          :          :.........14  FLROOF
    :          :          :.........13  CBALST
    :          :          :.........12  HTLL
    :          :          :.........11  PROJ
    :          :          :.........10  FRONT
    :          :.........27  OPPROT
    :          :.........16  LIVELD
    :          :.........2   GROUP
    :.............................-7   HEIGHT
    :.............................-6   NSTOR
    :.........4  EXBLDG
    :.........3  SPORD
    :..................-2  GROUP
    5  HTLMT
ENTER A PROGRAM COMMAND
u  SORT

ENTER THE VALUE FOR FIRST PRIORTY SORTING
u  FLOAT

ENTER THE MODE FOR FIRST PRIORTY SORTING
u  LARGE

ENTER THE MODE FOR SECOND PRIORTY SORTING
u  LARGE

ENTER A PROGRAM COMMAND
u  INGREDIENCE TITLE

ENTER THE ROOT NODE NUMBER---OR THE WORD 'COMPLETE'
u  COMPLIETE
```

Metz Reference Room
Civil Engineering Department
3106 C. E. Building
University of Illinois
Urbana, Illinois 61801

```
SORTED FIRST BY LARGE FLOAT AND THEN BY LARGE LEVEL

GLOBAL INGREDIENCE OF COMPLETE NETWORK

EXTREME LEVEL FROM OUTPUT

   0         1         2         3         4         5         6         7
    5  HEIGHT LIMIT
    1  FIRE ESCAPE ACCEPTABLE 2 3 4 6 7 8 9
    :.........9  MORE ADEQUATE EXITWAY IMPOSSIBLE
    :.........4  EXISTING BUILDING
    :.........3  SPECIAL ORDER OF BUILDING OFFICIAL
    :.................2  USE GROUP
    :.............................6  NUMBER OF STORIES
    :.............................7  HEIGHT, FEET
    :.........8  CONSTRUCTION IN ACCORD WITH RULES
              :.........27  HOUR OPENING PROTECTIVE
              :.........16  DESIGN LIVE LOAD
              :.........-2  USE GROUP
              :.........31  ACCEPTABLE DIMENSIONS
              :          :.........26  LANDING BELOW ACCESS
              :          :.........25  LANDING LENGTH
              :          :.........24  LANDING WIDTH
              :          :.........23  TREAD DEPTH
              :          :.........22  RISER HEIGHT
              :          :.........21  STAIR WIDTH
              :.........30  ACCEPTABLE MATERIAL
              :          :.........33  WOOD OR SIMILARILY COMBUSTIBLE
              :          :.........32  NUMBER OF OCCUPANTS
              :          :.........28  WITHIN FIRE LIMITS
              :          :.........20  FIRE DISTRICT
              :          :.........19  TYPE OF CONSTRUCTION
              :          :.........18  WOOD NOT LESS THAN TWO INCHES THICK
              :          :.........17  STEEL OR OTHER NONCOMBUSTIBLE MATER
              :          :.........-7  HEIGHT, FEET
              :          :.........-6  NUMBER OF STORIES
              :.........29  PROPER LANDING CLEARANCE
                         :.........15  ALLEY OR THOROUGHFARE LESS THAN 30
                         :.........14  FIXED LADDER TO ROOF
                         :.........13  COUNTER BALANCED STAIR TO STREET
                         :.........12  HEIGHT LOWEST LANDING ABOVE GRADE
                         :.........11  PROJECTION BEYOND BUILDING LINE
                         :.........10  FRONT OF BUILDING
SORTED FIRST BY LARGE FLOAT AND THEN BY LARGE LEVEL

GLOBAL INGREDIENCE OF COMPLETE NETWORK

EXTREME LEVEL FROM OUTPUT

   7         8         9        10        11        12        13        4

ENTER A PROGRAM COMMAND
u  STOP
#ET=26:43.3 PT=9.6 IO=5.7
```

Chapter Four

OUTLINE PROGRAM

4.1 General Description

The input for this program consists of the arguments selected for classification of the specification and the major provisions selected to appear in the outline. Each provision must be associated with at least one argument. The arguments represent headings in the outline. They are entered in a trial outline format which provides the guide for the final structure of the outline. Such arguments taken as a set must provide a reasonable basis for organizing the specification as described in volume 1 of this report.

The output of the program is a refined version of the trial outline given by the input. It is not related to the hierarchical structure of the information network of the specification, allowing an applicability to the wide variety of specifications and freedom to explore alternative organizations.

4.2 Definitions and Concepts

The operating mode philosophy and the method of data storage for this program are essentially the same as described previously for the decision table and information network programs (see sections 2.2.2 and 2.2.3).

The arguments used for headings in the outline are stored as a group of trees. A tree is a special kind of network that has one root and has no closed loops. The headings of a conventional outline with indentations can be

represented as a group of trees. Each of the headings on the extreme left is called a root and all the subsequent headings until the next root belong to one tree. Each heading in the tree is associated with its parent, which is the last previous heading that projects to the left. Figure 4.1 shows the correspondence of a set of headings and a tree for a hypothetical outline.

The program will accept up to 30 arguments and 30 provisions. Modification of this limit is relatively simple, but does require a few changes in the FORTRAN code of the program. No more than five arguments may be associated with any one provision.

The algorithm maps the provisions onto the argument trees. Where provisions are associated with more than one argument (a frequent case), the algorithm appends the trees of the secondary arguments onto the tree of the first argument listed for the provision. Thus no provision is mapped onto the outline until all of its associated arguments have been entered. Arguments are omitted from the appended trees if they are not associated with any provisions at that point in the outline. The overall order of the outline can be varied by changing the order in which the trees are taken.

4.3 Input and Operating Instruction

4.3.1 Starting the Program

The steps in starting the outline program are the same as those for the decision table and information network programs except that the name of the program to be used in the RUN command is OUTLINE. Refer to 2.3.1 and figure 2.3 for the details.

4.3.2 Data Input Routine

The following message is printed when the program enters the input routine:

BEGIN INPUT INSTRUCTIONS. ENTER THE WORD

END WHEN FINISHED.

The input is keyed to three headings which are nearly self explanatory. They may be entered in any order. The symbolism used in the following is the same as that used in chapters 2 and 3.

ARGUMENTS - begin the sequence for input of arguments.

The program will respond:

ENTER THE LIST OF ARGUMENTS AND THEIR PARENTS,

ONE LINE FOR EACH ARGUMENT

The correct format for the input is:

<arg. no.> <"title"> <parent arg. no.>

The title is not necessary for the calculations, but makes interpretation of the output much easier. The title must be inside quotation marks and less than 30 characters long. The argument numbers must be sequential, beginning with one.

PROVISIONS - begins the sequence for input of the provisions.

The program will respond:

ENTER THE LIST OF PROVISIONS AND ASSOCIATED

ARGUMENTS, ONE LINE FOR EACH PROVISION.

the correct format for the input is:

<prov. no.> <"title"> <assoc. arg. nos.>

The title has the same requirements as for the argument titles. The provision numbers must once again be sequential. There must be at least one associated argument for each provision. If there is more than one, separate the adjacent numbers with at least one blank space.

ORDER - prepares the program to accept the numbers of the roots of the argument trees in the order that the user desires to make the outline. The numbers may be entered on the same line as the word ORDER or on the line following.

END - terminates the sequence of input instructions.

4.3.2 Operational Commands

The mapping algorithm proceeds automatically after the input is completed. When the outline is finished, the program will request:

ENTER A PROGRAM COMMAND

Any of the following commands may be used:

ORDER - a new sequence of argument trees may be entered, as described for the input routine. The command END must be used when the order statement is finished. The new outline will be generated and the program will once again request a program command.

NEXT - a new problem may be entered. The first response of the program will be to request a data file name.

4.4 Output

The output contains the outline, or "table of contents," and consists of a hierarchical structure of headings and a list of related major provisions under these headings.

4.5 Error Messages

In general, the program cannot be killed by inadvertent errors in input. Improper types of input will be ignored and the user will be instructed again to enter the proper type of data. The program will stop if improper input is encountered ten times in succession.

If the program is being used in a batch mode, an input error will cause the message:

"PROGRAM STOPPED---INCORRECT INPUT"

Another possible error in the data structure that will stop the program is caused when any associated argument provision is not found in the hierarchical trees of arguments.

In this case, the message given by the program is:

"PROGRAM STOPPED---SOME PROVISIONS HAVE NOT BEEN OUTLINED"

4.6 Example

The following example is taken from the initial text of the section 618 from the 1974 BOCA changes.

The arguments selected for outlining are shown in table 4.1. It can be noticed, that there are three trees of arguments, headed by the following argument roots: No. 1, Types of Interior Stairways, No. 5, Design Requirements, and No. 11 Appurtenances. The major provisions along with their associated arguments, are shown in the table 4.2.

The order in which the argument trees will be expressed in this example is defined by the argument-root numbers 1, 5, 11.

The input, as well as the resulting outline, are printed on the following pages.

Table 4.1 List of Arguments and their Parents

ARGUMENT NUMBER	ARGUMENT TITLE	PARENT
1	TYPES OF INTERIOR STAIRWAYS	0
2	REQUIRED EXIT	1
3	SUPPLEMENTARY EXIT	1
4	OTHER	1
5	DESIGN REQUIREMENTS	0
6	DIMENSIONS	5
7	STRENGTH	5
8	MATERIALS	5
9	COMBUSTIBLE	8
10	NON-COMBUSTIBLE	8
11	APPURTENANCES	0
12	LANDING PLATFORMS	11
13	HANDRAILS AND GUARDS	11
14	ENCLOSURES	11
15	DOORS	11

Table 4.2 Major Provisions with their Associated Argument Numbers

PROVISION NUMBER	PROVISION TITLE	ARGUMENT NUMBERS
1	ADQ. INTERIOR EXIT STAIRWAY	2
2	ACC. RISE BETWEEN LANDING PLATFORMS	12
3	ACCEPTABLE WINDERS	4
4	ACCEPTABLE HANDRAILS	13
5	ACCEPTABLE HANDRAIL EXTENSION	13
6	ACCEPTABLE GUARDS	13
7	ACC. HANDRAILS AND GUARDS	13
8	ACCEPTABLE STAIRWAY DIMENSIONS	6
9	ACCEPTABLE LANDING PLATFORM DIM.	12
10	ACC. STAIRWAY EXIT DOORS	2, 15
11	ACCEPTABLE FIRE DOOR	2, 15
12	ACCEPTABLE SPIRAL STAIRWAY	3
13	MEETS COMBUSTIBILITY REQUIREMENTS	2, 9
14	ACCEPTABLE SUPPLEMENTARY EXIT	3
15	ACCEPTABLE STAIRWAY CONSTRUCTION	2, 5
16	ADEQUATE DESIGN LOADS	7
17	ADEQUATE ENCLOSURES	2, 14
18	ACC. TREAD AND RISER DIMENSIONS	6

```
RUN OUTLINE
#RUNNING 1377

ENTER P FOR OUTPUT ON THE ON-SITE PRINTER,
OTHERWISE THE OUTPUT WILL BE ON THE REMOTE TERMINAL  REM

DO YOU WANT THE PRINTOUT OF THE INPUT   NO

ENTER THE DATA FILE NAME     "BOCA/618"

BEGIN INPUT INSTRUCTIONS.  ENTER THE WORD END WHEN FINISHED.
INPUT ERROR --- RE-ENTER ON A NEW LINE<-
PROVISIONS

ENTER THE LIST OF PROVISIONS AND ASSOCIATED ARGUMENTS
ONE LINE FOR EACH PROVISION<-
 1 "ADQ. INTERIOR EXIT STAIRWAY" 2
 2 "ACC. RISE BETWEEN LANDINGS" 12
 3 "ACCEPTABLE WINDERS" 4
 4 "ACCEPTABLE HANDRAILS" 13
 5 "ACCEPTABLE HANDRAIL EXTENSION" 13
 6 "ACCEPTABLE GUARDS" 13
 7 "ACC. HANDRAILS AND GUARDS" 13
 8 "ACCEPTABLE STAIRWAY DIMENSIONS" 6
 9 "ACCEPTABLE LANDING DIMENSIONS" 12
10 "ACCEPTABLE STAIRWAY EXIT DOORS" 2 15
11 "ACCEPTABLE FIRE DOOR" 2 15
12 "ACC. SPIRAL STAIRWAY" 3
13 "MEETS COMBUSTIBILITY REQTS." 2 9
14 "ACCEPTABLE SUPPLEMENTARY EXIT" 3
15 "ACCEPTABLE CONSTRUCTION" 2 5
16 "ADEQUATE DESIGN LOADS" 7
17 "ADEQUATE ENCLOSURES" 2 14
18 "ACC. TREAD AND RISER DIMENSION" 6
ARGUMENTS

ENTER THE LIST OF ARGUMENTS AND THEIR PARENTS
ONE LINE FOR EACH ARGUMENT<-
1 "TYPES OF INTERIOR STAIRWAYS" 0
2 "REQUIRED EXIT" 1
3 "SUPPLEMENTARY EXIT" 1
4 "OTHER" 1
5 "DESIGN REQUIREMENTS" 0
6 "DIMENSIONS" 5
7 "STRENGTH" 5
8 "MATERIALS" 5
9 "COMBUSTIBLE" 8
10 "NONCOMBUSTIBLE" 8
11 "APPURTENANCES" 0
12 "LANDING PLATFORMS" 11
13 "HANDRAILS AND GUARDS" 11
14 "ENCLOSURES" 11
15 "DOORS" 11
ORDER 1 5 11
     END
```

O U T L I N E H E A D I N G	P R O V I S I O N	
TYPES OF INTERIUR STAIRWAYS		
REQUIRED EXIT		
. . . . .	1	ADQ. INTERIOR EXIT STAIRWAY
ENCLOSURES		
. . . . .	17	ADEQUATE ENCLOSURES
DOORS		
. . . . .	10	ACCEPTABLE STAIRWAY EXIT DOORS
. . . . .	11	ACC. FIRE DOOR
DESIGN REQUIREMENTS		
. . . . .	15	ACCEPTABLE STAIRWAY CONSTRUCTI
COMBUSTIBLE		
. . . . .	13	MEETS COMBUSTIBILIYY REQUIREME
SUPPLEMENTARY EXIT		
. . . . .	12	ACC. SPIRAL STAIRWAY
. . . . .	14	ACCEPTABLE SUPPLEMENTARY EXIT
OTHER		
. . . . .	3	ACCEPTABLE WINDES
DESIGN REQUIREMENTS		
DIMENSIONS		
. . . . .	8	ACCEPTABLE STAIRWAY DIMENSIONS
. . . . .	18	ACC. TREAD AND RISER DIMENSION
STRENGTH		
. . . . .	16	ADEQUATE DESIGN LOADS
MATERIALS		
COMBUTIBLE		
NONCOMBUSTIBLE		
APPURTENANCES		
LANDING PLATFORMS		
. . . . .	2	ACC. RISE BETWEEN LANDING PLAT
. . . . .	9	ACCEPTABLE LANDING PLATFORM DI
HANDRAILS AND GUARDS		
. . . . .	4	ACCEPTABLE HANDRAILS
. . . . .	5	ACCEPTABLE HANDRAIL EXTENSION
. . . . .	6	ACCEPTABLE GUARDS
. . . . .	7	ACC. HANDRAILS AND GUARDS

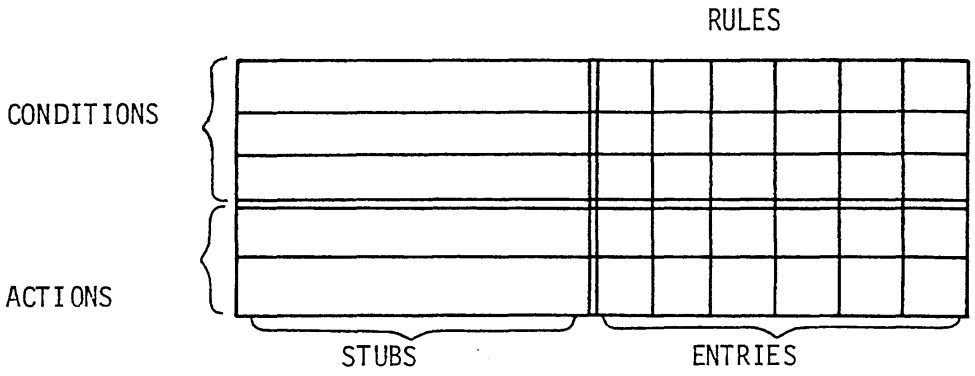
REFERENCES

1.1 Burroughs Corp. B6700/B7700 Command and Edit (CANDE) Language Manual, Form No. 5000318.

2.1 Pollack, A. L., Decision Tables: Theory and Practice, Wiley-Interscience, New York, N.Y., 1971.

2.2 M. Mantalbano, "Tables, Flow Charts, and Program Logic," IBM Systems Journal, Sept., 1962, pp. 51-63.

2.3 Knuth, D. E., The Art of Computer Programming, Vol. I, Fundamental Algorithms, Addison-Wesley, New York, N.Y., 1968.



a) Regions of a Decision Table

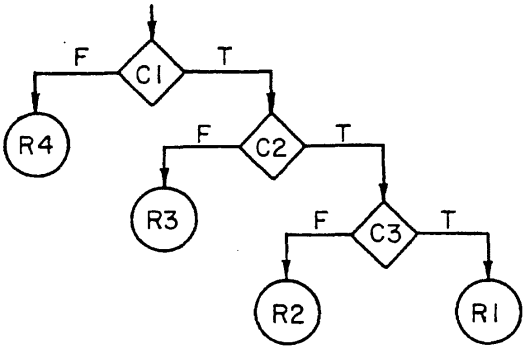
	RULE 1	RULE 2	RULE 3
RAINING?	F	F	T
EARLIER THAN 7:45?	T	F	.
USE BICYCLE	X		
USE CAR		X	X

b) A simple decision table with two conditions, two actions, and three rules written in limited entry format.

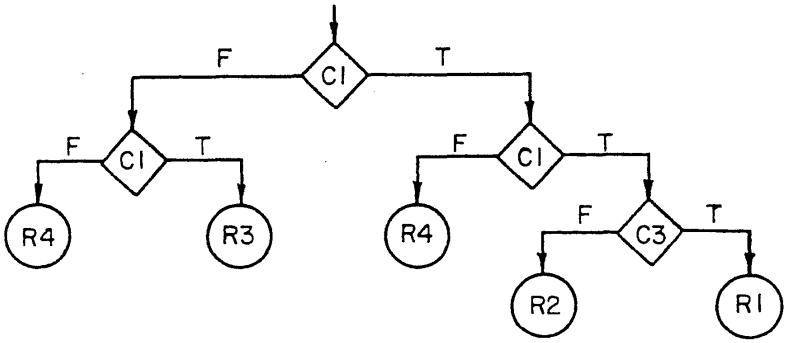
Figure 2.1 Decision Table Terminology

	R1	R2	R3	R4
C1	T	T	T	F
C2	T	T	F	•
C3	T	F	•	•

a) Decision table showing the condition entry only.



b) Decision network formed by testing condition C1 first, then condition C2.



c) Decision network formed by testing condition C2 first, then condition C1. Note that rule R4 appears on two paths, and that they converge at C2 where R4 has an immaterial entry.

Figure 2.2 Decision Table and Decision Trees

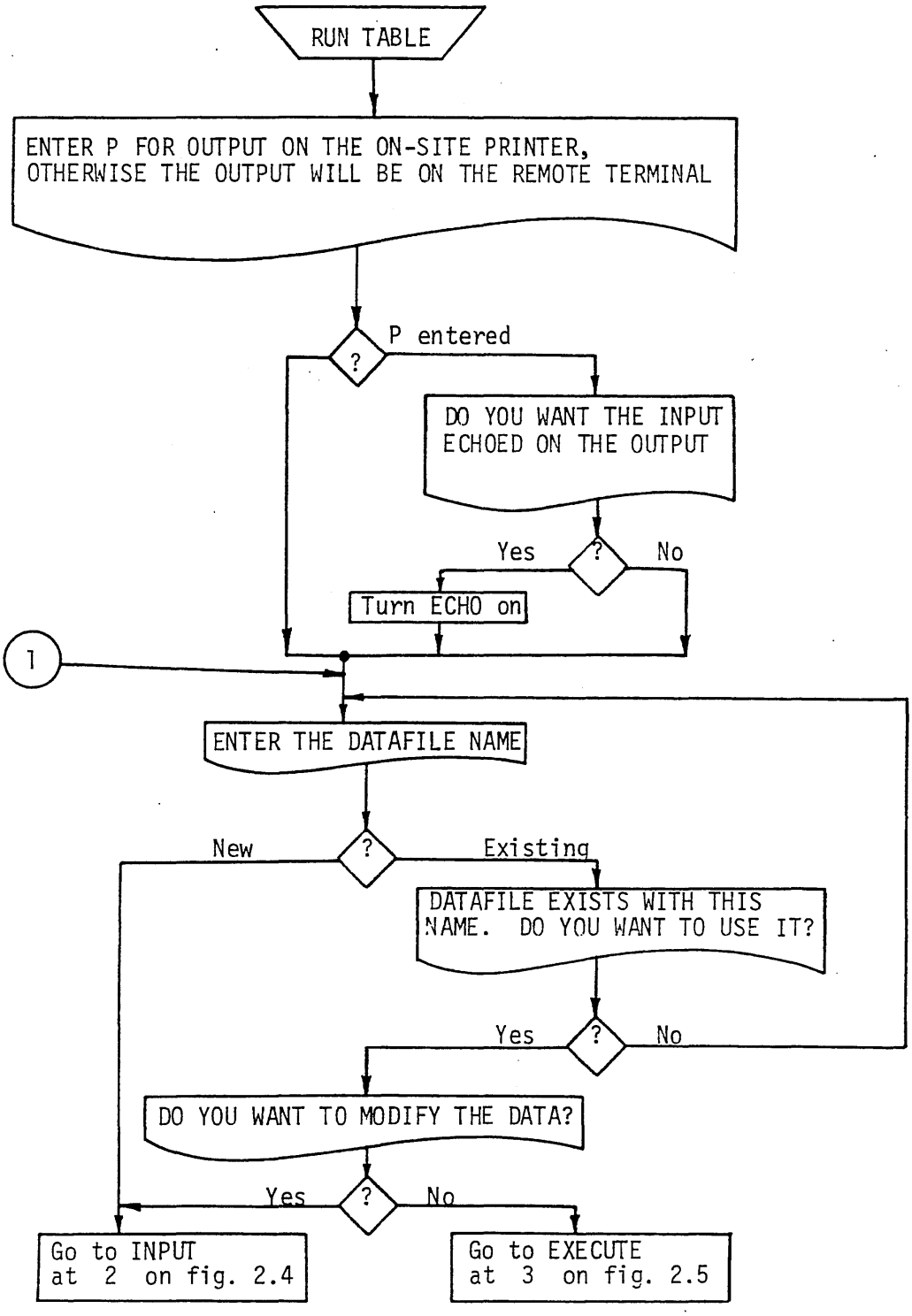


Figure 2.3 Flow Chart for Problem Initialization

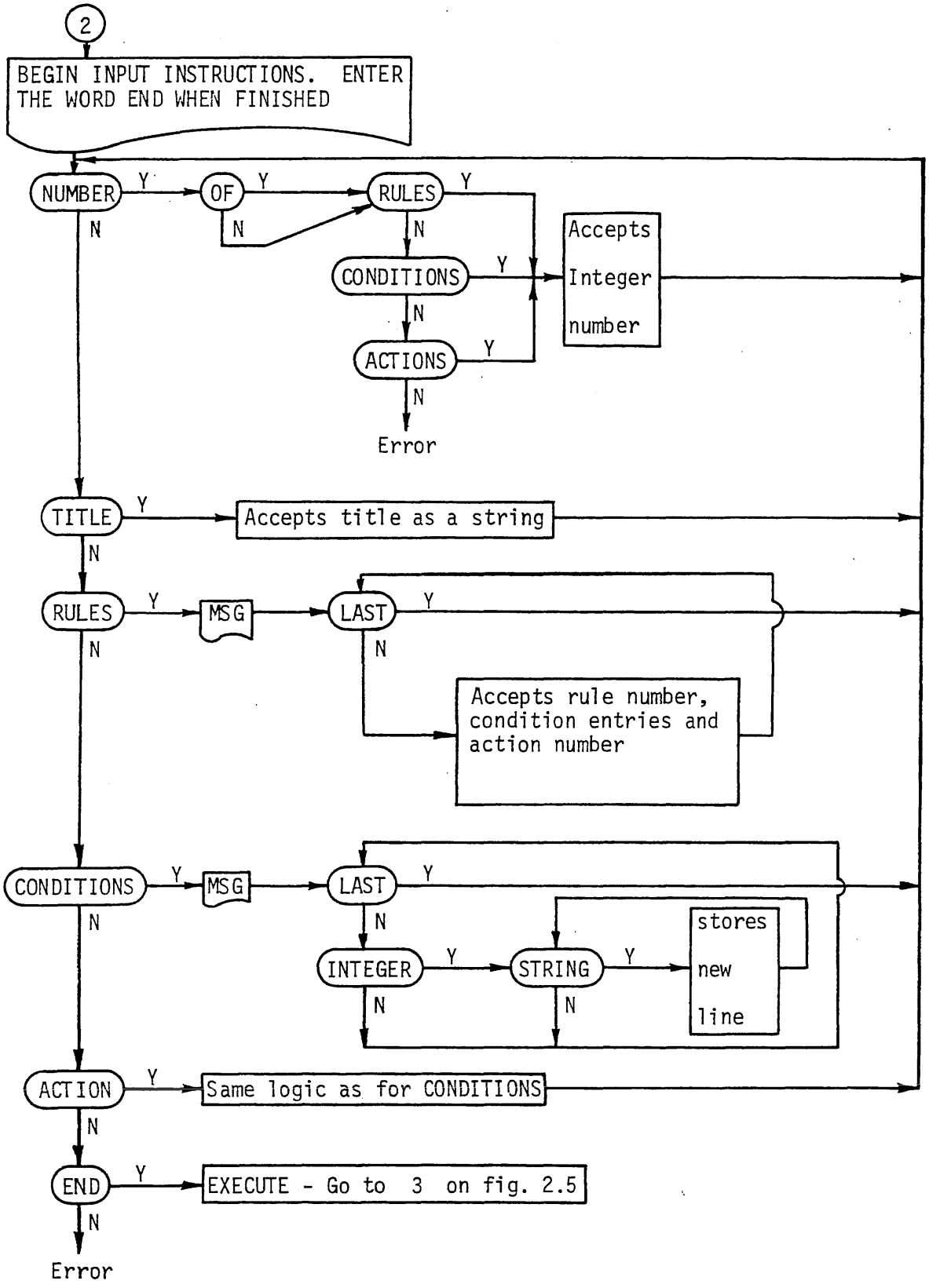


Figure 2.4 Flow Chart for Data Input

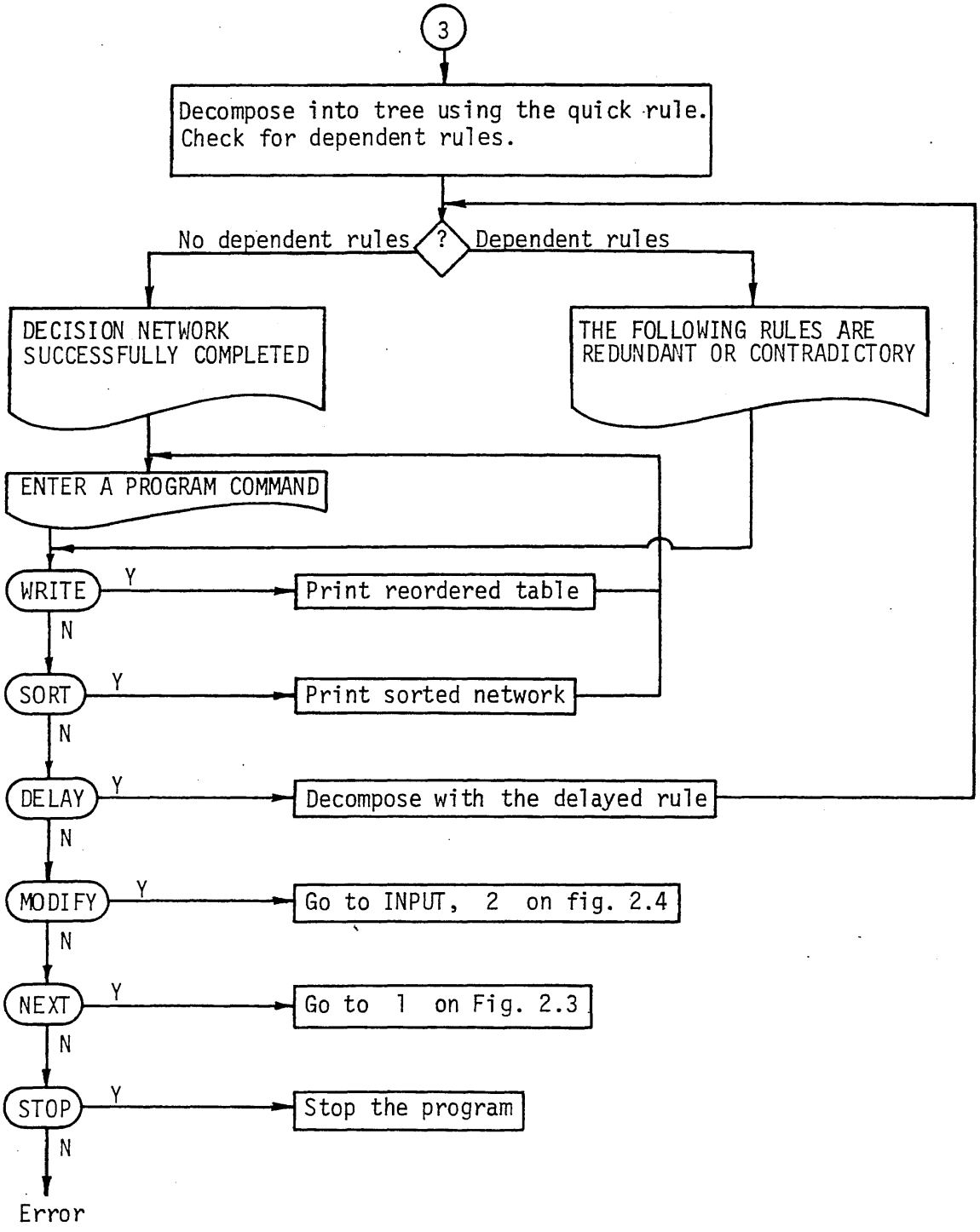


Figure 2.5 Flow Chart for Program Control

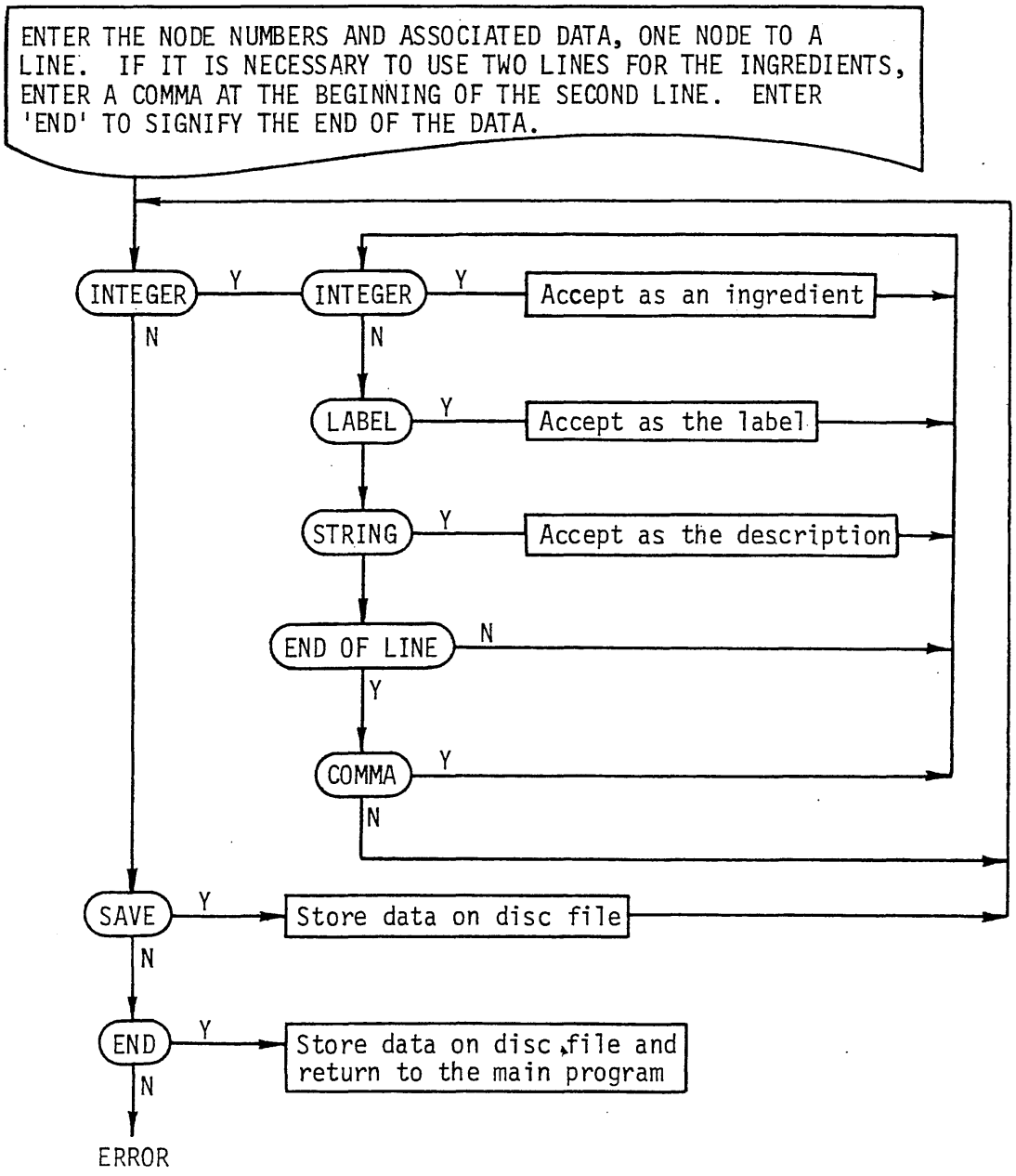


Figure 3.1 Flow Chart for Data Input

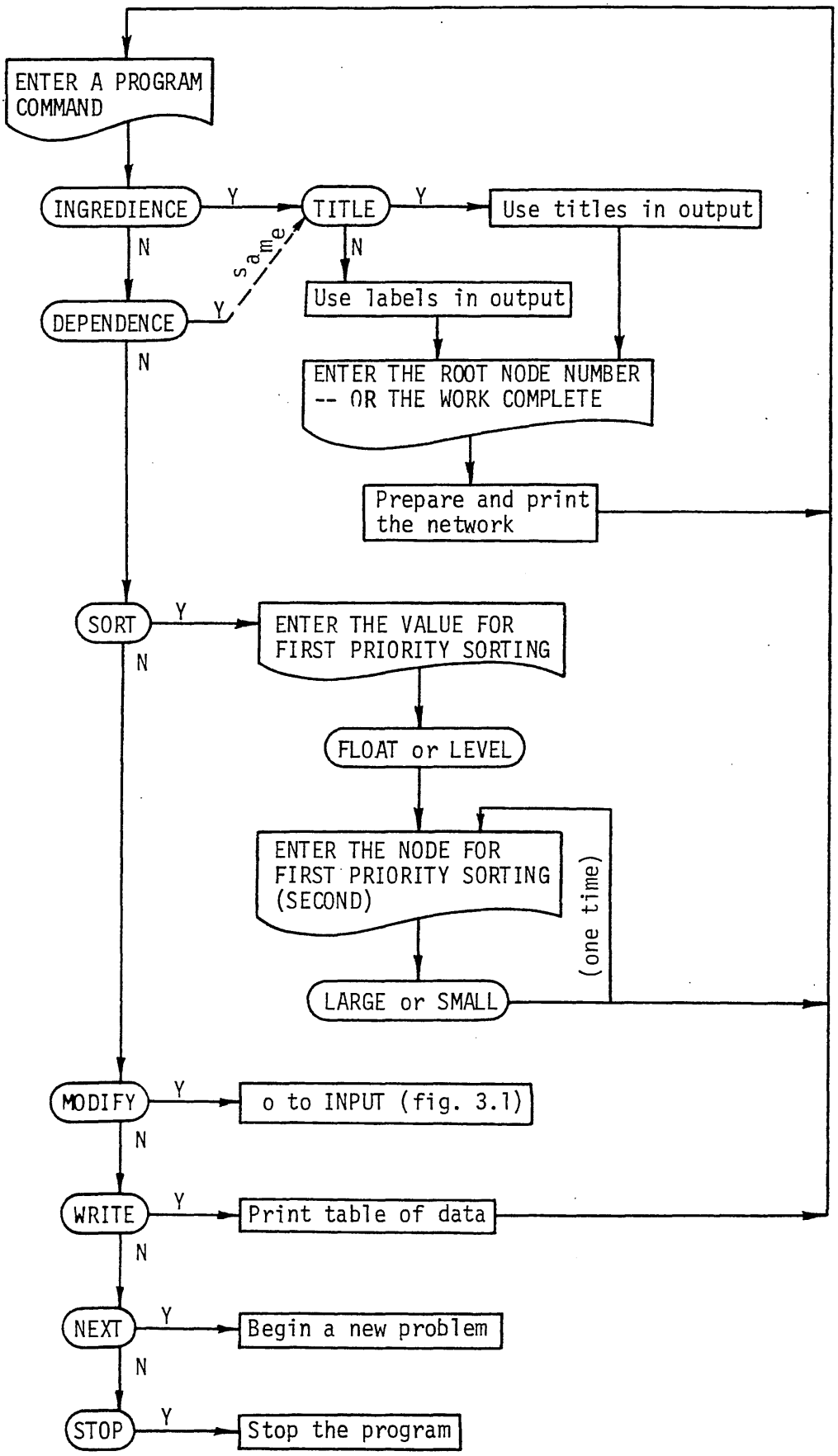
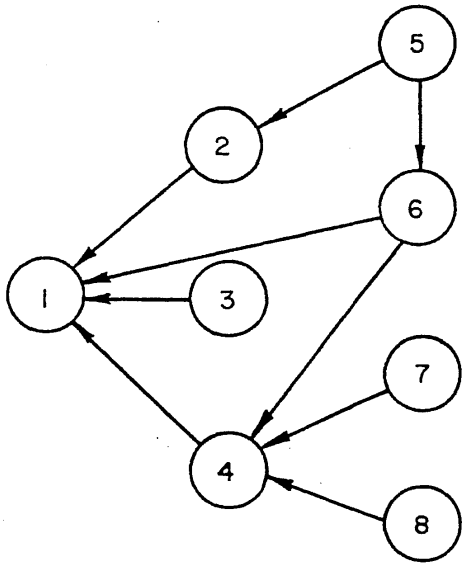


Figure 3.2 Flow Chart for Operational Commands



a) Hypothetical Network

```
GLOBAL INGREDIENCE OF COMPLETE NETWORK

EXTREME LEVEL FROM OUTPUT

    0         1         2         3

    1
    :.........2
    :         :.....................5
    :.........3
    :.........4
    :         :.........6
    :         :         :.........-5
    :         :.........7
    :         :.........8
    :...................-6*
```

b) Printed Representation

Figure 3.3 Network Representation

HEADING 1

HEADING 2

HEADING 3

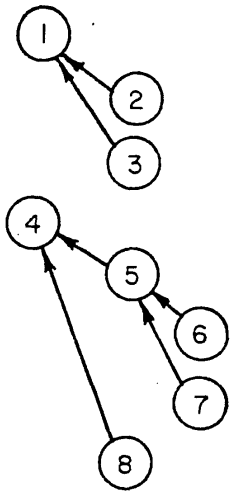
HEADING 4

HEADING 5

HEADING 6

HEADING 7

HEADING 8



a) Hypothetical Outline b) Trees

c) Table of Parents

Heading No.	Parent Heading No.
1	
2	1
3	1
4	
5	4
6	5
7	5
8	4

Figure 4.1 Argument Tree Structure

```
RUN OUTLINE
#RUNNING 3447

ENTER P FOR OUTPUT ON THE ON-SITE PRINTER,
OTHERWISE THE OUTPUT WILL BE ON THE REMOTE TERMINAL    REM

DO YOU WANT THE PRINTOUT OF THE INPUT      NO

ENTER THE DATA FILE NAME     EXAMPLE

BEGIN INPUT INSTRUCTIONS.  ENTER THE WORD END WHEN FINISHED.
INPUT ERROR --- RE-ENTER ON A NEW LINE<-
PROVISIONS

ENTER THE LIST OF PROVISIONS AND ASSOCIATED ARGUMENTS
ONE LINE FOR EACH PROVISION<-
1 "ADQ. INTEIOR EXIT STAIRWAY" 2
2 "ACC. RISE BETWEEN LANDINGS" 12
3 "ACCEPTABLE WINDERS" 4
4 "ACCEPTABLE HANDRAILS"1 13
5 "ACCEPTABLE HANDRAIL EXTENSION" 13
6 "ACCEPTABLE GUARDS" 13
7 "ACC. HANDRAILS AND GUARDS" 13
8 "ACCEPTABLE STAIRWAY DIMENSIONS" 6
9 "ACCEPTABLE LANDING DIMENSIONS" 12
10 "ACCEPTABLE STAIRWAY EXIT DOORS" 2 15
11 "ACCEPTABLE FIRE DOOR" 2 15
12 "ACC. SPIRAL STAIRWAY" 3
13 "MEETS COMBUSTIBILITY REQTS." 2 9
14 "ACCEPTABLE SUPPLEMENTARY EXIT" 3
15 "ACCEPTABLE CONSTRUCTION" 2 5
16 "ADEQUATE DESIGN LOADS" 7
17 "ADEQUATE ENCLOSURES" 2 14
18 "ACC. TREAD AND RISER DIMENSIONS" 6
ARGUMENTS

ENTER THE LIST OF ARGUMENTS AND THEIR PARENTS
ONE LINE FOR EACH ARGUMENT<-
1 "TYPES OF INTERIOR STAIRWAYS " 0
2 "REQUIRED EXIT" 1
2 "SUPPLEMENTARY EXIT" 1
3 "SUPPLEMENTARY EXIT" 1
2 "REQUIRED EXIT" 1
4 "OTHER" 1
5 "DESIGN REQUIREMENTS" 0
6 "DIMENSIONS " 5
7 "STRENGTH %5
8 "MATERIALS" 5
9 "COMBUSTIBLE" 8
10 "NONCOMBULSTIBLE" 8
11 "APPURTENANCES" 0
12 "LANDING PLATFORMS" 11
13 "ENCLOSURES" 11
14 "ENCLOSURES" 11
13 "HANDRAILS AND GURARDS" 11
15 "DOORS" 11
ORDER 1 5 11
```