UNIVERSITY OF QUEENSLAND
COMPUTER CENTRE

COMPUTER CENTRE BULLETIN

Volume 2, Number 6.
2nd June, 1969.

Editor:
H. L. Smythe.

THIS EDITION

This month's Bulletin continues to assist clients and readers with useful programming information about the discovery of further PDP 10 FORTRAN IV errors. An article describes and explains the new FORTRAN coding sheet for the benefit of clients.

A description of the sorting programs available at the Computer Centre should interest readers, and the use of a program to calculate electric lightning performance in Canada shows that international use is being made of Computer Centre facilities.

Library accessions are listed, staff changes are described, and some recent publications are noted.

STAFF OF THE COMPUTER CENTRE

In this month's Staff article, we extend a very warm welcome to four new members of staff, three Computer Operators and a Data Preparation Assistant. At the same time, however, we regretfully say farewell to some old, familiar faces, and wish them every success and happiness in their new fields.

INTRODUCING ....

Dianne Ball, Colleen Kelly and *Dianne Neill* have recently been appointed Machine Operators. All three girls have had experience in card-punching; Dianne Ball, in fact, was formerly a Data Preparation Assistant for the Computer Centre.

The new Machine Operators undergo a month's training before they can operate the computers. During this period, they receive lectures from Graham Jerrard and Jim Sokoll on the mechanics and operating procedures of the machines. After about four months' practical experience in operating, they are classified officially as Computer Operators.

To strengthen forces in the Data Preparation Room, *Heather Murray* has been appointed as Data Preparation Assistant. Both Heather and Angela Vidanovic operate two IBM 029 Card Punches and an IBM 059 Card Verifier.

FAREWELLING ....

Ian Oliver, a Lecturer in Computing, has left the Department to begin practice as a Consultant. Ian joined the Computer Centre in January 1963 as a Programmer, and, in 1967, was appointed Senior Programmer at Ohio State University. In 1968 and 1969, he was a Lecturer in Computing in the Department, and successfully combined the teaching duties of the postgraduate Diploma in Automatic Computing with the programming activities associated with the implementation of the PDP 10 Operating System. As Ian was one of the earliest staff members, we are very sorry to see him leave, but wish him every good fortune in his new venture.

Bill Fulton, a Programmer with the Computer Centre since February 1968, has left to join Digital Equipment Australia Pty. Ltd. As a member of their Software Support Team, he will spend about six months at Maynard, Massachusetts. His special task is to write programs for the Melbourne Stock Exchange Contract. We wish him much success in a very promising career.

We also say good-bye to *Mrs. Pat Matthews* who has been a Machine Operator at the Computer Centre since November 1966. To Pat go our sincere wishes for a life filled with happiness.

PDP 10 FORTRAN IV ERRORS

C.C. de Voil
J.S. Williams

The Bulletin continues to inform readers of any errors discovered in PDP 10 FORTRAN IV. Should clients have any difficulties, programmers are available for consultation at specified times.

It would be greatly appreciated if users who feel they have discovered an error, report it to the Administrative Officer (Mr. John Jauncey, extension 8471), together with supporting evidence such as card decks and listings.

Other users may then be notified and the error fixed.

1. During compilation, the BATCH program which passes input data to the actual FORTRAN compiler, deletes trailing blanks from input records.

   This will result in a reduction of the charge for input, but where these blanks form part of a Hollerith field in a format statement, unfortunately they are not regenerated by the compiler. Thus the format statement will probably be adversely affected. For example, a format statement of the form:

```
10  FORMAT(20X,'HEADING'/'A          B          C
   1D       E       F')
```

   will not yield the desired result, and one of the form:

```
10  FORMAT(20X/90H A         B          C
   1D       E    F)
```

   will yield a diagnostic error.

   This can be remedied by either rephrasing the format statement to remove significant trailing blanks, or by suppressing this feature by inserting a sequence number on the offending cards.

2. The function ATAN returns an incorrect value (0.0) for an argument of $\frac{\pi}{2}$. For other than this limit value, it appears to work correctly.

   The function DATAN returns values that have significant errors.

   These two problems will be remedied as soon as possible.

3. The compiler fails internally to pass on the sign change when it generates a negated subexpression store into a temporary location. For example:

```
A = SIN(X/Y) + SIN(-X/Y)
```

   This problem can be avoided by changing the expression order to force a positive store.

4. This section in last month's Bulletin mentioned a point of inconsistency viz: as a result of an error in a DIMENSION statement, none of the arrays in the statement is defined as an array, and compilation error messages are not produced for statements that have an array on the left hand side of an equation.

The situation has been improved so that the compilation error message is omitted only if all of the following points are true:

(i) the array appears on both sides of the equation;

(ii) at least one occurrence of the array on the right hand side has a subscript in common with the left hand side use of the array;

(iii) the subscript that is common has not been defined at this point in the program.

For example, the following statement would not produce an error message (provided M has not been defined prior to the occurrence of this statement):

```
A(M) = A(M) + 1.0
```

5. The section on *Programming Advice* in last month's Bulletin included a misprint in the sixth suggestion given to convert from GE 225 to PDP 10 FORTRAN IV. The following statements are now correct:

> Do any FORMAT statements use X, rather than 1 X, to omit one character position?
>
> In PDP 10 FORTRAN IV, X is not interpreted as 1 X. It produces a compilation error message.

LIBRARY ACCESSIONS

This section lists the books and periodicals concerning the computer field, that have been acquired by Libraries of the University of Queensland in February 1969.

Lee, T.H. *Computer Process Control.* 1968. (658.507 LEE, Elect.Engin.Lib.)

Li, David Hsiang-fu. *Accounting Computers, Management Information Systems.* 1968. (657.018 LI, Main Lib.)

Tucker, Dorothy *Computers and Information Systems in Planning and Related Governmental Functions.* 1968. (Q651.8 TUC, Architecture Lib.)

International Federation for Information Processing. Congress. 1965. Vol. 1 to date.

Laden, Hyman N. and Gildersleeve, T.R. *System Design for Computer Applications.* 1967. (651.8 LAD, Engin.Lib.)

Martin, James T. *Design of Real-Time Computer Systems.* 1967. (651.8 MAR, Engin.Lib.)

Mills, Arthur E. *The Dynamics of Management Control Systems.* 1967. (658 MIL, Engin.Lib.)

Sellers, F.W. and Bearnson, Leroy W. and Hsiao, Mu. *Error-Detecting Logic for Digital Computers.* 1968. (651.8 SEL, Engin.Lib.)

Sherman, Phillip M. *Programming and Coding Digital Computers.* 1963. (510.7834 SHE, Engin.Lib.)

Weinberg, Gerald M. *PL/I Programming Primer.* 1966. (651.8 WEI, Engin.Lib.)

RECENT PUBLICATIONS

The following publications are available at the Computer Centre:

TECHNICAL MANUAL NO. 1. Supplement A.

Additional Features of PDP 10 FORTRAN IV

R. E. Kelly

This supplement describes those additional features of the PDP 10 FORTRAN IV language that are available. However, as many of these facilities are non-standard, programs using them may not be capable of being simply transferred to another computer system.

TECHNICAL MANUAL NO. 1. Supplement B.

Operating Procedures for PDP 10 FORTRAN IV

R. E. Kelly

This supplement describes in detail the control cards used for batch processing work, the different ways in which card decks are assembled to run various types of jobs, and it also details compiler error messages.

PROGRAMMING ADVICE

NEW FORTRAN IV CODING SHEET

J. S. Williams

The FORTRAN coding sheet, produced by the Computer Centre, has been redesigned to eliminate the deficiencies of the old coding sheet, viz:

(a) the problem of choosing the correct character size and spacing to fit 66 characters (to be punched in columns 7 to 72) of a long statement on one line of the sheet.

(b) the problem of showing the number of spaces that are to be included in a Hollerith constant or Hollerith FORMAT specification.

(c) the tedious task of filling in the sequence numbers.

The main features of the new coding sheet (see Figure 1) are as follows:

(a) *Project Number* - the programmer must insert the project number assigned to him for work on the PDP 10. If the programmer is using the GE 225, his Department/Customer/Charge-to number is placed here.

(b) *Character Representation* - some generally accepted character representations are difficult to distinguish when written quickly, and since the computer cannot tell from the context which character is intended, accurate card punching is essential. The Computer Centre has, therefore, adopted a convention for representing these characters, and this is provided on the new coding sheet.

(c) *Character position* - each line of the sheet is divided into 80 "squares", each corresponding to a card column. Only one character is to be placed in any given square. Each character is punched in the card column corresponding to its position on the line of the coding sheet.

(d) *Program Identification Code* - a four character program identification code can be placed in the 73rd to 76th position of the first line. These will *automatically* be reproduced on all subsequent cards punched from *that coding sheet*. If these squares are left blank on a subsequent sheet(s), then the corresponding field on the cards punched from that sheet(s) will also be left blank. Identification codes can be inserted in the shaded area, if desired. (This would be necessary when submitting a page of corrections and/or insertions for several different programs).

(e) *Page and Card Number* - if the number of the page (coding sheet) is placed in squares 77 to 78 of line one (numbering from zero), the cards will *automatically* be sequenced, e.g.

0000, 0005, 0010, 0015, etc.

Incrementations of the sequence number by 5 allow for cards to be inserted with intermediate sequence numbers. Page numbers can be written in the shaded area, and card numbers can be crossed out and the new one placed in the margin. (This would be necessary when submitting a page of corrections and/or insertions for a program(s)).

These new FORTRAN coding sheets are available in pads of 50 sheets, on request, from the Administrative Officer (Mr. J. Jauncey, extension 8471).

TRANSMISSION LINE LIGHTNING PERFORMANCE CALCULATIONS

Dr. M. Darveniza

This article is the first of what we hope will become a regular or semi-regular series about the varied uses that clients make of the computers.

UNIVERSITY OF QUEENSLAND
COMPUTER CENTRE

FORTRAN CODING SHEET

PROJECT NUMBER	PROGRAMMER	PROGRAM	DATE

0 or ϕ = ZERO
Θ = ALPHA O

I or 1 = ONE
I = ALPHA I
/ = SLASH

2 = TWO
Ƶ = ALPHA Z

5 = FIVE
S = ALPHA S

COMMENT | STATEMENT NUMBER | CONTINUATION | FORTRAN STATEMENT | PROGRAM IDENTIFICATION CODE | PAGE NUMBER | CARD NUMBER

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

	Card number
1	0 0
2	0 5
3	1 0
4	1 5
5	2 0
6	2 5
7	3 0
8	3 5
9	4 0
10	4 5
11	5 0
12	5 5
13	6 0
14	6 5
15	7 0
16	7 5
17	8 0
18	8 5
19	9 0
20	9 5

Figure 1

In this article, Dr. M. Darveniza, Reader in Electrical Engineering at the University of Queensland, describes lightning performance calculations carried out on behalf of the Hydro-electric Power Commission of Ontario, Canada.

The Editor would be pleased to publish articles describing individual or group projects relating to some aspect of computing. Articles of up to 2,000 words may be printed. In this way, it is hoped to broaden and diversify the scope of the Bulletin.

The Department of Electrical Engineering at the University of Queensland has recently completed a detailed study of the lightning performance of a major 220 kV double-circuit (d-c) transmission line being constructed by the Hydro-electric Power Commission of Ontario, Canada.

The line will provide the first link between Ontario Hydro's East supply system and its West system on the Manitoba side - hence, is referred to as the East-West tie. The line is situated along the foreshores of the Great Lakes, and runs between Algoma on the North shore of Lake Huron via Mississagai, Marathon on Lake Superior, to Atikokan. The total length of the line is over 400 miles, and its capital cost is about 60 million Canadian dollars.

The lightning activity in the region traversed by the line is not particularly high (20 to 30 thunderdays per year). However, the terrain is rough and the soil resistivity is high, and hence high tower footing resistances are often unavoidable. Despite the widespread use of continuous counterpoises, the lightning outage rates of existing 115 kV d-c lines in this area are high, typical values being about 9 outages per 100 miles per year. Further, and more importantly from a system point-of-view, some 2/3 of these outages involve simultaneous faults on both circuits i.e. d-c faults.

Because the East-West tie line provides the only transmission link between Ontario Hydro's two systems, it was necessary to design for a high degree of reliability against simultaneous double circuit outages. Whereas a fault causing a single circuit outage might be tolerated, double circuit faults would disrupt the tie, thus depriving the West system of the support from the more extensive East system.

In late 1967, the University was approached by Ontario Hydro to undertake the lightning performance calculations for the East-West tie line, with particular reference to the prediction of the likelihood of double circuit outages. A computer program developed by *Drs. M. Darveniza* and *M.A. Sargent* was available for the study. The validity of its application to the Ontario-Hydro system was firstly checked by carrying out calculations on an existing 115 kV line in the same area, for which detailed outage statistics were available from 7 years of service. The agreement between calculated and observed outage rates was satisfactory, particularly in regard to the prediction of double-circuit faults.

The computer program was then used for the lightning performance studies on the 230 kV East-West tie line. At the time the calculations were started, a conventional tower design had been chosen, and much of the line was already under construction. However, the Research and Planning Departments of Ontario-Hydro were considering various design proposals aimed at increasing the security of the line against simultaneous d-c faults due to lightning, and detailed studies of their likely influence were required. The studies carried out at the University included an investigation into the need for providing continuous counterpoises, and an evaluation of the use of differential insulation (14 insulators on one circuit and 10 on the other). The results showed that counterpoise earths were essential, and that differential insulation would achieve only a marginal improvement in the double-circuit outage rate, and this at the expense of increasing the single circuit outage rate. Studies were also made of a novel double-triangular circuit configuration which could be mounted on the existing conventional towers with little modification, and hence negligible extra cost. In this arrangement, the conductors of circuit 1 form a triangle with two conductors on the top crossarm and one on the centre arm, and circuit 2 conductors are placed on the other centre arm position and on thetwo bottom arm positions. The calculations predicted a substantial improvement in the double-circuit outage rate for this arrangement, and Ontario-Hydro have decided to use it on major sections of the tie-line. As far as is known, this is the first occasion that this arrangement has been applied to a major transmission line.

Details of the lightning prediction method have been described in the technical literature. The method utilises a digital computer program for calculating the voltages stressing line insulation under lightning conditions, and incorporates a Monte-Carlo technique for randomly manipulating the complex variables associated with lightning strikes to transmission lines. At present, the program is processed on the Computer Centre's GE 225 computer and utilises a core memory of 8K. The computer time for carrying out the extensive calculations requested for the Ontario-Hydro East-West tie line amounted to 23 hours. It is anticipated that the computation time will be greatly reduced (by at least a tenth) with the availability of the new PDP 10 computer system.

It may be of interest to note that the Computer Centre's lightning prediction program has been utilised by other major supply authorities in Australia and North America, including:

Northern Electric Authority, Queensland	(132 kV)
Southern Electric Authority of Queensland	(110 & 275 kV)
Electricity Commission of New South Wales	(132, 330 kV)
State Electricity Commission of Victoria	(220 kV)
Darwin Electricity Supply Undertaking	(132 kV)
Tennessee Valley Authority, U.S.A.	(161 kV)

AN INTRODUCTION TO SORTING

R. N. Buchanan
E. J. Sokoll

In simple terms, the term *sorting* means the arranging of records of information into order; this order is determined by the information contained in each record. A typical record would be the information on a punched card. Each of the discrete pieces of information on a record is called a *field*. On a punched card, a field would be the contents of a group of columns.

Suppose the first field of a record contains a person's surname, and the remainder of the record contains information about this person. It is desired to arrange the records so that names are in alphabetical order. The order of records is determined by the field containing the name; this field which determines the order is called the *key*. This, briefly, is an example of sorting. The need to sort is obvious - imagine how useless a telephone directory would be if the names were not sorted!

The sorting methods which have been employed are as numerous as the reasons for sorting. Two important considerations influencing the choice of a sorting method are its speed and its storage requirements. For a given number of records, some methods require more storage than others. All require an area for the original file, but some require, as well, a great amount of temporary storage. This will restrict the number of records which can be sorted *internally*, i.e. sorted entirely in memory without the necessity of using magnetic tape.

However, the number of records to be sorted is often so large that all the records cannot be sorted in memory at the one time, no matter which method is used. In this case, as many records as possible are read into memory and are sorted internally. This sorted string of records is written onto *magnetic tape*. Similar strings are produced and written on tape, and are merged to produce longer strings. Finally, a complete sorted string containing all the records is produced on tape.

A comparison of times for two sorting methods, the *Exchange Sort* and the *Sort-Merge-Sift*, may be of interest. The Exchange Sort is slow but simple to program; the Sort-Merge-Sift is one of the fastest methods. The sorting time is proportional to the square of n for the Exchange method, and approximately proportional to $n^{1.4}$ for the Sort-Merge-Sift method where n is the number of elements in the file. The observed times for sorting 2000 integers were 25 mins. 12 secs., and 22 seconds respectively. These figures apply to programs run on the GE 225. The times taken to sort 1000 single length integers were 325 secs. and 9 secs. respectively. Identical programs on the PDP 10 are not yet available, but the times are expected to be much shorter.

The Sort-Merge-Sift is a merging method adapted to internal sorting. This method will be explained for the special case when 16 records are to be sorted. The first pass produces 8 ordered pairs. The 1st and 9th records are compared, and if they are in the wrong order, they are exchanged. The 2nd and 10th, 3rd and 11th, ..., 8th and 16th records are treated in the same way.

Records which are linked form a sorted string as shown in Figure 1.

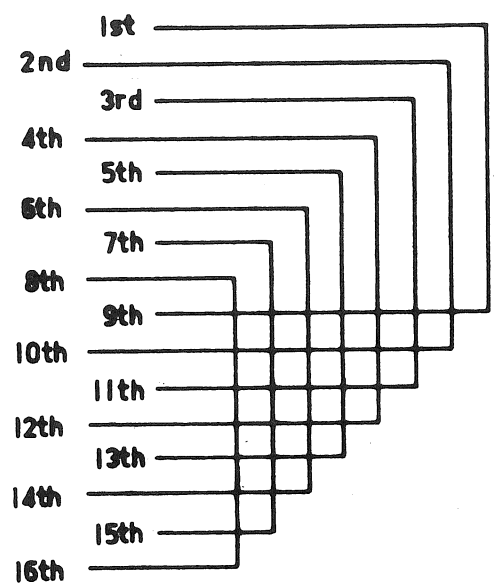

Figure 1

The 2nd pass merges these ordered pairs, and produces four strings, each consisting of 4 records as in Figure 2.

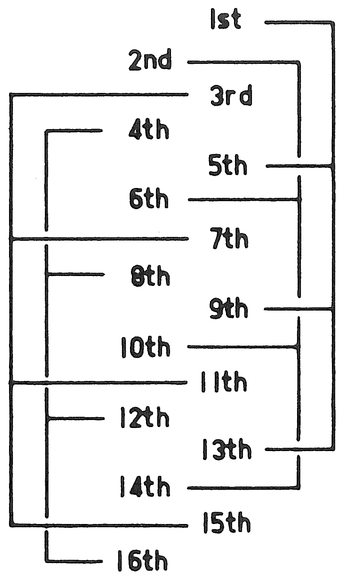

Figure 2

The third pass produces 2 strings, each consisting of 8 records as in Figure 3.

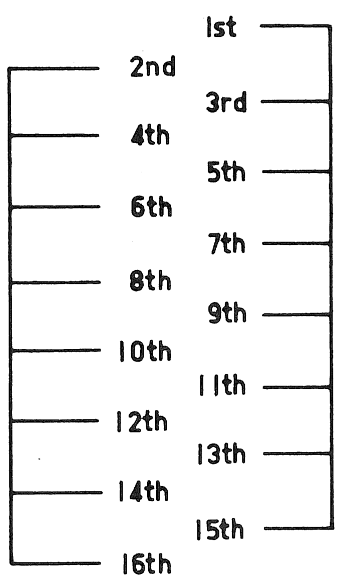

Figure 3

After the 4th pass, all the records are in order.

In each pass, comparisons and exchanges take place to produce sorted strings of increased length. In general, the number of records to be sorted will not be a power of 2, but the method can be extended easily to handle this situation.

Moving large records during sorting can be time-consuming. This can be avoided, however, by identifying each record by an index and exchanging indices instead of records.

Two standard subroutines are available for internal sorting - the GAP subroutine SØRT and FØRTRAN SØRT. Both use the Sort-Merge-Sift method combined with the indexing technique mentioned above. Keys may be integers or alphanumeric information, and may be in any position in the record. SØRT is on the subroutine library tape, and FØRTRAN SØRT is in the FØRTRAN Suffix Deck. Write-ups can be obtained at the Computer Centre.

Also available is a program called Forward Sort/Merge Generator which will generate a sorting (or merging) program from the user's own specifications punched on parameter cards. Preparation of the parameter cards requires only an understanding of the files to be processed and the results that are desired. The user does not have to be acquainted with the complex logic of sort and merge programs.

For a minimum application only, 4 parameter cards need be submitted to the generator. The following are sample parameter cards:

```
SRT 8  027    5    D           SAMPLESRT
INPUT  Y N 020 15              TESTDATA 150765
ØUTPUT  Y N 020N               SØRTDDATA250765
SØRT                           12,13,14
```

These are now explained in detail.

The first parameter card is called a Program Card. SRT signifies that a Sort Program is to be generated (in contrast to a Merge); 8 means that the computer to be used has a memory of 8192 words; the record size is 27 words; 5 words are to be used for keys; D indicates decimal information; SAMPLESRT is the program name.

Y in the Input Card indicates that FØRWARD's input subroutine is to be used; the following N means that the user is not providing his own input routine; 20 is the blocking factor - the number of logical records (27 words, in this case) to be grouped together in one physical record on tape; the input tape is to be on plug 1 handler 5; the Beginning Tape Label on the input tape is TESTDATA; 150765 is the date.

In the Output Card, Y indicates that FØRWARD's output subroutine is to be used; N means the user is not providing his own output routine; 20 is the blocking factor; and SØRTDDATA is the Beginning Tape Label of the output tape.

The Sort Card gives the plug numbers and handler addresses of the collation tapes, i.e., the additional tapes required in the sorting process.

Using these parameter cards, the generator would take about 2 minutes to write a complete 2500 instruction Sort Program. Hence weeks of programming effort and much computer time are saved.

Suppose 10,000 cards are to be sorted on 30 alphanumeric columns. If an 800 card per minute card sorter were used, 2 passes would be necessary for each of the 30 columns. This would take 12 hours 30 minutes.

Alternatively, Forward Sort/Merge could be used to generate a program to do the job. Initially, the information from the cards is written onto magnetic tape, taking about 35 minutes. Sorting the information would occupy approximately 15 minutes, making 50 minutes in all.

The graph contained in Figure 4 gives the times taken by Forward Sort Programs with various numbers of records, record lengths and tape handlers.

ESTIMATED FORWARD SORT TIME

8192 word memory; 41.7 Kc magnetic tape; 1 controller;

25, 50, 75 and 100 word BCD records; 5 word key

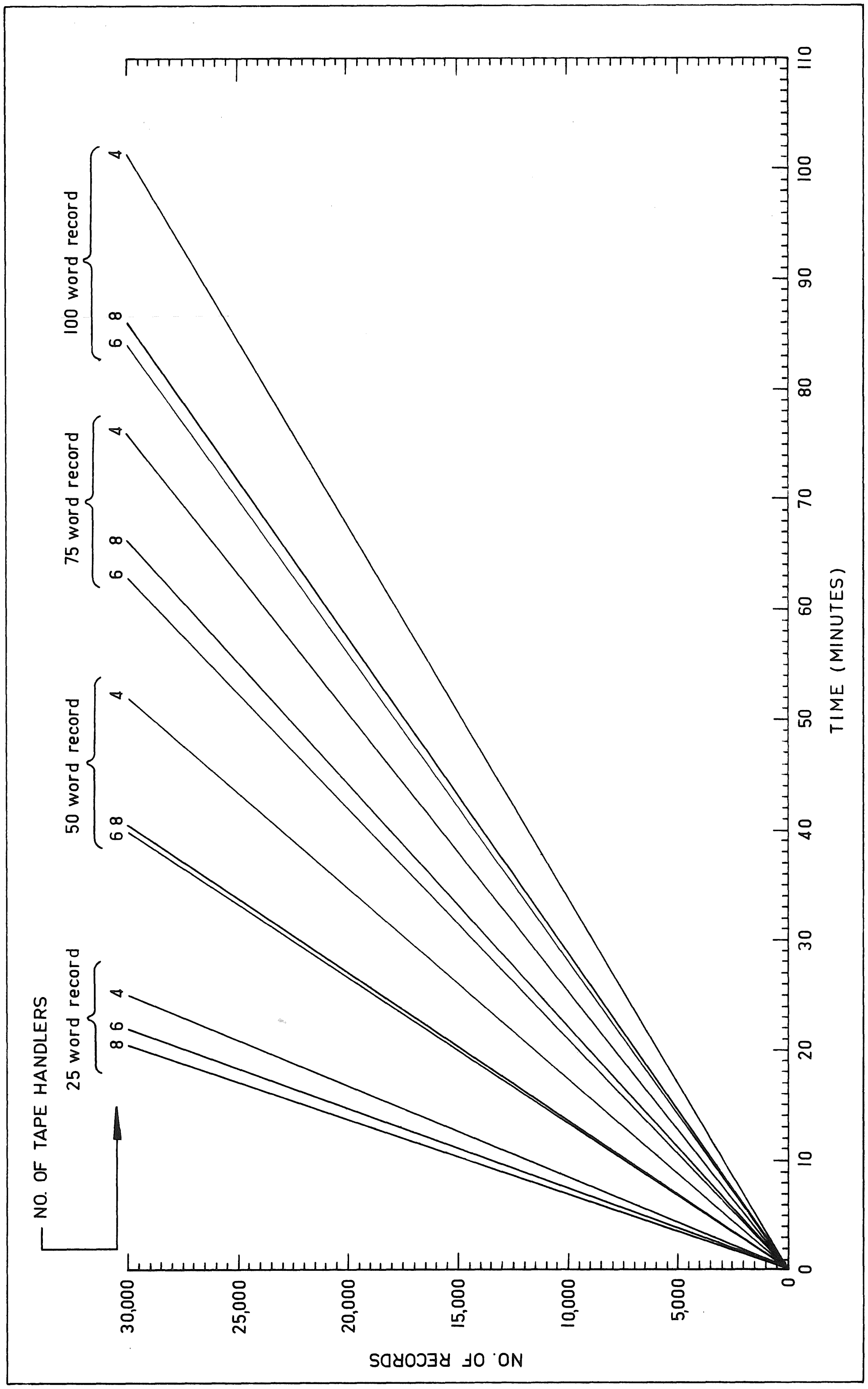

Figure 4

With Forward Sort/Merge Generator, the user may include any unusual features in the program by inserting sections of GAP coding of his own with the parameter cards.

While sorting is a fascinating subject, it is not necessary for users to concern themselves with its logic. The programs available at the Computer Centre should satisfy all sorting requirements.