Z77-3014

TECHNICAL INFORMATION EXCHANGE

A SURVEY OF SORTING TECHNIQUES

Miss Marilyn M. Jensen
IBM Corporation
3223 Wilshire Boulevard
Santa Monica, California

IBM Corporation, Technical Publications Dept., 112 E. Post Road, White Plains, N.Y. 10601

IBM®

July 18, 1963

A paper concerning the range of sorting techniques available for the Phase I internal sort. The discussion encompasses straight and von Neumann merging; simple, centered, and binary inserting; exchanging two methods of radix sorting; counting; address calculating; linear, quadratic, cubic, n-degree, and replacement selecting; and tree and forest sorting. Advantages and disadvantages of each method are evaluated, with examples of existing programs of implementation furnished.

Z77-3014

TABLE OF CONTENTS

Introduction

A number of internal sorting techniques will be presented in an elementary manner. Hopefully, the reader may gain an appreciation of the diversity and growth of sorting techniques that have been developed. This paper is, therefore, offered as a supplement to "Sorting Methods for IBM Data Processing Systems" for the neophyte, and as a bibliographical guide to the experienced Systems Engineer in search of more detailed and specific information.

Once an appreciation and knowledge of the magnitude of alternative sorting techniques is realized, the Systems Engineer will be able to more effectively guide and counsel the customer in making modifications of existing sorting programs, or designing new ones.

A program for sorting a file stored on one or more magnetic tape reels can be divided into three separate and distinct phases. Phase 1 accomplishes the original distribution and internal sort. The group of input records are sorted internally in some manner to produce a single block of sequenced records. Phase 2 consists of a merge. The groups of records on one tape, produced by Phase 1, are merged to produce a single tape reel with all of the records in the requested sequence. Phase 3, the final merging, is a collating run which ultimately produces the single sequenced file.

This paper will limit its concern to a survey of the techniques available for the Phase 1 internal sort. The discussion will encompass straight and von Neumann merging; simple, centered, and binary inserting; exchanging; two methods of radix sorting; counting; address calculating; linear, quadratic, cubic, n-degree, and replacement selecting; and tree and forest sorting. Advantages and disadvantages of each method are evaluated, with examples of existing programs of implementation furnished.

One notable exception to the following discussion of internal sorting techniques is the AMOR sort, for it is applicable only to associative memory devices.

Straight and von Neumann Merging

Merging has been defined as the producing of a single sequence of items, ordered according to some rule (that is, arranged in some predetermined sequence) from two or more sequences previously ordered according to the same rule, without changing the items in size, structure, or total number.

Two methods of internal merging are used in sorting: the straight internal merge and the von Neumann, or natural, merge. The straight internal merge uses fixed length strings. Alternating between

two work areas, the merge will build up sequenced strings of 2, 4, 8, 16, ... n records until the last merge produces the sequenced file (i.e. a single string). The leading item of each string is compared, and the lowest placed in the output area and the next sequential record of the string furnishing the output record is analyzed. This record is compared with the other string, the lowest being placed in the output area. The procedure is continued until both strings are exhausted. When exhaustion occurs, two new strings are entered into the work area to be merged.

The von Neumann, or natural, merge takes full advantage of all ordering existing in the original file. The strings are not restricted to a fixed length, as in the straight internal merge, and the number of passes required to sort a given file is dependent upon the number of strings that exist in the file. For a file in correct sequence one pass is required to check for the desired ascending (descending) sequence.

The principle of the von Neumann merge is quite simple. Two strings are merged into the receiving area until one string suffers a step-down (i.e. a descending (ascending) break in an ascending (descending) ordering). At this point, the sort will commence extracting from the step-down string, and pull from the remaining (second) string until it, too, suffers a step-down. This condition is termed the "double-step-down" and will terminate the formation of the string. The formulated string will then be written as output, and two more strings brought in and processed in a like manner. Sort 90 for the IBM 7070 uses this method for sequencing files.

Both types of merge are very fast. The number of passes required to sort by merging is related to the number of natural sequences existing in the file, and not the number of items. But, conversely, the number of items of input does have a direct effect upon the number of operations that are required to complete a pass. Concerning the von Neumann method, difficulties arise because both initiating areas or streams of input must be monitored for identification of step-down records. Also, the receiving areas must be large to accommodate variable string length. And, if the total number of strings of input data is more than half the number of records in the file, the von Neumann merge saves no passes over a straight internal merge.

Sort 90 for the IBM 7070 uses merging in each of the three stages of the sort program.

Simple, Centered, and Binary Inserting

Insertion is analagous to the method many people use to arrange their bridge hand. They will pick up two cards, sequence them, and pick up the third. This card is then inserted in the proper slot while cards one and two are eased to the left or right to make room for the third. The procedure is continued until all thirteen cards are arranged in the player's hand.

The following methods of sorting utilizing this principle will be discussed: Simple Insertion, Centered Insertion, and Binary Insertion.

Simple insertion consists of building up the final sequenced file one record at a time by determining where in a partial file each record should be inserted, and moving all succeeding records to make room for it. Sort III for the IBM 650 uses insertion for the internal sort and follows with merging onto tapes.

Centered insertion initiates the sequencing by checking the center of the partially constructed file. If the control word is higher, the search continues in the upper half of the file until a lower control work is detected. The old records are moved up to provide for insertion; the new record is inserted, and the next control word to be sequenced is analyzed. The technique of central insertion is superior to simple insertion, for only one half of the table need by searched to sequence.

Binary insertion inspects the control word of the record nearest the middle of the partial file. If the test indicates the location is in the first half of the partial file, the control word of the record nearest the quarter point of the partial file is next inspected. This method reduces considerably the number of control words that must be inspected if the total number of records is large. The insertion of the new record into the partial file is identical to centered insertion.

The principle of binary insertion can be applied to a binary search for the 1311 without the scan feature. In most cases, the binary search will be completed in a shorter length of time than by scanning.

The major disadvantage of binary insertion is the size of program. Evaluating the insertion techniques as a group, the degree of entropy (i.e. lack of organization) of the original file is quite significant. A low degree of entropy will not reduce the number of searches required, but the amount of record movement will be proportional to the degree of entropy. The number of record movements is roughly proportional to the square of the total number of records in the file. And finally, the record length is important because of the character transfer rate involved in the number of records to be moved.

Exchanging

The technique of exchanging to accomplish sorting is executed in the following manner. A comparison is made between the first two control words, and they are exchanged if out of sequence. Comparison is then made of the second and third control words and exchanged if necessary. Each adjoining pair of records in the file is successively listed and exchanged if out of sequence. The process is continued until the file is in sequence, which is signalled by a pass that requires no exchange.

Counting

Counting as a method of sorting is accomplished in the following manner. Each record is read into memory. After it has been read, its control field is compared with the control fields of each of the other records already in memory. For each comparison in which the new record is equal to or greater than the other, an accumulator which will be associated with the new record is increased by one. For each comparison in which the new record is less than the other, the accumulator associated with the other record is increased by one. After the last record has been processed, each accumulator contains the total number of records lower than the record associated with the accumulator. Subsequent pass(es) would be required to call out and arrange the records into the desired output sequence.

Counting is most effective when processing a small file.

Address Calculating

Address calculation is most feasible in a random access system. For each item to be sorted, the location in the file is determined by a linear sort equation of the form $y = a + bx$.[1] The record is place there if the location is empty. If another record is there, a search is made to find the closest empty storage area to the calculated address. The record at the calculated address, together with any (and all) contiguous records are moved so that the record to be filed may be entered in the appropriate sequence into the file.

The address calculation method of sorting places records directly into their proper relative position within the file, and the entire file is in correct relative order just after the last record is inserted. In addition, the file requires only simple processing (i.e. packing) before being transferred to output devices. This is probably the fastest method of sorting a medium size file of small to medium size records. Finally, the programming is quite simple, taking about two hundred program steps to sort.

The disadvantages are linked to the amount of storage available. If the density of the file exceeds the forecast, a great deal of execution time is spent in re-allocating and re-distributing the partial file. Estimates of internal file capacity required range from 20-150% in excess of the file to be sorted to produce an efficient sorting function.

Linear, Quadratic, Cubic, n-degree, and Replacement Selecting

A number of variations are grouped under the general method called selecting: linear, quadratic, cubic, n-degree, and replacement selecting.

---

[1] See "Address Generation for the IBM 1405" TIE Order Number 6207-0002 for a detailed description of the analysis required to develop the linear functions.

The number of passes required to complete the sort is equal to the distance (which is measured by the number of records) which separates the record the greatest distance in a higher order position from its final position in the lower order. Exchanging takes advantage of any lack of entropy in the original sequence, and timing is dependent upon the expected number of passes and the expected number of exchanges.

Radix Sorting

The term "radix" is derived from the fact that this method of sorting requires as many storage areas as the radix (base) of the numbering system used. Two methods of Radix sorting will be discussed: Binary Radix operating on the base of two, and Digital Radix, operating on the base of ten.

Digital Radix Sorting Method 1, is just like punched card sorting. Each item (or record) is dispersed to one of ten storage areas or bins, representing the digits 0-9. The number of passes depends upon the number of control word digits. This is an exceptionally fast method of sorting, but the amount of memory required is prohibitive.

Digital Radix Sorting, Method 2, is identical in operation to Method 1 with the following exceptions. The first pass scans the low order digit of the control field, and makes a distribution count for storage allocation. The second pass sorts the record by the low order digit of the control field into its assigned bin and scans the next higher order digit for the distribution count. At the end of the pass, the storage bins are re-allocated, and the sorting procedure is represented until the last digit is sorted. The number of passes required to sort using this method is equal to $(N + 1)$, when (N) is equal to the number of digits contained in the control field.

Binary Radix sorting is the method employed by the IBM 709 and 7090. The concept is similar to the Digital Radix Method in that two areas are set aside for 0 and 1. These areas are contiguous in storage, thus giving the required two areas for each record, but the length of only one file rather than two. (By making the area continuous, only one file length is required since the data can be only 0 or 1.) The method of sorting is somewhat analogous to block sorting or punched card equipment, for the zero and one quantities are manipulated within the internal file range.

The length of the control word is important in evaluating this sorting method, for the number of passes is dependent upon it. The degree of entropy of the original file has no significance, because the sequence is lost after the first pass. The number of records in the file has importance too, for the number of records which must be moved is proportional to the products of the number of records in the file and the number of digits contained in the control word.

To accomplish linear selection, the record with the lowest control word is selected and placed in the first record storage position by the first pass. The second pass operates in the same manner on the second (n) record and places the second (n) lowest in the second (n) slot. The storage area is compared in turn with each of the remaining records, and if the control field of any record is low, the storage area is replaced. That is, the storage area is replaced only when the control field of the new record is lower than the control field of all previous records.

At the end of each pass, the lowest control field and record of that pass is in the storage area. It is then transmitted to the output area, and cancelled on the input tape in order to insure that it will not be selected again. The process is repeated until an output group of a convenient size is developed at which time it is written, and the process continued until the last record has been sorted and written.

Linear selecting requires a considerable amount of record movement. For example, a file of five hundred records in reverse sequence would require 124,750 interchanges to complete the sort. And there is no way to take advantage of any original ordering.

Quadratic selecting has, as its first step, a counting of the n input records. The n input records are then split into $\sqrt{n}$ sections of $\sqrt{n}$ records or to the nearest integer. Linear selection is then applied to each of the $\sqrt{n}$ sections. $\sqrt{n}$ will contain the control field and address of the record of the data record lowest in its section. Linear selection is then applied to the $\sqrt{n}$ areas, with a quadratic storage area receiving the lowest control field and accompaning address or record. This method saves considerable time in selecting subsequent records since an initial linear pass need be made only over that one of the $\sqrt{n}$ sections which contributed the last low item to the quadratic output area.

The Minneapolis Honeywell 800 Sort utilizes the quadratic selection method for its internal sorting.

Cubic and N degree selecting employ the same techniques as linear and quadratic selecting, and only the power changes.

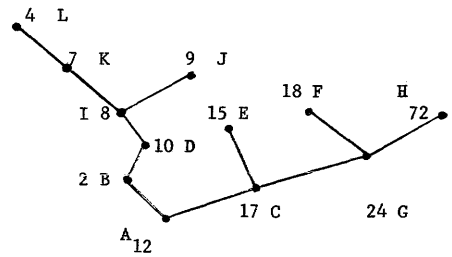
Replacement selecting may be linear, quadratic, cubic, or n degree. Each new record is read into an input area, replacing the low record after each selecting pass, thus permitting a newly read record to be eligible for selection. This technique requires more compares over non-replacement selecting, for each record to be placed in storage must satisfy two conditions: 1) itis lower than the control field in the associated linear storage area, and 2) it is equal or higher than the control field of the previous high control field of the record last transmitted to the output area.

Tree and Forest Sorting

Tree sorting can be more readily understood by examining the following example

Sequence of Numbers
12
17
15
2
24
10
18
8
7
72
4
9

Resulting Tree



The first control number of the input string is termed the root. There is one and only one root for a tree, so that for any node (number) n, there exists only one path which begins at the root and ends at n.

For every sequencing, the high number is placed to the right and the lower number to the left of an existing node. In order to make a meaningful sequence of nodes, a left address, right address, and back address is assigned to each item.

Location	Item	Left Address	Right Address	Back Address
A	12	B	C	-
B	2	-	D	A
C	17	E	G	A
D	10	I	-	B
E	15	-	-	C
F	18	-	-	G
G	24	F	H	C
H	72	-	-	G
I	8	K	J	D
J	9	-	-	I
K	7	L	-	I
L	4	-	-	K

By computer examination of the example string it is found that location B contains the lowest item since it does not have a left address. D is then examined, found to have a left address I, which in turn has a left address of K, which has a left address of L. L does not have a left address, therefore, it is the next higher record, building the sequenced string 2, 4.

L has no address other than the back address, which is added to the sequence 2, 4, 7. In like manner 8, 9, 10 are picked in sequence from locations I, J, D. Since D has already been picked, this signals the end of sequencing of the left branch of the tree. The right branch is then sequenced by the same method.

The address of the root point is the only parameter with which the system grows and picks. Therefore, it is possible to have two trees growing without interfering with each other. A multi-tree system is termed a forest. Items with the same control word are accumulated as they are placed on a tree. At no time is it necessary, then, to store the same key twice.

As a system, tree or forest sorting has a speed comparable to a merge. In addition, insertions are made very easily.

Conclusion

This paper has concerned itself with a brief survey of the major sorting techniques for computers.

In answer to the question "Which method is superior?", there is no single answer. Each method must be considered and evaluated in terms of the specific machine configuration, the speed of input and output devices, the record length and size of the control word, and the state of organization of the input data.

Every Systems Engineer must be aware of the multiple internal sorting techniques in order to more effectively guide customers in making modifications to existing sorts, and in designing special purpose programs.

BIBLIOGRAPHY

Bell, D. A.: The Principles of Sorting, Computer Journal, Vol. 1, pp. 71-77, July, 1958.

Bose, R. C. and Nelson, R. J.: A Sorting Problem, Journal of ACM, Vol. 9, pp. 283-296, April, 1962.

Douglas, A. S.: Techniques for the Recording of, and Reference to Data in a Computer, Computer Journal, Vol. 2, pp. 1-9, April, 1959.

Flores, I.: Analysis of Internal Computer Sorting, Journal of ACM, Vol. 8, pp. 41-80, January, 1961.

Flores, I.: Computer Time for Address Calculation Sorting, Journal of ACM, Vol. 7, pp. 389-409, October, 1960.

Friend, E. H.: Sorting on Electronic Computers, Journal of ACM, Vol. 3, pp. 134-168, July, 1956.

Gotlieb, C. C.: Sorting on Computers, Communications of ACM, Vol. 6, pp. 194-201, May, 1963.

Hall, M. H.: A Method of Comparing the Time Requirements of Sorting Methods, Communications of ACM, Vol. 6, pp. 259-263, May, 1963.

Hibbard, T. N.: Some Combinatorial Properties of Certain Trees with Applications to Searching and Sorting, Journal of ACM, Vol. 9, pp. 13-28, January, 1962.

Hidebrandt, P. and Isbitz, H.: Radix Exchange - An Internal Sorting Method for Digital Computers, Journal of ACM, Vol. 6, pp. 156-163, April, 1963.

Isaac, E. J. and Singleton, R. C.: Sorting by Address Calculation, Journal of ACM, Vol. 3, pp. 169-174, July, 1956.

McCracken, D. and Weiss, H. and Lee, T.: Programming Business Computers, John Wiley & Sons, Inc., New York, 1959.

Nagler, H.: Amphisbaenic Sorting, Journal of ACM, Vol. 6, pp. 459-468, October, 1959.

Windley, P. F.: Trees, Forests, and Rearranging, Computer Journal, Vol. 3, pp. 84-88, July, 1960.

..... Sorting Methods for IBM Data Processing Systems, IBM Corporation, Form Number F 28-8001