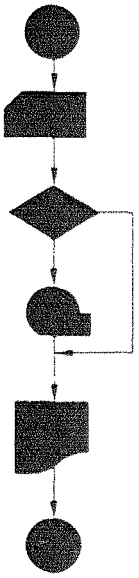
THE NORTHEASTERN FOREST-INVENTORY DATA-PROCESSING SYSTEM. II. DESCRIPTION OF SUBSYSTEM EDIT.



by
Robert W. Wilson Jr.
and Robert C. Peters

U. S. FOREST SERVICE RESEARCH PAPER NE-70
1967
NORTHEASTERN FOREST EXPERIMENT STATION, UPPER DARBY, PA.
FOREST SERVICE, U.S. DEPARTMENT OF AGRICULTURE
RICHARD D. LANE, DIRECTOR

About the Authors

ROBERT W. WILSON, JR. took his Bachelor's degree at The Pennsylvania State University in 1947 and his Master's and Ph.D. degrees at Yale University in 1948 and 1965, respectively. He joined the U. S. Forest Service in 1948 and has worked in various research capacities for the Northeastern Forest Experiment Station. From 1961 to 1965 he was in charge of the Station's biometrics unit at New Haven, Conn. He is assigned at present to the Forest Insect and Disease Laboratory at West Haven, Conn.

ROBERT C. PETERS obtained his Bachelor's degree from the University of California in 1960 and his Master's at Yale University in 1961. He joined the Forest Service in 1961 as a research forester, and was assigned to the Station's biometrics unit from 1961 until 1965, when the unit was discontinued. Mr. Peters played a key role in the development of the data-processing system reported here.

THE NORTHEASTERN FOREST-INVENTORY DATA-PROCESSING SYSTEM. II. DESCRIPTION OF SUBSYSTEM EDIT.

■

Contents

PREFACE

THIS paper is the second in a series of ten papers prepared to describe the forest-inventory data-processing system of the Northeastern Forest Experiment Station. This system was devised for using modern, large-scale, high-speed computers in processing forest-inventory data. The series will comprise the following papers:

I. Introduction.
II. Description of subsystem EDIT.
III. Operation of subsystem EDIT.
IV. Information for programmers — subsystem EDIT.
V. Description of subsystem TABLE.
VI. Operation of subsystem TABLE.
VII. Information for programmers—subsystem TABLE.
VIII. Description of subsystem OUTPUT.
IX. Operation of subsystem OUTPUT.
X. Information for programmers — subsystem OUTPUT.

II-A. INTRODUCTION

ONE of the major projects of the U.S. Forest Service is a nationwide forest survey, which is designed to obtain useful and timely information about the timber resources of the United States. In the course of the surveys, which are made mainly on a state-by-state basis, great masses of detailed data are collected about timber volumes, growth, timber cut, and other characteristics of the timber resource.

In recent years the volume of information obtained from forest-survey field plots has increased greatly. The task of compiling and analyzing this mass of data with mechanical computing machines was both cumbersome and time-consuming.

A solution to this problem was seen in the development of the high-speed electronic computers. The Northeastern Forest Experiment Station, which was responsible for conducting the forest survey of the heavily forested Northeastern States, investigated the possibilities and devised the Northeastern Forest-Inventory Data-Processing System.

The purpose of this paper is to describe a part of the system, the subsystem EDIT, which is designed for high-speed editing of large numbers of unit records.

The editing functions are of three general kinds: checking values in each record for legality or reasonableness; generating additional values for each record; and accumulating values from each record in one-dimensional tables to produce simple summaries of information from the subsets of the records.

Each unit record is processed independently of all other records in the input file. Processing is controlled by a set of operations chosen from among those available in the program. These operations give the scope and flexibility needed to tailor the program to specific editing problems of great variety and complexity. And, although designed primarily to edit records, the ingenious user will find the program serves other purposes as well. For example, it may be used to form record-frequency distributions; to accumulate sums and sums of squares of specified values in records; to update master summary files; or to search a record file for specified sets of records.

The program is written in the standard FORTRAN IV language, and is operative at the Yale University Computer Center on an IBM 7094/7040 Direct Coupled System under the IBSYS Operating System with IBJOB Processor.[1] It will operate with little or no modification on other comparable systems.

Part IV in this series contains a selection of programming information that will be useful if the standard version of the program must be modified for any reason. Detailed instructions for setting up and executing jobs with the standard version are given in part III of the series. Copies of these publications and information about the FORTRAN IV source decks for the program can be obtained from the Northeastern Forest Experiment Station, 6816 Market Street, Upper Darby, Pennsylvania 19082.

II-B. PROGRAM OUTPUTS

The program provides four kinds of output: (1) a magnetic tape file of records that have passed successfully through all editing operations, (2) a printed list of messages, (3) printed tabulations of the information summaries for record subsets, and (4) a deck of punched cards containing the same information summary tables. The user may exercise several options with respect to all but the last of these outputs.

Ordinarily, the magnetic tape file of correct, augmented records is the primary goal and most important output of an editing job. However, in some types of problem, this output may not be required; so provision is made to suppress it (part III-C, item 161). Again, the correct output will normally be used only as input to another processing program; so provision is made to write the output file in binary mode. However, the alternative of writing the tape in binary coded decimal (BCD) mode is also available, should it be necessary to print the output. Finally, the correct record output may be punched in cards (BCD mode), but this option should be used only if the number of output records is small.

[1] Mention of a particular product should not be construed as an endorsement by the Forest Service or the U. S. Department of Agriculture.

Each correct output record may consist of up to 132 data fields that contain either values from the corresponding input record or values generated for the record during processing. The values are listed in the record in the same order that they are input and generated. Any format may be specified for BCD output records (part III-C, item 162).

The messages printed fall into three categories: (1) those that summarize the actual processing of the job; (2) those that signal an error in preparing the job control deck; and (3) those that describe errors detected in input records by the editing operations. Appropriate messages from the first two categories will always be printed, but the user may elect to suppress messages in the third category (part III-C, item 161). If the input record error messages are not suppressed a message will be printed for every input record in which an error is detected. The message will describe the error and will include a copy of the input record. If an input record contains more than one error only the first one detected will be identified in the message.

The printed tabulations of information summarizing sets of records are optional. Such tables are formed in the computer only if specified by the user (part III-C, item 131); and if formed, they are printed only if so specified (part III-C, item 171). If formed, the tables will always be punched in cards for use in a subsequent pass (part III-C, sec. 140). If printed up to four tables of the same length may be printed side by side on a page. The tabulation may include a table which identifies each line of output, and any table may be printed more than once.

II-C. DATA INPUTS

The data input to this program consists of a single magnetic tape file of records to be edited. The total number of records to be processed from the file must be specified. Records may appear in any order that is convenient. The tape may be written either in binary or in BCD mode.

The input records must all have a common format but the format can be chosen to fit the problem: the input format is variable. A format specification (part III-B, item 152) must always be pro-

vided in the job control deck. Up to 132 data fields may be specified for an input record.

II-D. PROGRAM LOGIC AND PROCEDURES

The program has three primary phases that are performed in sequence: (1) the job control deck is read and checked, as far as is possible; (2) the records are processed; and (3) the end-of-pass procedures are carried out, including the printing of a job summary and the punching of special summary tables requested by the user (fig. 1). The program will not proceed to record processing until the control deck has passed all checks.

The user is concerned primarily with the second or processing phase. This phase uses the logic of unit record processing; that is, only one record is handled at a time. It is read, processed, and written out before the next record is read. The processing of any given record is independent of any other record and the sequence of records in the input file determines the sequence in which they are processed. The number of consecutive records to be processed from an input file is specified by the user. Any number up to the maximum number in the file may be processed in one program pass. Additional files can be processed only by additional passes with the program, using if necessary the updating procedures described below.

The user also specifies the maximum number of errors that can be recorded in a pass before processing is automatically halted. An unreasonably large number of errors usually signifies some undetected error in the control deck. This control allows the user to check the control deck before processing all the records. If no error is found, processing may be continued by the updating procedure; otherwise, the control deck is corrected and the pass is repeated from the beginning.

The processing of an individual record is controlled entirely by a set of 11 operations. Each operation performs a particular editing function, using the contents of specified data fields and other pertinent information. The operations contained in the program are:

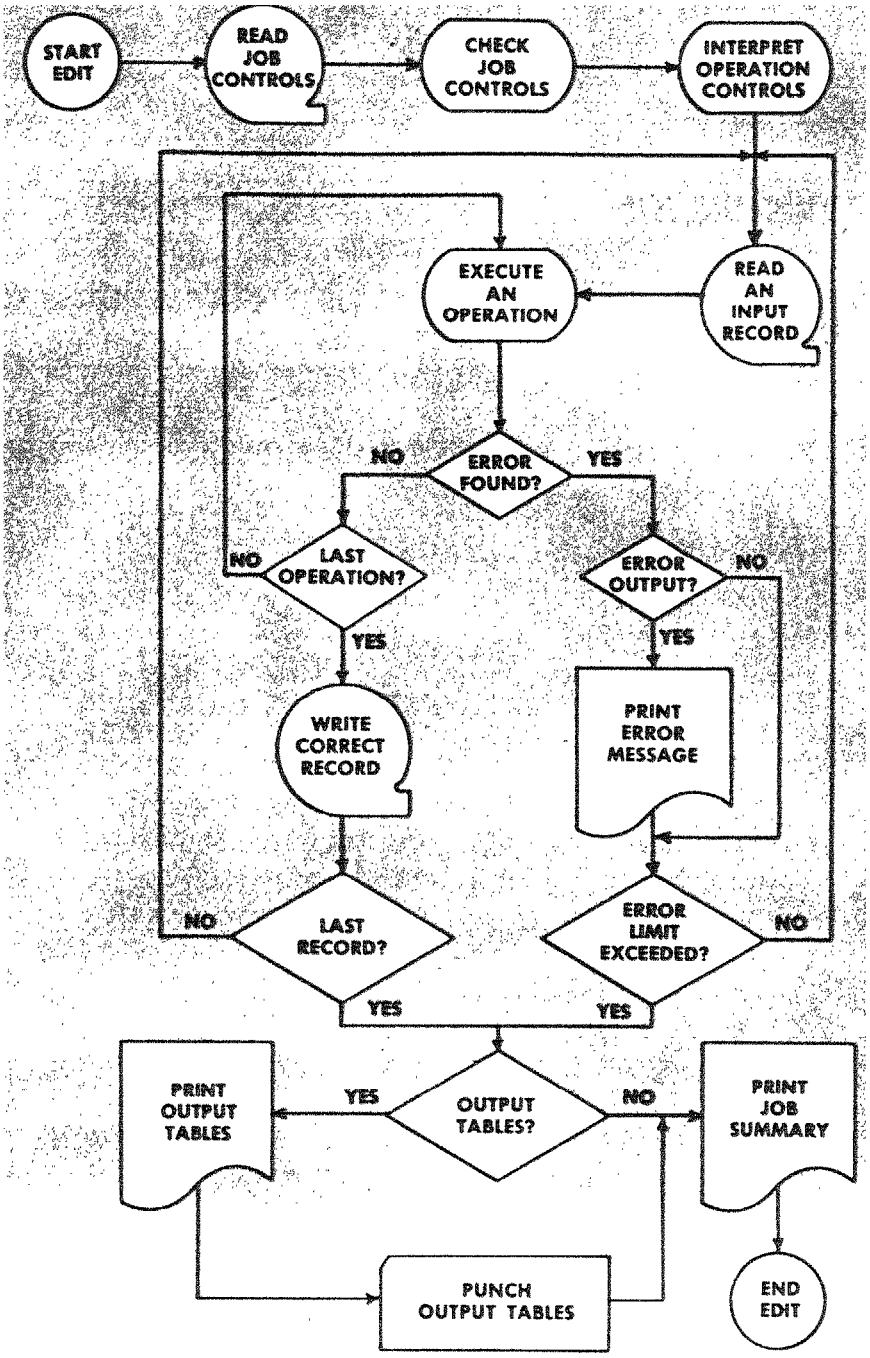


Figure 1. — A generalized flow chart of EDIT.

1. FLOAT. — The operation that converts up to nine values from fixed point to floating point numbers. The operation also allows the floating point numbers to be scaled after conversion, using division by powers of 10.

2. FIX. — The operation that converts up to nine values from floating point to fixed point numbers. The operation also allows the floating point numbers to be scaled before conversion, using multiplication by powers of 10.

3. LIST C (for list check). — The operation that checks a single value against a list of values. An input table gives the list. There are two checking options: the legal option records an error when the value is not found in a list of all values permitted; the illegal option records an error if the value is found in a list containing all values not permitted.

Any data field used in this operation must contain a fixed-point number. The input table must contain as many entries as there are values permitted (or not permitted, if the illegal option is used). Each entry contains just one of the permitted values.

4. CROSSC (for cross check).—The operation that checks the consistency among up to nine values. There are two cheking options: the legal option records an error when the combination of values is not found in an input table containing all combinations of values permitted; the illegal option records an error if the combination of values is found in an input table containing all combinations of values not permitted.

All data fields used in this operation must contain fixed-point numbers. The input table must contain as many entries as there are combinations of values permitted (or not permitted, if the illegal option is used) in the data fields being checked. Each entry of the input table contains one combination of values, in specified order.

For example, assume that the following relationships must exist between the values in the first and the second data fields of a record: if the value in the first field is 1, then the value in the second field must be 1, 2, 3, 4, or 5; and, if the value in the first field is 2, 3, or 4, then the value in the second field must be 2, 3,

4, or 5; and, if the value in the first field is 5 or 6, then the value in the second field must be 6. There are 19 possible combinations of these values, so the input table must contain 19 entries, using the legal checking option. The first digit of each entry is a permitted value of the first field and the second digit is a corresponding value permitted for the second field. The input table entries for this example are: 11, 12, 13, 14, 15, 22, 23, 24, 25, 32, 33, 34, 35, 42, 43, 44, 45, 56, and 66.

Note that the input table entries are arranged so that the values are in ascending order. This arrangement must hold for any input table.

5. CROSSR (for cross range check).—The operation that checks a single value against a range. The range is selected from an input table on the basis of one other value. Actually, two checks are made: an error is recorded if the value used to select the range is not found in the input table (a list check); and, an error is recorded if the value being checked is not equal to or greater than the minimum value and less than or equal to the maximum value of the range.

Both data fields used in this operation must contain fixed-point numbers. The input table used must contain as many entries as there are values permitted for the data field used in selecting the range. Each input table entry consists of three fields: the first contains one of the values used to select the range; the second contains the minimum value of the range; and, the third contains the maximum value of the range.

For example, assume that the following relationships must exist between values in the first and the second data fields of a record: if the value in the first field is 1, then the value in the second field must fall between 1 and 13, inclusive; and, if the value in the first field is 2, then the value in the second field must fall between 18 and 62, inclusive. The input table must contain two entries: 10113 and 21862.

6. SEQUEN (for sequence check). — The operation that checks the sequence of values in a single data field from one record to the next. Either an ascending or a descending sequence may be

checked for a constant increment (or decrement). An error is recorded if the increment or decrement is not constant.

Any data field used in this operation must contain a fixed-point number. The initial value of the sequence and the value of the constant are specified in calling the operation.

7. LOGIC (for logical expression). — The operation that determines the truth value (true or false) of a given logical expression, and executes a branch specified for that value. For the occurrence of a given truth value, one of the following actions is specified: record an error and begin processing the next record; skip a specified number of operations and continue processing the current record. For an occurrence of the alternate truth value, the action is always to execute the next operation in sequence with the current record.

All numeric values used in this operation must be fixed-point numbers. The values can be those in data fields, or they can be constants; up to two constants can be used in one logical expression.

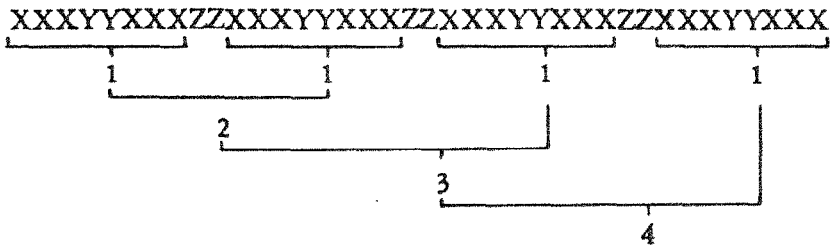
The logical expression is constructed from simple logical statements or comparisons of two values according to one of six relational operators:

EQ = equal to	NE = not equal to
GT = greater than	GE = greater than or equal to
LT = less than	LE = less than or equal to

The logical expression can contain up to four such comparisons, each associated with the others by one of three logical connectives:

AN = and
OR = or
TH = if . . . then

To determine the truth value of the logical expression, the truth values of the simple comparisons are obtained first. These values, and their logical connectives, are then scanned from left to right to obtain the truth value of the entire expression:



where —

XXX = a numeric value
YY = a relational operator
ZZ = a logical connective

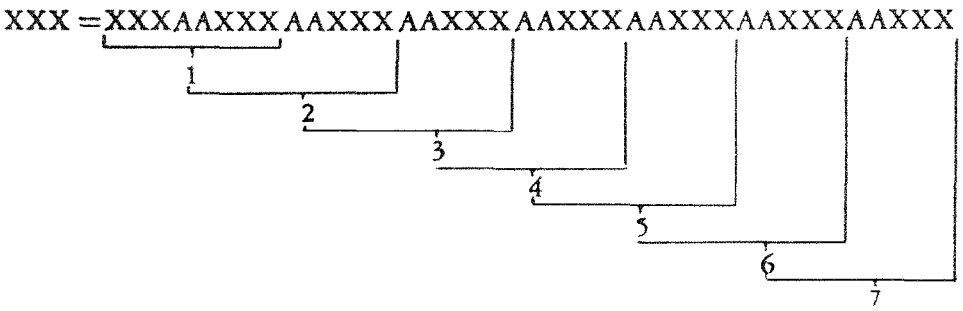
The truth sets for the compound expressions involving statements "p" and "q" are:

P	q	p *AN* q	p *OR* q	p *TH* q
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

8. ARITHE (for arithmetic expression). — The operation that supplies a value for a data field by evaluating a given arithmetic expression containing up to eight other values (including up to two constants), and any of the following arithmetic operators:

\+ = add
— = subtract
* = mulitply
/ = divide
** = raise to the power of

All values used in the operation must be floating-point numbers. The value generated is also a floating-point number. The expression is evaluated from left to right:



where

XXX = a numeric value

AA = an arithmetic operator

9. GENERA (for generate values). — The operation that supplies values for one or more data fields. The values are supplied from an input table according to a combination of other values. This operation also records an error if the combination of values is not found in the input table. If the combination is found, the corresponding generated values are moved from the input table to the record.

All data fields used in this operation must contain fixed-point numbers, and all generated values must be fixed-point numbers. The input table must contain as many entries as there are combinations of permitted values. Each input table entry contains one of these combinations followed by the corresponding generated values.

For example, assume that the permitted values in the first data field are 3, 4, 7, and 9; that when the value in the first field is 3 or 4, a value of 1 is to be generated for the second field; that when the first field contains a 7, the second field is to contain a 2; and that when the first field is 9, the second field must be 3. Since there are 4 values permitted for the first field, the input table must contain these 4 entries: 31, 41, 72, and 93.

10. CALCUL (for calculate values).—The operation that may be specially programmed to perform any unusual or complex calculations required to obtain values for data fields.

Data fields used in this operation may contain either fixed-point

or floating-point numbers. No input tables may be used. The operation is programmed in FORTRAN IV, according to instructions given in part IV-A.

11. ADD. — The operation that forms an output table by summing the value in a data field, or a constant value, into a cell of the output table each time a record is processed. The particular cell in which the value is accumulated is selected by a combination of up to eight or nine other values in the record, using an input table for identification. This operation also records an error if the combination of values used to select the cell is not found in the input table. If the combination is found, a value is summed into the corresponding cell of the output table.

The values used to select the cell in the output table must be fixed-point numbers. The value to be summed into the cell must be a floating-point number. The input table must contain as many entries as there are combinations of permitted values used to select the output table cell. Each input table entry consists of one of these combinations.

* * *

In calling an operation (part III-B, sec. 120), the user gives the operation name, specifies the data fields to be used, and gives the constants or input tables (part III-B, sec. 110) required for the operation. The description of each combination is recorded on an operation-control card. Any operation may be called more than once with different arguments, but the total number of operations per record cannot exceed 100. The operations are performed in the same sequence as the operation cards appear in the control deck. Thus, the user controls the logic of record processing by the way he orders the operation cards. Logical control can be extended by use of the LOGIC operation to cause a branch during the processing of selected records.

The processing of a record will be discontinued by the detection of an error in a record. An error message will be printed and processing will go on to the next input record without completing the operations on the current record. This must be considered in constructing the logic of record processing. For example, in a case

where records are to be checked for correctness and a frequency distribution of correct records is to be formed (by an ADD operation), the checking operations obviously must precede the operation that will accumulate the frequencies. Those applications in which the hierarchy of operations is important will often be complex enough to warrant the use of a flow chart in selecting and sequencing the operations.

To correct records found to be in error and add them to the output tables obtained at the end of the initial pass, provision has been made to punch out each such table (part III-B, sec. 140) in a form that can be added to the original control deck for a subsequent pass of the program with the corrected records. During this pass the corrected records are again checked and those that are correct are added to the tables that have been read in. This updating procedure can also be used with an initial pass to enter non-zero values in tables of accumulation such as might be required in an accounting or an inventory application.

II-E. CONCLUSION

In the preceding chapters program EDIT has been described as a general unit-record processor in which all data-handling functions as well as a comprehensive set of editing operations have been pre-programmed. In a given application, the actual editing of the unit records is programmed to fit the requirements of the application by defining each step in the process in terms of the pre-programmed operations and the format of the input records.

The definition of these steps and the provision of certain other information are always the responsibility of the user. The description of the job control deck in part III can be used as a checklist in assembling the description of any processing problem.

Preparing and checking the job control deck is not easy. The deck contains a great deal of detailed information about the problem, not all of which can be checked by the program prior to test runs. Consequently, while the program offers an efficient means to solve a variety of processing problems involving large amounts of data, some other means will generally be better when only small amounts of data (less than 5,000 records, say) are to be processed.