COLLATION METHODS FOR THE UNIVAC SYSTEM

VOLUME I

V + 169 pp

ACKNOWLEDGMENTS

This report on the adaptation of collation methods to the UNIVAC system is the work of the Computational Analysis Laboratory of the Eckert-Mauchly Computer Corporation. The basic methods were conceived and developed by Frances E. Snyder and Jean J. Bartik. The preparation of the entire report was under the immediate supervision of Miss Snyder.

In addition to Miss Snyder and Mrs. Bartik, the following individuals prepared the programs in Volume II: Margery K. League, Gaetana Melilli, Hildegard Nidecker, and Albert B. Tonik. Arthur A. Katz and William F. Schmitt assisted with the checking.

Volume I was written by Mrs. League, Miss Nidecker and Mr. Tonik. The flow charts for Volume II were drawn by Helen M. Diehl and Anthony P. Haimbach. The typing of both volumes was done by Mildred Beibel and Louise LaMas, who were aided in proofreading by Mildred Koss.

Herbert F. Mitchell, Jr.
Computational Analysis
Laboratory

TABLE OF CONTENTS

CHAPTER 6

EXTERNAL COLLATION

CHAPTER 7

MERGING

CHAPTER 8

TIMING OF COLLATION ROUTINES

CHAPTER 9

APPENDIX

CHAPTER 1

INTRODUCTION

Any mechanized system for processing large quantities of non-mathematical data must provide an efficient method for sorting that data according to specified criteria. It is the purpose of this report to explain the collation method of sorting as it has been adapted to the UNIVAC system.

In the discussions which follow, no restrictions are placed on the input data except that the items be uniform, and that they be arranged in random order. The location of key information, the size of the item, and the number of items to be collated are completely flexible considerations. The body of data may consist of any number of complete or partial tapes; the key digits may be numeric, alphabetic, or alpha-numeric in character.

The collation process is divided into three major sections. Two of these, Internal and External Collation, are employed together to sequence the items on a single tape. At the conclusion of Internal Collation, the items within each block on a tape are in monotonic sequence, but every block is random with respect to every other one. At the conclusion of External Collation, all the items on a single tape are in the desired order.

If there is only one data tape to be sequenced, collation is then complete. If there are additional tapes, each one must be sequenced in the same manner. At conclusion, the items on

each individual tape are in the desired order, but every tape is random with respect to every other tape.

The third division, Merging, sequences the items on the several collated tapes. At the completion of this last process, all of the items on all of the tapes are in sequence with each other.

The remainder of this present volume is devoted to a detailed explanation of the techniques employed in programming the collation process just described. Chapter 2 contains a glossary of computer terms which may be used for reference throughout the entire report. The basic binary method of collation, as applied to a small quantity of data, is described and illustrated in Chapter 3. Chapter 4 defines the manner in which the basic collation method is adapted to the input-output equipment of UNIVAC. Chapters 5, 6, and 7 are concerned with Internal Collation, External Collation, and Merging respectively. Each of these chapters reviews the techniques associated with one of the three divisions of the complete process, and explains, in detail, the logical development of at least one of the illustrative programs.

Chapter 8 contains a complete explanation of the methods used in timing collation programs. From the charts and tables included in these sections, collation times for additional item sizes, and varying quantities of data can be estimated. Finally, in Chapter 9, the present flow chart conventions are summarized, and a glossary of the collation flow chart symbols is listed.

Volume II of this report contains all the illustrative routines and tables which have been designed to accompany the discussion in Volume I. Although a few programs are presented by flow chart only, most include both flow chart and coding. In all cases where the flow chart alone is given, the major portion of the program is similar to a previously coded routine, or is related to another program.

Reference between Volume I and Volume II should not be difficult. The first digit of each page number in the second volume indicates the chapter in Volume I to which it is related. In addition, the pages of the two volumes can be distinguished by the second digit of the page number. In the first volume this digit is zero, and in the second volume, it is one. Thus, in Volume I, the discussion of Internal Collation begins on page 5.0.1.1, and in Volume II, the associated tables and programs begin on page 5.1.1.1. It should be convenient to refer to discussion and program simultaneously.

The programs in Volume II are designed to indicate the manner of processing several different key digit locations in items of many different sizes. No group of programs, therefore, can be fitted together. However, a single complete collation program can be produced with relatively simple adjustments in the existing routines.

For example, the Internal Collation routines illustrate the method of processing three specific item sizes. Each of the pro-

grams is developed according to the techniques recommended for the size of item concerned. To produce a program for an alternate item size, it is necessary to (1) decide upon the most satisfactory location of the key digits within the item, (2) select the modification of the basic method which is applicable to the item size chosen, and (3) consult the table, beginning on page 5.1.1.1, for the recommended composition of strings and cycles. The changes necessary can then be incorporated in a new program modeled after the appropriate sample routine.

For reasons discussed in Volume I, the techniques used in External Collation and Merging are independent of the size of the item collated. Any of these programs can be revised to process other key digit locations, or other item sizes, with no change in the logical development of the problem as shown on the flow chart. Only those routines concerned with the comparison of items and their transfer within the memory are affected. The corresponding adjustments in coding are uncomplicated.

Suggestions for other adjustments in programming are contained within the chapters of this report. Most are desirable for reasons of economy in computing time. If applicable, these alterations should be incorporated whenever other programming changes are being made. Additional changes may suggest themselves in time, and these, too, should be incorporated if efficiency or rerun considerations warrant them.

CHAPTER 2

DEFINITIONS OF COMPUTER TERMS

Block	The smallest input-output unit of the UNIVAC. A group of sixty words.
Block, Auxiliary	The memory locations allocated to sixty words of input data in addition to those already stored in the input block; also, the contents of those locations.
Block, Input	The memory locations allocated to sixty words of data from an input tape; also, the contents of those locations.
Block, Output	The sixty memory locations allocated to the output of computer processing operations; also, the contents of those locations. When sixty locations have been filled, the block is written on the output tape.
Block, Partial Last	A final data block which is not completely filled with true items. The words which are not data or sentinel are of no importance.
Block, Sentinel	A block which contains one or two words of sentinel in certain specified locations. Two or more sentinel blocks follow the last full block of data on every UNIVAC tape.
Collation	The method of sorting whereby increasingly larger strings of items are sequenced according to the magnitude of the entire group of key digits in each item.
Collation, External	The UNIVAC process which operates upon an internally collated tape, sequencing the items in every block with respect to those in every other block.

Collation, Four-way	The collation method by which the components of four input groups are compared and sequenced.
Collation, Internal	The UNIVAC process which rearranges all the items within each block on a data tape in the desired sequence, but leaves every block in random order with respect to every other one.
Collation, Two-way	The collation method by which the components of two input groups are compared and sequenced.
Collation, Three-way	The collation method by which the components of three input groups are compared and sequenced.
Cycle	A period of computing time during which the number of units in sequence within a string is increased, while the total number of strings is reduced.
Digit	A single decimal numeral, alphabetic character, or other symbol used in UNIVAC operations. The unit component of a word.
Digits, Key	Those digits within an item which have been designated as the basis for sorting operations.
Digits, Satellite	All the digits of an item, other than the key digits.
Group	In two-way collation, approximately half the total number of input strings to be collated. In three-way collation, approximately one third of the total number of input strings to be collated.

Identification, Tape	A system for internally labelling each of a series of related output tapes with sufficient information to uniquely identify that tape with regard to all other tapes in the same series, and in every other series, and to make possible the rerun of the tape, if necessary.
Interlock, Tape	The interruption of all computational operations for the period needed to complete the tape operation currently in progress, and to initiate the one required by the current instruction. Whenever two successive read or write instructions, or a read and write instruction involving the same tape occur too closely together, the computer is interlocked at the moment of carrying out the second instruction, until the first operation is complete.
Item	Any number of digits or words which are linked together by a common relationship or characteristic to form a single unit of data.
Label, Tape	A combination of two alphabetic characters and two numeric digits which is assigned to a data tape in order to distinguish it from all other such tapes.
Merging	The UNIVAC process which operates upon the several individually collated tapes, placing all the items on a single string of tapes which are then in sequence with each other.
Rerun	The reprocessing of a data tape which has been incompletely or unsatisfactorily processed before.
Sentinel	A complete word of some non-data character with extremely high pulse combination which is used to indicate the end of the data stored on a tape.

Sequence, Ascending	A sequence of items in which the item containing the key digits of least magnitude is found at the beginning, and the item containing the key digits of greatest magnitude is found at the end.
Sequence, Descending	A sequence of items in which the item containing the key digits of greatest magnitude is found at the beginning, and the item containing the key digits of least magnitude is found at the end.
Sequence, Monotonic	A series of units which are in order according to a single group of specified key digits.
Sorting	A term which is applied to any method of rearranging a group of random items in a desired sequence.
Sorting, Digital	The method of sorting whereby all items are sequenced according to the magnitude of one key digit at a time.
String	A sequence of one or more units of data.
String, Incomplete	A string which contains fewer than the number of units of data specified for a string in the current cycle.
String, Input	A sequence of one or more units of data which is to be processed during the current cycle.
String, Output	A sequence of two or more units of data which has resulted from the collation of a string from each of the input groups.
Word	A sequence of twelve digits; the unit of all computer operations except those which involve multiple word transfers; the contents of one memory location.

CHAPTER 3

THE BINARY METHOD OF COLLATION

3.0.1. The Basic Collation Method.

In collation, the foremost problem is one of rearranging a series of random items into a monotonic sequence, according to one or more key digits. This may be done by examining the first two items, placing them in proper order to form a string of two in sequence, and then successively examining every other item to determine where it should be placed in that string. This procedure, however, requires many comparisons, and, if the quantity of data to be processed is large, it is not an efficient plan.

The more general method of collation is a binary one. If 2^W items are to be sequenced, the first step is to form 2^{W-1} strings of two items each. The second step is to collate these strings of two into strings of four. In each succeeding step the strings which resulted from the last collation are collated by pairs to produce strings of twice the previous length. The final arrangement into one string is achieved by W such collations.

The simple example below indicates how a group of items may be processed to form a descending monotonic sequence. Each item is represented by two digits. The first (the key) is a number. The second (the satellite information) is an alphabetic character. The arrangement of data before the beginning of the first cycle of operation is shown in the left-hand column. Rows of dots mark off the groups of items which are to be formed into a string during any cycle.

Original Data	Output Cycle 1	Output Cycle 2	Output Cycle 3
8A	8A	8A	8A
3B	3B	5D	7E
2C	5D	3B	6H
5D	2C	2C	5D
7E	7E	7E	4G
1F	1F	6H	3B
4G	6H	4G	2C
6H	4G	1F	1F

At the beginning of the first cycle, 8 is compared with 3. No rearrangement of the items is necessary, as 8 is larger and already appears first. Then 2 is compared with 5. These items must be rearranged to form a string in descending order. In doing so, D is carried along with five, and C is transferred with 2. To collate the next pair, no inversion is needed. In the last pair, 6H and 4G are interchanged. At the end of the cycle, there are four strings of two items each.

During the second cycle the first item of the first string is compared with the first item of the second string. As 8 is larger than 5, 8A becomes the first entry in the new string being formed. The next comparison is between 5 and 3. This time, 5 is the digit of greater magnitude and, therefore, 5D is transferred. The next item in the second two-item string is 2C. When 2 is compared with 3, it is found that 3B should be transferred first, followed by 2C. This completes the first string of four items. The second string of four is arranged by similar techniques. The items are compared in the order (7,6), (6,1), and (4,1).

In the third cycle, the two strings formed during the second cycle are collated into a single string of eight items. The order

in which the pairs of key digits are drawn and compared is: (8,7), (7,5), (5,6), (5,4), (4,3), (3,1) and (2,1). Collation is then complete.

A subroutine, showing the collation of 64 two-word items is included in section 3.1.1 of this chapter. In all essential operations, it follows the pattern of the method just described. The only significant change in technique concerns the manner of selecting the items to be compared during all cycles prior to the last. For purposes of programming, it is more convenient to divide the total quantity of data into two equal groups at the outset, than to delay until the final round of comparisons. This affects the contents of the various intermediate strings, but the final result is the same.

To illustrate, the previous example is recollated below. As before, the dotted lines indicate the manner in which the data is grouped. The items included in each string are bracketed together.

Original Data	Output Cycle 1	Output Cycle 2	Output Cycle 3
8A	8A	8A	8A
3B	7E	7E	7E
2C	3B	4G	6H
5D	1F	2C	5D
7E	4G	6H	4G
1F	2C	5D	3B
4G	6H	3B	2C
6H	5D	1F	1F

Regardless of the number of strings to be collated in any cycle, the initial comparisons are between the items of the first string in group one and those in the first string in group two. The remaining strings in the two groups are paired similarly.

In so far as possible, the collation process has been programmed to save both memory space and computing time. To conserve memory space, the comparison and data transfer routines are coded only once. To save time, each routine is coded as efficiently as possible.

During each cycle, the comparison routine is iterated whenever there are two items to compare. Depending upon the nature of the data, one of the two transfer routines is iterated for every item in the two input groups. The only variation from one cycle to the next, is in the number of items which are to be collated together.

To make these iterations possible, several counters have been introduced for computer control. Each counter is increased and compared with its limit at some time during the routine with which it is associated. As long as counter and limit are unequal, additional iterations of the routine are required. When counter and limit are equal, the program is adjusted to initiate a different series of computer operations.

Two paths of operation must be provided for every test. When the events which precede the test require changes in the sequence of events which normally follow the test, all alternate routines must be provided as well. Only those routines which apply to the current flow of the collation process are available or "set" at the moment when the test is made.

At the start of the subroutine in section 3.1.1, the input data $A_0 \ldots A_{127}$ is divided into two groups of 32 items each;

$B_0 \ldots B_{63}$ and $C_0 \ldots C_{63}$. The first word of item 1, group B, is compared with the first word of item 1, group C. The keyword of grea er magnitude and its satellite information (word two) are transferred to the first two output locations. The keyword of smaller magnitude and its satellite are transferred to the second two output locations.

Continuing in the same manner, successive items in group B are compared with those in group C, always transferring the item of greater magnitude first. When the last item in each group has been transferred, the output is a series of two-item strings, each properly collated. Additional iterations yield collated strings of 4, 8, 16, 32, and 64 items. Thus, at the completion of the sixth iteration or cycle, the data has been transferred to the output position in a single series of 64 two-word items arranged in descending order with respect to the first word of each item.

The number of cycles and the length of the string in each cycle are controlled by the quantities n, n_1, and n_2. At the start n, the number of items to be compared for a single string, is equa to one. When n_1 and n_2, the number of items transferred from B and C respectively, are each equal to n, a string in the cycle is complete. The next comparison begins a new string. When all 64 items have been transferred, the cycle is complete. The limit, n, is advanced to 2n, and the data is transferred back to the input locations for the next iteration. In the second cycle, n equals 2, and therefore when n_1 and n_2 both equal 2, every string in the output

location contains four collated items. The number of items in a single string is doubled for each successive cycle. When 2n equals 64, the sixth cycle is complete and the data is in the desired order.

The constants for this subroutine are stored in memory locations 397 through 419. The input data is in locations 420 through 547. The output data is in locations 600 through 727. The routine is entered at 300 from the main routine and reenters the main routine at a point inserted in instruction 346. Provisions are made to change the instructions as necessary, in order to complete the sixth cycle before returning to the main routine.

CHAPTER 4

THE COLLATION METHOD APPLIED TO LARGE AMOUNTS OF DATA

4.0.1. Adaptation of the Collation Method to UNIVAC.

The subroutine which accompanies Chapter 3 is of use when collation is a small part of a more extensive program, and the number of items to be processed is not great. With minor adjustments in coding, the number of words per item, or the number of items to be collated, may be somewhat expanded. This is a limited improvement at best, because the capacity of the internal memory is soon exhausted. To obtain complete flexibility in both size of unit and quantity to be processed, the external memory of the computer must be utilized. This, in turn, necessitates the adjustment of the basic collation method to the input-output equipment of UNIVAC. It is the purpose of Chapter 4 to indicate how this may be done.

UNIVAC input units may be considered to be of three orders of magnitude: the block, the tape, and the group of related tapes. The smallest of these is the 60-word block. Every data tape must contain at least one such block, and may contain the maximum number if complete, or any smaller quantity, if not complete. The group of tapes which comprise the entire body of data to be processed may consist of any number of partial or complete tapes.

Similarly, there are three major divisions of the complete collation process. The first, Internal Collation, sequences the items within each block but leaves every block on the tape random with respect to every other one. External Collation, which follows, sequences each block with respect to every other block, but leaves

the tape random with respect to every other tape. Finally, the Merging programs process the several individually collated tapes to form a series of tapes which are in sequence with each other.

The Internal Collation program is the only one which closely resembles the subroutine of Chapter 3. No important change in basic operations has been introduced, although computer instructions are reduced in order to process only those items included within a single block, and augmented in order to process successive blocks on a complete tape. However, this general method is not always economical. When it is wasteful for a particular size of item, variations are introduced to increase computer efficiency. At first glance, the modified programs may not appear comparable, but the logical development is actually the same.

The External Collation programs operate upon the output of Internal Collation. Item by item, the contents of the collated blocks are sequenced into strings of two, four, eight, etc. blocks. At the end of the eleventh External cycle, an entire tape containing the maximum number of blocks is in order. A tape of fewer blocks is completely collated in a smaller number of cycles.

The item size does not affect the operation of External Collation in the same manner as it does Internal Collation, and therefore, major changes in External Collation programming do not arise from this source. Efficiency considerations make it desirable to reduce the number of cycles, however, and various modifications which accomplish this have been developed.

Both Internal and External Collation programs are necessary in order to completely sequence the items on a single tape. If

there is more than one data tape to be processed, collation is not complete until all tapes have been merged together. Item by item, the contents of the collated tapes are sequenced in strings of two, four, eight, etc. tapes. The operator determines the number of cycles required to complete merging from the total number of tapes to be processed. At the conclusion of the final cycle, all items of data are in the desired sequence.

Methods developed in order to reduce the number of External cycles required are generally applicable to Merging as well. All are discussed at greater length in section 4.0.3 of this chapter.

4.0.2. Collation Procedure.

The collation method employed in the programs of this report has been described in broad general outline, without mentioning the techniques used to implement it. At least a few of these require explanation, and are treated briefly below. Those which are not unique to the collation process will not be discussed further, but additional explanation of the others may be found as necessary in Chapters 5, 6, and 7.

Alternation of Tapes: The input source for Internal Collation is a single data tape. The final output consists of two tapes, each containing approximately half of the total number of collated blocks. By alternating the designated output servo after the collation of each block, the efficient division of the data into two approximately equal groups is assured. If the two tapes do not contain exactly the same amount of data, the first is the longer, and contains only one additional block.

The output of Internal Collation is the input for the first cycle of External Collation. As the block is the unit of subdivision for External Collation, the input data may be considered to consist of two nearly equal groups of one-block strings. Two output tapes are again used, but the alternation takes place following the writing of each string of two collated blocks. This again insures the division of the data into two groups of approximately the same number of strings for the next collation cycle. If one group is larger than the other, it must again be the first, and it cannot contain more than two additional blocks (one string) of data. These tapes then become the input tapes for the second External cycle.

Each External cycle which follows maintains the same plan, alternating output tapes after the collation of each string. There is only one final output tape as the output of the last cycle is a single string containing all the items of data in sequence.

The Merging programs employ multiple input and output servos also. The input for each run of the first Merging cycle is two completely collated data tapes. When the items from these tapes have been merged, the output string consists of two tapes in sequence. In order to conserve computing time, it is convenient to assign two UNISERVOs for these tapes. When the first tape contains the maximum number of blocks, the second is used.

In the later Merging cycles, when each of the input strings consists of two, four, eight or etc. tapes in sequence, similar economy considerations make it convenient to assign two servos for each input string. The first tape in the string is read from one servo, the second from another and the third from the first servo

again. This type of alternation does not affect the collation method

Strings: The interpretation of the term string in the Internal Collation programs differs from that in the External Collation programs, although the comparison of one item with another is basic to both. The unit component of a string in Internal Collation is an item. The string is formed by comparing R items from one group with an equal number from the second group, and arranging them in a monotonic sequence of 2R items. The unit component of a string in External Collation is a block. The string is formed by comparing the items in R blocks from one input tape with those in R blocks from the second input tape, forming a string of 2R blocks in the desired sequence. Tests must be made to determine when the end of the string and the end of the group have been reached. Some flexibility in the order in which these tests take place is possible. The order employed in the accompanying programs is the one which requires the fewest tests, and thereby, permits the greatest economy in computing time.

In the Internal Collation programs, a test for the end of the group concerned follows the transfer of an input item to the output location. If the end of the group has been reached, the end of the current input string in that group must have been reached also, and the output string is completed with the items remaining in the other group. If the end of the group has not been reached, then a test for the end of the current string in that group follows. If the end of the input string has been reached, the current output string is completed with the items remaining in the current string in the other group. If the end of the string has not been reached, the

next item is processed. No test for the end of a block is made during Internal Collation, as only one block of data is processed at a time, and all items in that block have been processed when both groups have been collated.

In External Collation, more than one block of data is included in an input string from each group during every cycle except the first. For this reason, a test for the end of the block concerned must be made after the transfer of each input item to the output location. If the current block in the group has not been exhausted, collation continues with the processing of the next item in the block.

If all the items in the current block have been processed, the next block in sequence is transferred to the working location of the group. A test is made to determine whether it contains data or an end-of-tape indicator. If no data is present, the group has been exhausted, and the current output string is completed with items remaining in the other group. If the group has not been exhausted, a test is made to determine whether the new block is part of the current string, or the first of a new string. If the current string has been exhausted, adjustments are made to complete the output string with the remaining data from the other input string before processing the new block. If the current input string has not been exhausted, collation continues with the processing of the first item in the new block.

In Merging, the end of a group is an external matter determined by the operator in charge of the program, who must know the

total number of tapes to be merged during the current cycle. The program tests for the end-of-block and end-of-tape in the normal manner. However, as there are several input tapes in each group during every cycle after the first, the end-of-tape test is followed by a test to determine whether the last tape in the string concerned has been processed. If the input string has been exhausted, the output string is completed with the items remaining on the tape(s) of the other string. If the current string is not exhausted, collation continues with the processing of the first item on the new tape.

Cycles: In Chapter 3, the term cycle was interpreted as a period of computing time, during which the number of items in sequence in each string is doubled, and the total number of strings is therefore halved. If W represents the number of cycles required to collate N items in a single string, then $2^W = N$. This same interpretation applies equally well to the Internal and External Collation programs, if unit is substituted for item, and if N is temporarily assumed to equal an integral power of two at all times. The number of units (N items) in a block determines the number of Internal cycles required, and the number of units (N blocks) on the original data tape determines the number of External cycles required.

The number of cycles is an integral part of the Internal Collation programs. Computer controls are established for each cycle to insure the iteration of comparison and transfer routines as frequently as necessary. When the items within one block have been completely collated, the resulting sequence is written on an output tape. The next input block is then processed, beginning with

the first cycle.

If the data tape contains the maximum number of blocks, eleven External Collation cycles are required to complete collation. Because it is expected that some tapes may not contain the maximum number, the External Collation programs are designed to process a tape of any shorter length, with a corresponding decrease in the number of External cycles required. It is assumed that information regarding the number of blocks on each tape will not be available to the operator. A count is made within the computer during Internal Collation and a routine at the end of the Internal program computes the number of External cycles required from this quantity. A routine in each External cycle tallies the cycle about to be started. When tests indicate that the current cycle is the last, controls are established which will conclude the program after all the data has been placed in an ascending monotonic sequence.

This methods of establishing computer controls internally is of advantage when a large number of tapes of varying lengths are to be collated, and the length of each is not conveniently available. If the tape lengths are already known , the collation program might be altered to allow this previous count to be used in establishing the necessary controls. If the Supervisory Control must be employed for this purpose, it is doubtful whether any economy in computing time will result.

In Merging, the number of cycles is controlled by the operator in charge who maintains a schedule for the complete collation of the total quantity of data. The Merging programs, as they now

exist, are independent of the total number of unit tapes to be processed.

Ascending and Descending Order: Throughout the collation and merging programs it is assumed that the desired final order of the data is an ascending monotonic sequence. Internal Collation, External Collation, and Merging might all be programmed to collate the data in this fashion. Intermediate output tapes would then require rewinding before use as the input tapes for the next cycle. As the ensuing delay would be wasteful of computer time, an alternate technique has been developed which eliminates the necessity of rewinding during External Collation. The nature of the Merging program, where an input string may consist of more than one tape, makes an attempt at this type of economy impossible. Therefore, the data is always collated in ascending order during the final External cycle and all Merging cycles.

When rewinding is eliminated, the input tapes are read into the memory with the tapes moving in a backward direction. The first digit to appear at the reading head is the least significant digit of the last word in the last block written on the tape. The input mechanism automatically assembles the digits in each word and the words within the block, in reverse order. Thus, the original sequence of the data within the block is not disturbed, regardless of the direction in which the tape is moving at the time of reading.

When there is only one block in a string, as at the beginning of the first External cycle, collation may be carried out either in the same monotonic sequence, as previously, or in re-

verse order. However, during those cycles when the input strings each contain a larger number of blocks, the first block available is the last in the string. There is not sufficient memory space available to store all of the blocks in each string for sequential processing, beginning with the first.

This complication is eliminated by alternating ascending and descending sequences in successive cycles. If the output of a given cycle is in ascending order, then the first block of each input string to be processed during the next cycle includes the items of largest magnitude. The last item in the current string from group one is compared with the last in the current string from group two. The item of largest magnitude is transferred to the first output location. Moving upward through the items in the successive blocks of the two strings, additional iterations of the comparison and transfer routines yield an output string collated in descending sequence. The process continues until all the data has been treated in this manner. During the next cycle, the order is reversed again, to obtain strings which are collated in ascending series. At the conclusion of the last cycle, all items are included in a single string in the desired sequence.

A full data tape requiring eleven External cycles for complete collation, must be internally collated in descending sequence if the final output is to be in ascending sequence. The first External cycle and therefore the last, is completed in ascending sequence. Tapes of intermediate lengths requiring an odd number of cycles to complete collation, are sequenced in the same manner, although the processing is shorter. Tapes requiring an even number

of External cycles are not collated in the desired final sequence by this method. To avoid reprocessing the original data, a special descending program is inserted immediately following Internal Collation. In this cycle the data is again collated in descending order. The second, and all even-numbered cycles are in ascending sequence , and the final result is then comparable. If the desired final sequence is the reverse order, the programs must be adjusted accordingly to yield a final output tape in descending monotonic sequence.

Figure 1 below briefly illustrates the technique of collating by alternating ascending and descending cycles. A twenty-word item is assumed, and is identified by its key digits only. (Satellite digits may be considered to be processed in the manner indicated in Chapter 3). The number appearing at the left of each block of three items indicates the relative order in which it would be read in or written out. Separating lines appear between input blocks; separating lines appear between output strings.

Figure 1. EXTERNAL COLLATION OF EIGHT BLOCKS OF THREE ITEMS EACH.

Internal Collation

	Output
Tape 1	8
#7	4
	3
	18
#5	17
	13
	17
#3	15
	4
	30
#1	19
	16
Tape 2	20
#8	7
	2
	12
#6	9
	5
	2
#4	1
	0
	10
#2	6
	0

First Cycle - External Collation

Input		Output	
Tape 1	8	Tape 1	2
#1	4		3
	3	#1, #2	4
	18		7
#3	17		8
	13		20
	17		0
#5	15		1
	4		2
	30		4
#7	19	#5, #6	15
	16		17
Tape 2	20	Tape 2	5
#2	7		9
	2		12
	12	#3, #4	13
#4	9		17
	5		18
	2		0
#6	1		6
	0		10
	10	#7, #8	16
#8	6		19
	0		30

Second Cycle-External Collation

Input		Output	
Tape 1	4	Tape 1	30
#1	15		19
	17		17
	0		16
#3	1		15
	2		10
	7	#1,#2,	6
#5	8	#3,#4	4
	20		2
	2		1
#8	3		0
	4		0
Tape 2	16	Tape 2	20
#2	19		18
	30		17
	0		13
#4	6	#5,#6	12
	10	#7,#8	9
	13		8
#6	17		7
	18		5
	5		4
#7	9		3
	12		2

Third Cycle-External Collation

Input		Output	
Tape 1	1	Tape 1	0
#1	0		0
	0		1
	6		2
#3	4	#1	2
	2		3
	16	thru	4
#5	15		4
	10	#8	5
	30		6
#8	19		7
	17		8
Tape 2	4		9
#2	3		10
	2		12
	8		13
#4	7		15
	5		16
	13		17
#6	12		17
	9		18
	20		19
#7	18		20
	17		30

End of Tape: Throughout all discussions concerning UNIVAC, the phrase "end of tape" is used to mean the end of the data which has been stored thereon, rather than the physical end of the tape itself. Although the two limits may frequently coincide, there will be many occasions when a partial tape is to be used. It is therefore necessary to standardize a method for indicating the end of the data to the computer. This is accomplished by the insertion of an end-of-tape sentinel in one or more specified locations in the block which follows the last full block of data.

The end-of-tape sentinels have not been specified in the programs which accompany this report. A word of any character which has an extremely high pulse combination may be used, as long as it easily distinguished from a word of true data. All initial input and final output tapes containing only full blocks of data carry this sentinel in the first and last position of the block following the last data block. Similar tapes containing a partial last block of data carry the sentinel in the first non-data and the final location in that block. The Internal Collation programs which use these tapes test each data block for the presence of sentinel in the last position. If sentinel is not present, the current block is a complete block of data. When sentinel is present in the last position, the first location in the block is tested. If sentinel is present in both positions, all the data has been processed. If sentinel is present in the last position and not in the first, the current block is a partial block, and successive items are tested to determine the limit of the data.

A group of related programs may utilize many intermediate output tapes before the final one. If the data can be arranged in integral block units, a less extensive system for end-of-tape indication can be used. A simplified procedure is used in the collation programs, which cannot process incomplete blocks of data. The sentinel blocks placed on the Internal Collation output tapes contain a sentinel indication in only one location in the block. That location is the only one tested in the External Collation program which uses these tapes for its input source. The technique is continued throughout External Collation and Merging until the final Merging cycle is reached. At this time, the normal method of sentinel placement is resumed on the final output tape.

When output tapes are to be rewound before re-use as input tapes, the end-of-tape sentinels are written after the last block of data. If intermediate output tapes are to be used without rewinding, as they are in External Collation, the sentinels must be placed on the output tape before the first block of data is written. Subsequent routines will read the tapes as they move in a backward direction. The number of sentinel blocks to be written on a tape varies with the control problem introduced by the next program in the processing. In general, two are required for use with all programs utilizing continuous read instructions, and one or two data storage blocks per tape. Data tapes to be used in programs employing additional auxiliary storage locations require three sentinel blocks.

Because the collation programs cannot process incomplete

blocks of data, the Internal Collation program includes a routine which converts a partial last block into a full block by inserting appropriate words for the missing items. If the final sequence of the data is to be in ascending order, the inserted items consist entirely of ignore symbols, which have the lowest pulse combination. When this block is collated, the key digits of the ignore items are of smaller magnitude than the key digits of all true items and are grouped at the end of the descending monotonic sequence. Throughout subsequent External Collation and Merging, the ignore items continue to be the items of the least magnitude. They appear at the end or beginning of a string, depending upon the sequence of the data which follows. During all Merging cycles prior to the last, a complete block of these items is discarded if it occurs in the collated sequence. Only those ignore items necessary to a complete first block in each string are retained. At the beginning of the last Merging cycle, all remaining ignore items appear before the true items in the collated sequence and are eliminated. The normal technique of placing sentinels in a partial last block of data may then be resumed.

When a data tape is to be finally collated in descending monotonic sequence, a partial block may be expanded by the insertion of words containing characters of the highest pulse combination. These words will be regarded throughout collation and merging as the items of greatest magnitude, and will finally appear at the beginning of the input strings in the Merging cycles. They may be eliminated during the final cycle in the same manner that ignore symbols are eliminated.

Incomplete Strings: Earlier in this chapter, the expression $2^W = N$ was given to demonstrate the relationship between N, the number of units to be collated and W, the number of cycles required to complete collation. In the discussion which followed, N was assumed to be a quantity such that W was always an integer, and the problem of intermediate values was not introduced. It is, however, undesirable to limit the flexibility of the collation program in this manner. On the other hand, if N is not a perfect power of two, the number of cycles to be programmed must be determined in another manner.

In the programs which accompany this report, the problem of fractional powers has been avoided by providing as many complete cycles as would be required if N were the next perfect power of two. In the same notation, $2^W \geq N > 2^{W-1}$. Compensation for the discrepancy in the number of units to be collated is made by allowing some strings to have less than the proper number in one or more cycles. These are incomplete or partial strings.

All collation programs establish a series of counters to aid in maintaining control of the number of units in a group, the number of units to be compared for a string in each cycle, and the total number of cycles required to complete collation. During the iteration of each associated subroutine, a specified counter is increased and is compared with its limit to determine whether the current iteration is the last. When incomplete or partial strings are to be processed, fewer iterations of certain subroutines are required, and the normal sequence of computer operations is altered

by changing the necessary counters. It is a matter of convenience whether the counter itself, or its limit is adjusted.

In Internal Collation, each individual program is designed to collate a specified number of units, and the programmer is able to pre-assign the location of a partial string within the data storage block. Depending upon the number of items to be processed, the incomplete string originates in the first, or in some later cycle, and may be disregarded during the operation of several cycles prior to the last. When a string is disregarded, the counters which control the number of units in each group must be adjusted, because the number of items under consideration, and hence, the number in each group, is reduced. Similarly, when the string is again included, the same counters must be restored to their initial values.

The normal method of doubling the number of items in an output string for successive cycles is maintained throughout the program, as only one string can vary from the normal length. If the practice of testing for end-of-string before testing for end-of-group were to be continued, under these circumstances, an incomplete string would never be discovered and errors would ensue. For this reason, the Internal, and all other collation programs, have been altered to test for end-of-group first. As the end of a group and the end of the last string in that group coincide when strings are complete, no error results from the change in programming, and a small economy in time is actually introduced.

When counters must be altered several times during the course of an Internal Collation program, the changes are made at the com-

pletion of one cycle in preparation for the next. The programmer who predetermines the existence of a partial string, also provides the necessary routines to accomplish this.

The External Collation programs are more flexible, and can be used to collate a complete tape or any smaller number of block units. It is not possible, therefore, to preassign the number of cycles required to collate a given data tape, or to provide the alternate routines necessary for the correct processing of a single specific partial string. Adjustments of this type must be accomplished automatically within a program designed to provide for all variations in the quantity of data to be operated upon.

If the number of units to be collated is odd, an incomplete output string appears during the first External cycle. If N is even, however, no partial string occurs until the cycle following the one in which an odd number of output strings first appear. In either case, deviation from the normal program is required by the occurrence of an extra string on the input tape which contains group one. When tests indicate that such a string exists, the routines which follow permit the extra data to be transferred to the current output tape. The counter which controls the number of blocks in the output string on the tape does not reach its maximum limit at this time. This fact is used to determine the adjustments in input and output block counters required for proper collation at the beginning of the next cycle. After an incomplete string has once appeared, additional ones will appear during every cycle thereafter, until collation is complete. All are processed in the same manner.

The four counters used to control the number of blocks in the External input and output strings are initially set to zero at the conclusion of the Internal Collation program. In the cycles which follow, u and v count blocks transferred from group one and group two input strings respectively, and s and t count blocks transferred to the current string on the first and second output tapes, respectively. When u and v both equal R, s equals 2R, and one output string is complete. In preparation for the next string, counters u, v, and s are reset to zero, and the alternate output tape is designated. When u and v again equal R, t equals 2R, and a second output string is complete. Again the counters are restored to zero, and the output tape is alternated. This process is continued until all the data has been collated. The final subroutine tests s and t to determine whether both have been reset. If so, every output string is complete, and no adjustment in any counter is required for the following cycle.

If either s or t has not been reset to zero, the final output string contains data from an extra string in input group one. Because the technique of reading input tapes backward is employed, the partial string is the first in its group to be processed in the following cycle. In order to continue collation correctly, the first output string in that cycle must contain the items from the incomplete input string and those from a complete string in the other input group. No permanent change in counters or counter limits can be used to accomplish this, as subsequent input and output strings prior to the last will be complete.

It is therefore convenient to preset the affected counters to compensate for the missing items. If 2R represents the number of blocks in a complete output string, and 2R-s is the number of blocks missing from the current incomplete string, then 2R-s is the value to which the first s and u in the next cycle are preset. During this next cycle, when the value of R has been doubled, u is increased after the processing of each block, until all the items in the incomplete string have been collated. Computer-wise, R blocks from this input string seem to have been processed, and the remaining units for the output string are obtained from the complete string in the other input group. Similarly, s appears to equal 2R when the incomplete string and complete string have been collated together. If 2R-t is the number of blocks missing from the current incomplete string in any cycle, v and s are preset with this value, and the procedure in the next cycle is comparable.

By adjusting counters in this way, the partial output string collated at the beginning of a cycle is made to appear complete. Subsequent output strings prior to the last must of necessity be complete. The final output string will also be complete if it includes items from a complete string in each input group. However, if there is an extra string or an incomplete string in an input group, the final output string will again be incomplete. The new partial output string is processed in the manner already described.

The partial output string collated at the beginning of a given cycle is the last input string in its data group in the next cycle, and creates another incomplete output string when it is processed. The components of this string will be the components

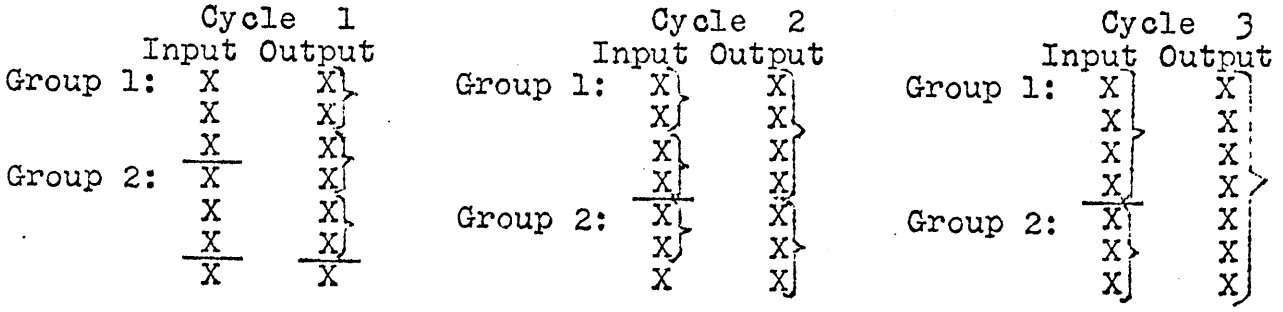
of another final partial string two cycles later. Additional incomplete strings will appear in every cycle thereafter.

The Merging program is designed to process only a single string from each input group at any one time and it is not affected by incomplete strings in the same manner. Counters established at the beginning of each "run" serve as computer controls which automatically terminate the program when all the unit tapes in the input strings have been processed. The number of output units is also adjusted automatically. Therefore, no special routine must be established for a string which is incomplete. Section 4.0.5. of this chapter, and Chapter 7 discuss this situation at greater length.

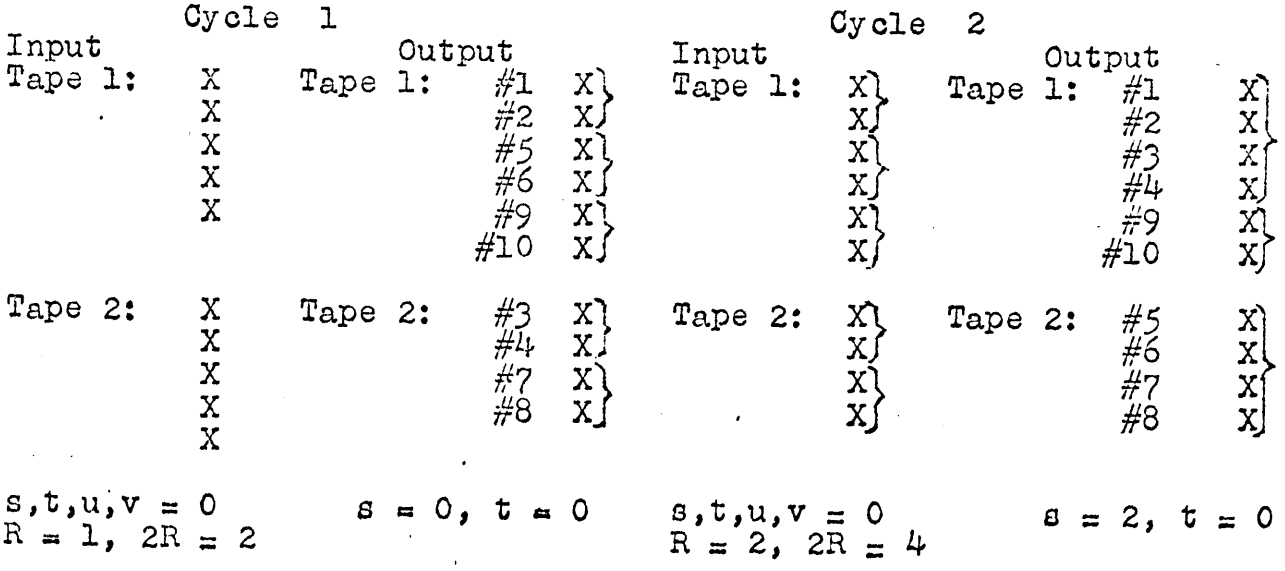
The diagramatic scheme in Figure 2 indicates the method of processing incomplete strings in two typical cases. Example I demonstrates the manner of collating one block of seven eight-word items (each item represented by an X). This is the method recommended by the table on page 5.1.1.3. Example II demonstrates the manner of Externally collating one tape containing a total of ten blocks of data. At the beginning, the blocks, each indicated by an X, are divided into two groups as they would be at the conclusion of Internal Collation. The status of the counters s, t, u, v, R and 2R is shown for the input of each cycle. The status of the counters s and t is shown for the output of every cycle. The numbers beside the output blocks indicate the order in which they were completed.

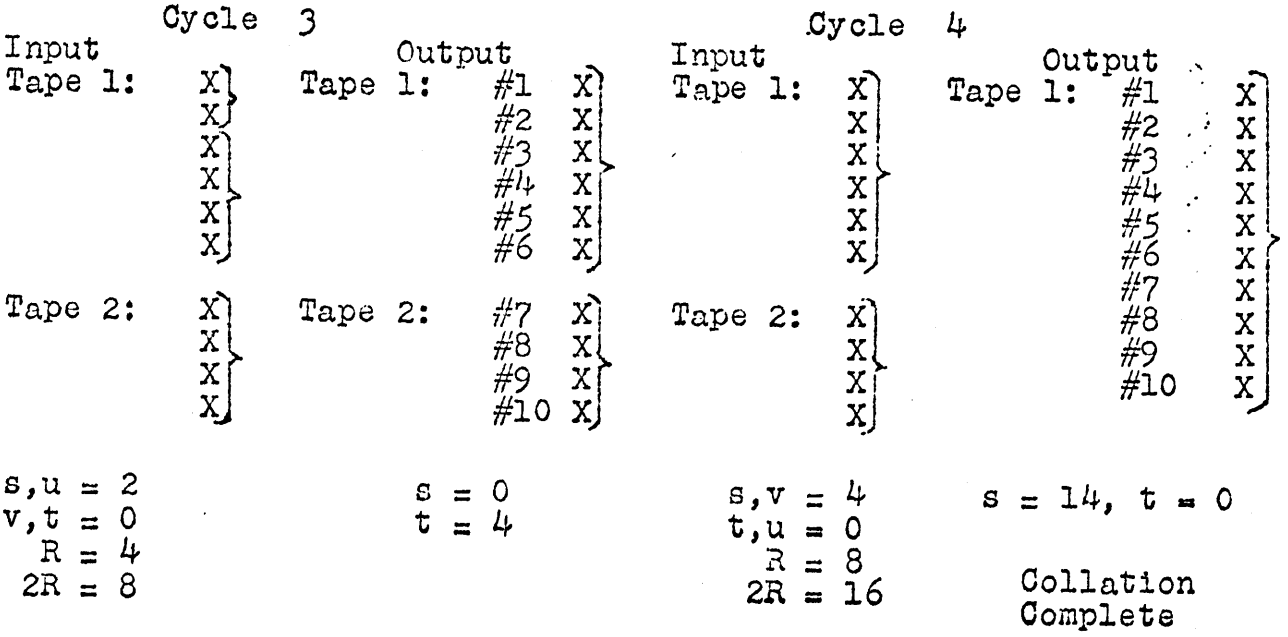
Figure 2. COLLATION OF INCOMPLETE STRINGS

I. Internal Collation: One block of seven eight-word items.



II. External Collation: Ten blocks of data.





In the final cycle of Example II both input strings are incomplete although only one partial string is anticipated. No counters have been adjusted for the partial string on tape one because it appeared complete in cycle three by virture of previous counter adjustments. The output string in the fourth cycle is therefore incomplete also. However, collation is concluded in the fourth cycle, and the final status of input and output block counters is immaterial.

Special External Collation Programming: When the total number of External cycles is even, a special descending program is inserted between Internal Collation and the first ascending External cycle. Following completion of the special cycle, ascending and descending series programs are alternated in the usual manner. The second External cycle, and therefore all even-numbered ones including the last, place the data in ascending sequence. In this manner, it is possible to maintain the same general plan, and yet avoid completing collation in descending sequence.

The special descending-following-descending program differs from the other External programs in that it does not reverse the sequence of the data. Input items are in descending order from Internal Collation. Comparisons are made between the items in a block from each string, beginning with the item at the top of each block. The item of greatest magnitude is transferred to an output block, beginning with the first location in that block. Each output string is composed of two blocks of data in descending order. Alternation of output tapes, and all other subroutines operate in the manner already described for the first External

cycle where collation is in ascending sequence.

It is expected that the data tapes to be collated will be the output of some previous UNIVAC operation. If so, the majority will be complete, and only the last tape in each group will contain a smaller number of blocks. The collation instruction tape has been organized to take advantage of this fact. Internal Collation instructions appear first, followed by the instructions for the ascending and descending External cycles. The instructions for the special cycle appear last. Only when an incomplete data tape requires an even number of External cycles, are the special instructions needed.

It is essential to this plan that only one block of data be collated at a time during Internal Collation. If two blocks were to be collated together, an even number of External cycles would be required to collate a complete tape. For efficiency under these circumstances, Internal Collation would be completed in ascending sequence. The first External cycle would be in descending sequence, and all even cycles in ascending sequence. The special cycle would be required to collate four blocks in ascending order following an ascending sequence. It is not certain that input instructions for this cycle could be programmed to eliminate tape interlock (see section 4.0.3).

However, if, in a specific instance, it is desirable to collate two blocks together internally, the two blocks are collated individually, and retained in the memory. Item by item, the data of one block is then compared with the data of the other, merging the items to form two blocks in sequence. The adjustments in

coding required to accomplish this are relatively simple.

4.0.3. Elimination of Tape Interlock From the Collation Process.

The foregoing discussion implies that two-way collation is the only fundamental method which may be used. To the contrary, both three-way and four-way programs have also been developed. Other than the limitations of the computer itself, the only obstacle to the use of still more-extensive systems is the consideration of comparative efficiency.

No Internal Collation program included in this report employs the three-way or four-way method completely. Only one, the program for the collation of a ten-word item, utilizes a hybrid scheme combining the two-way and three-way methods. In general, the two-way method is suitable for Internal Collation, because the number of units to be processed is comparatively small. However, the program for a particular item size may be revised to include any method or combination of methods which proves to be efficient.

The Internal Collation program which is to be followed by three-way External Collation places the sequenced data on three output tapes. These become the input tapes for the first External cycle. The routines which process the data differ from those already described only in the fact that three units instead of two are collated together. There are again three output tapes, which are now alternated after the sequencing of 3R blocks. If the first input tape contains an extra string of data, or if the first two input strings both contain an extra string of data, an

incomplete output string is created. Additional counters are required to count the blocks in input and output strings. All are treated in the manner previously explained if an incomplete string is indicated.

Alternating ascending and descending series as before, seven External cycles are required to collate a complete tape by the three-way method ($3^W \geq N > 3^{W-1}$). This is four cycles fewer than the number required to collate the same quantity of data by the two-way method. The reduction in computer operations per item, brings about increased computer efficiency. In general, similar advantages result when three-way merging is used in place of two-way merging, although the increase in efficiency is not as uniform.

Four-way collation programs have been developed recently, and none are included in this report. To establish the method in parallel with the other two, the External program is designed to collate a complete tape in five cycles. This requires two blocks to be collated together during Internal Collation. Other changes are necessitated by the large volume of data and computer instructions which must be stored in the memory at the same time. These adjustments simplify the individual routines, and lengthen them. It is not yet evident whether the reduction in the number of cycles will counteract the increased bulk of the program, in all instances.

Two other considerations affect the decision as to the method of collation to be employed. Two-way External Collation nor-

mally requires six UNISERVOs, but can be programmed with four. Three-way External Collation requires eight servos, and the four-way method employs ten. There is little choice of method if the number of servos available is limited. Furthermore, two-way collation has been programmed to avoid the delay (called tape interlock) which may accompany extensive use of the input-output mechanism. Three-way and four-way collation cannot be programmed in the same way because of the storage problems which are encountered. It is certain that the three-way method will nevertheless increase computer efficiency, but it is not certain that a corresponding increase in efficiency will accompany the four-way method.

The time required to complete a UNIVAC instruction which employs the input-output mechanism is measured in milliseconds. The time required to execute a typical instruction which employs the central computer alone is measured in microseconds. The computer has been designed to execute reading, writing and computational operations in parallel, rather than in series, because of this disparity in time.

The term "reading operation" as used in this sense refers to those read instructions which are associated with a forward or backward movement of an input tape. The instructions which only transfer data from register I to the internal memory are also, techniquely, read instructions. However, they involve no equipment external to the central computer, and can therefore be executed in a much shorter period. They are executed in series with computational instructions; not in parallel.

The time required to read 720 decimal digits from an input tape is approximately 85 milliseconds. Of this time, 3.5 milliseconds are used to establish the necessary electronic controls to execute the instruction. If register I contains data from a previous read instruction, these digits are transferred to the specified memory locations within the same period, but no other instructions can be executed within the computer. A write instruction already in progress at the moment when the read instruction is first initiated, continues without interruption.

At the conclusion of the set-up period, the reading operation is executed without further use of the central computer. The next instruction in the program is then initiated without further delay. Approximately 200 computational operations can be carried out before the last digit to pass the reading head has been transferred to register I.

When a write instruction is carried out, the sequence of events is similar. During the first 3.5 milliseconds, the necessary controls which will automatically complete the write operation are established, and the sixty words which are to be written on the output tape are transferred to the O register. At the conclusion of this time, the central computer is released for the next instruction. The writing operation continues in parallel for the remaining 81.5 milliseconds required to place 720 digits on the specified tape.

If two read instructions, or two write instructions, or a read and write instruction involving the same UNISERVO, are separated by at least 200 computational instructions, every input-

output instruction is executed without delay. If a smaller amount of computing time elapses between successive read or write instructions, the necessary controls cannot be established and all computer operations are delayed until the operation in progress is complete. During this waiting period, the computer is idle or "interlocked". The length of the idle period, "tape interlock time", may be as much as 81.5 milliseconds if the two successive input or output instructions are not separated by any computational instructions.

In collation, tape interlock on output instructions is not an important problem. Under the present manner of programming, two successive write instructions involving output data cannot be initiated in an interval smaller than that required to transfer the specified number of items to the output block. The iterative routine in which this transfer takes place is lengthened by additional instructions required for purposes of computer control. In general, therefore, the time consumed in filling the output block is sufficient to permit the previous write instruction to be completed. If the number of items per block is very small in a given program, the possibility of the occurence of tape interlock on output is increased. Interlock is always present at the time when sentinel blocks are being written on an output tape, but as this occurs only once per cycle during External Collation, and only once per tape in the other collation programs, the idle period is of little concern.

On the other hand, tape interlock on input instructions during the External Collation and Merging processes cannot be dis-

missed. Interlock does not occur during Internal Collation, because there is only one input source, and the operations which are performed during the cycles of collation require more time than that required to complete the reading of a new block into register I.

Throughout the External and Merging programs, whenever the current block of one of the input strings is exhausted, the next block from that string is read and placed in the memory for processing. If it is assumed that the next items have been placed in register I previously, the instruction executed when the block is exhausted is one which transfers the data in register I to the memory, and reads another block into register I.

At this time, the second input block may have only one item left, and the third input block may contain only two more items. Following the next iteration of the main comparison routine, the current item in one of these blocks will be transferred to the output location. The selection of the item to be transferred will depend upon the magnitude of the three items, and the order in which they are being sequenced.

If the next item to be transferred is the one remaining in the block from the second input group, a new block of data from the second input tape must be obtained. When the new read instruction is initiated, the computer is immediately interlocked, because the previous read instruction has not been completely executed. If the item transferred is one from either of the other two input groups, no input block is exhausted at this time, and

additional comparisons may be made. The block from the second input group may be empty at any time after the first transfer has been made. The block from the third input group may be empty after the second transfer has been made, or at any time thereafter. The block from the first input group cannot be empty again until all the items within it have been compared and transferred.

The amount of time required to compare and transfer five items from the input block is approximately the time required to execute a tape instruction. If five or more items are processed between tape read instructions, there is no interlock. If less than five items are transferred between tape read instructions, the computer is halted until the first read operation is completed, and the second has been initiated.

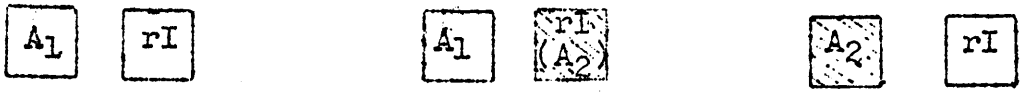
Tape interlock is eliminated by controlling the number of operations which occur during the intervals between read instructions. Where memory space permits, a method of total elimination is employed. Where memory space is limited, tape interlock is partially eliminated. Nowhere in the collation programs is memory space so restricted that no precautions against the occurence of interlock are possible.

If there is no elimination of interlock, the following sequence of events occurs: The last item in a data block is processed, and tests show the block to be exhausted. No data was previously left in register I, therefore two read instructions are required to position a new block of data in the memory. The first instruction requires 85 milliseconds. When it is concluded,

one block from the data tape has been stored in register I. During the second instruction, no tape is moved, but the data in register I is positioned in the memory. Only 3.5 milliseconds are required to accomplish this transfer. However, the second instruction must follow the first, and computation is delayed for the entire period of 88.5 milliseconds. The same delay occurs whenever a block is read from any of the input tapes.

In Figure 1 below, the three steps in this sequence are shown for one input tape. In this illustration, as in all others which are to be found in the discussion of interlock, shaded areas indicate the presence of items of data. Where more than one block from a data tape is to be considered, successive blocks are numbered in the order in which they appear on the tape; e.g., A_1, A_2 etc.

Figure 1. One Input Source; No Auxiliary Storage Locations



Step 1. Block A is exhausted, rI is empty.

Step 2. At completion of 1n instruction, rI contains A_2.

Step 3. At completion of 30 instruction, block A contains A_2.

If only one input group is concerned, interlock may be partially or totally eliminated by leaving a block of data in register I at the conclusion of each read instruction (see Figure 2). However, when two different input groups are to be processed,

this plan is not satisfactory as there is no way of predicting the group which will first be exhausted. If data from the second input group is left in register I, and the input block from the first group is the first to be exhausted, the contents of register I cannot be used. If the alternate choice is made, and the block associated with the second input group is the first to be exhausted, the situation is similar (see Figure 3).

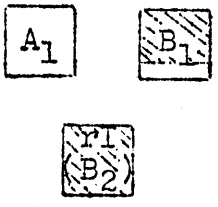
Figure 2: One Input Source; One Auxiliary Storage Location

Step 1. Block A_1 is exhausted.

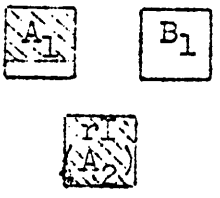
Step 2. Block A_2 is transferred from rI.

Step 3. Block A_3 is positioned in rI.

Figure 3: Two Input Sources; One Auxiliary Storage Location



Case 1. Block A_1 is exhausted, rI contains B_2.

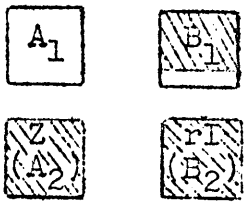


Case 2. Block B_1 is exhausted, rI contains A_2.

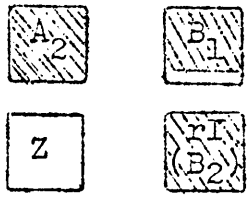
If a storage block (in addition to register I) is reserved for the input data, tape interlock can be partially eliminated when there are multiple input groups. If there are two groups, one extra block from the first is stored in the auxiliary location when the data is first read into the memory. At the same

time, one extra block from the second input group is stored in register I. When a block is exhausted, a test is made to determine the origin of the contents of the auxiliary block. If the extra data is a block from the same input group as the exhausted block, it may be transferred to the working location and processed. If it is from the second input group, the data in register I must be from the first input tape, and this data is transferred to the working location for processing. In either case, another block is read from the input tape of the exhausted block, and this data is left in register I. (See Figure 4.)

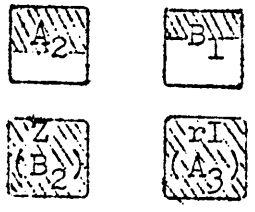
Figure 4. Multiple Input Sources; Two Auxiliary Storage Locations



Step 1. Block A_1 is exhausted.



Step 2. Auxiliary block A_2 is transferred from Z.



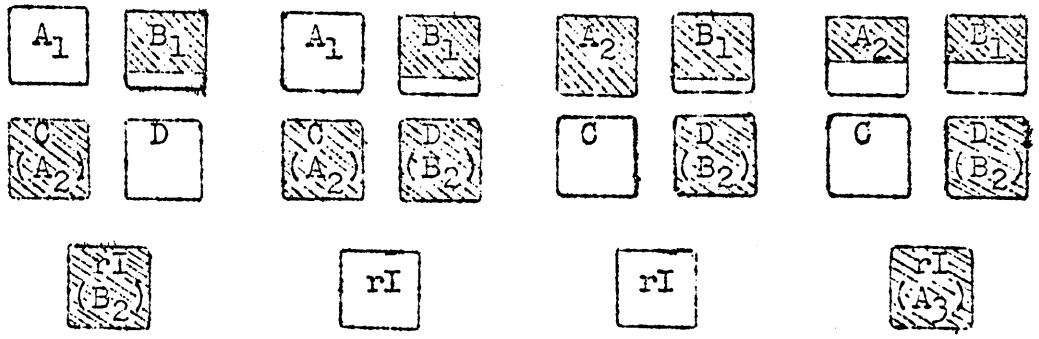
Steps 3 & 4. Block B_2 is transferred from rI to Z, and block A_3 is positioned in rI.

The disadvantage of this method lies in the fact that it is always necessary to determine the origin of the contents in the auxiliary location. If an alternate method of positioning the correct auxiliary data is used, the efficiency of the program is decreased but the associated computer controls are, to some extent, simplified.

The alternative system allocates an auxiliary storage block

within the memory to each of the input groups. Register I is also used as a storage block, making the third such location. When an input block is exhausted, the data from the auxiliary location is transferred to the input block, the contents of register I are placed in the designated location, another block from the tape concerned is placed in register I, and the data from the auxiliary location is transferred to the input block. This method of partial elimination of tape interlock is shown in Figure 5.

Figure 5. Two Input Sources; Three Auxiliary Storage Locations



Step 1. Block A_1 is exhausted.

Step 2. Block B_2 is transferred from rI to auxiliary location D. A new tape read instruction is initiated.

Step 3. Block A_2 is transferred from location C to working location A. Computation now continues.

Step 4. The read instruction initiated in Step 2 is completed and block A_3 is now positioned in rI.

Only one tape instruction is required by this method. Two milliseconds are required to manufacture the tape instruction and 3.5 milliseconds are required to initiate it. Unless there is interlock, computation proceeds after the transfer of data from aux-

iliary location to working location. This transfer requires 8.5 milliseconds, making a total of 14 milliseconds for the entire process.

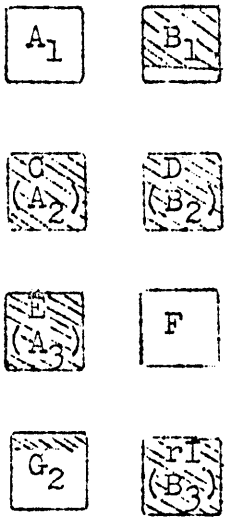
If another input block is not exhausted before 200 instructions have been executed, no interlock results. Unfortunately, when the collation process continues from this point, the second input block may be exhausted in a much shorter period. In fact, there may be only a single item, or two items, or any number, to be processed in the other block. When tests show that this second block has been exhausted, an instruction is manufactured, in order to position the data in register I and to read a new block from the correct tape. This instruction cannot be initiated by the computer until the previous read instruction has been executed. When the waiting period is over, the new read instruction is carried out, the data in the auxiliary block for the exhausted group is transferred to the working location, and collation then proceeds.

If a sufficient number of blocks are involved in the collation process, the average interlock time per block can be mathematically determined. This quantity varies for items of different sizes, but is approximately 15 milliseconds for three-way collation. It is slightly less for the corresponding item sizes when collated by the two-way method. The average interlock time is added to the 14 milliseconds previously mentioned. The total time of 29 milliseconds is the average time which elapses between the moment when an input block is first exhausted, and the instant when collation continues again with a new block of data.

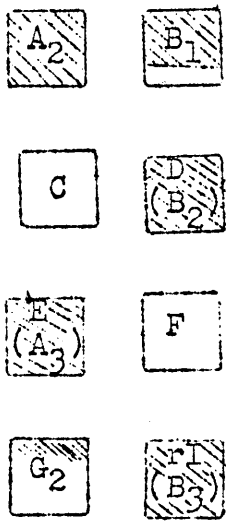
The only way in which tape interlock may be completely eliminated when multiple input tapes are concerned is by spacing the read operations by equal intervals. These intervals must be at least as long as the amount of time required to read 720 digits from a tape. In collation, the write operations occur only when an output block has been filled, and the time of completing this process is known to be longer than the time required to read 720 digits. Therefore, if an input block is read into the computer whenever an output block is written on a tape, there is no interlock. However, the mechanics of the process require the use of more storage blocks than have previously been employed. For each input group there must be as many auxiliary storage blocks in the memory as there are total input groups. Thus, for two-way collation, two storage blocks per input tape are required in addition to the two working locations and register I. This is a total of six blocks of memory space which cannot be used for other purposes. In the same way, a total of twelve memory storage blocks are required for a three-way collation process. This does not allow enough memory space for the instructions which are needed to program the problem itself. For this reason, complete elimination of tape interlock is not feasible in the three-way collation program. The two-way collation programs have been programmed to eliminate interlock.

The mechanics of the method for the complete elimination of interlock are outlined in Figure 6.

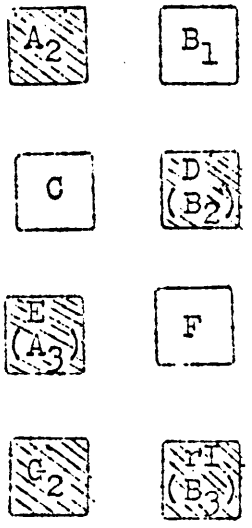
Figure 6. Two Input Sources; Five Auxiliary Storage Locations



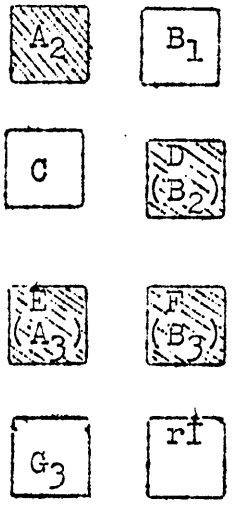
Step 1. Block A_1 is exhausted. Next tape to be read will be tape A.



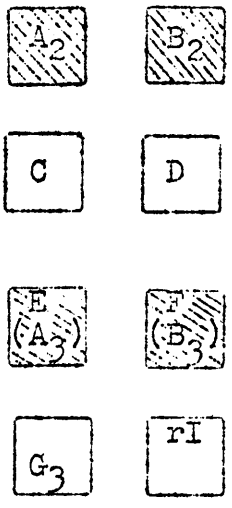
Step 2. Block A_2 is transferred from C to A. Computation continues.



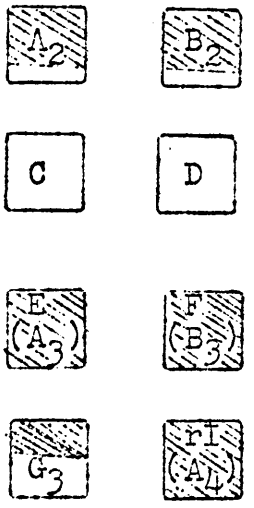
Step 3. Block G_2 is completely filled with items remaining in block B_1.



Step 4. Block G_2 is written on output tape; block B_3 is transferred to F; a new read instruction from tape A is initiated.



Step 5. Block B_1 is now discovered to be empty. The next tape to be read will be tape B. Block B_2 is transferred from D.



Step 6. Computation continues with the items in A_2 and B_2. When the current read instruction is complete, rI contains block A_4. When the next output block is completed, A_4 will be transferred to C.

As the two-way External Collation and Merging programs now stand, the elapsed time between the emptying of an input block, and the moment when collation can continue is 15 milliseconds. Of this time, nine milliseconds are required to determine the identity of the storage block from which the new data is to be obtained, and to transfer that data to the working location. An additional 1.5 milliseconds is required to fabricate the read instruction which will be used next. At the moment of initiating this read operation, another 3.5 milliseconds are required for the instruction. One additional millisecond is used later in adjusting the address of the next block of data.

In the total elimination of tape interlock as accomplished in the two-way programs included in this report, three end-of-tape sentinel blocks are required for each input tape. When a data tape is finally exhausted one sentinel block may be in the working location, a second in one of the storage locations, and the third may be unread, or may have been placed in register I. When fewer auxiliary storage locations are employed, as in the three-way program, only two sentinel blocks are needed. All the Merging programs have been planned with a view to combining two-way and three-way methods, and therefore, all External programs provide three sentinels for the final output tape, as do all three-way Merging programs.

4.0.4. Flexibility of Classification of Data.

Theoretically, data to be used in UNIVAC operations may be placed on the original input tape without regard to the number of words or fractions of words to be grouped together as an item. In order to reduce computing time, however, it is desirable to introduce as much uniformity as possible. The demands of a given problem will necessarily dictate the minimum number of digits for each item, but the data is inherently flexible, and can be expanded to any reasonable limit. Therefore, a short period should always be used to analyze the optimum item length for both the data concerned and the programming required to process it.

In general, more computer time is wasted in processing an item which does not include an integral number of words, than is saved by conserving the number of digits to be operated upon. Such items are awkward, and impractical to process at all; impossible to sort or collate. For this reason alone, it is preferable to plan the use of items in integral word lengths, unless a pre-processing program is to be used. A program of this type edits the item, expanding it (by the inclusion of additional digits) or contracting it (by combining several quantities) in order to provide a more workable unit for future operations.

In some instances, the data under consideration may be typed directly on the input tape in completely uniform item lengths. In many, however, the nature of the data will indicate the use of a unit which varies in length from item to item. There is no difficulty in processing variable length items, if the initial word of

each is clearly identified, and there is no confusion concerning the contents of the words which expand a particular one beyond the basic length common to all. Presumably, it will not be necessary to sort or collate data of this type prior to the first computing or pre-processing operations which transform the data into items of uniform length. However, if collation is the first operation to be performed, the initial subroutine of the Internal Collation program must be one which expands each item to a size sufficiently large to include the longest, and at the same time ascertains that the key digits assume the same position in each item.

The collation routines were planned to indicate the method of handling data in a wide variety of item sizes. Except for a few, (see Chapter 5 on Internal Collation) which have been specifically designed to internally collate a certain item size in an economical fashion, all may be easily adjusted to accomodate any size of item from one word to sixty words. However, not all intermediate sizes are practical to use. In iterative routines such as are employed in the collation programs, the simple maneuvering of an eight-word item from one memory location to another requires nine instructions to build the transfer instructions, and eight more to accomplish the transfer itself. By expanding the item to include two more words, the same results may be achieved in three and two instructions, respectively. The additional quantity of tape required is insignificant compared with the saving in computing time. Similarly, the use of items in the size range between 13 and 19 words is poor practice because of the great economy made possible by the use of a 20-word item.

Items greater than 720 digits in length may be processed in the normal fashion, although additional read and write orders are required between the iterations of the main comparison routine. The collation programs included in this manual, however, make no allowance for items larger than one block in length. Additional programming is required to make the necessary adjustments.

When data is to be sorted on one or more criteria, certain digit locations within each item must be designated to represent the criteria. These are called key digits, and may be as many or as few as necessary as the case requires.

Some key digit locations are more practical to use than others. If all can be assembled together at the extreme left of any word in the item, comparisons may be made with the contents of the entire word. The resulting sequence of data is in order by satellite digits of the key words as well as by key digits, but the additional refinement is not harmful. If the data does not have enough flexibility to permit the key digits to be assembled together at the left of a word, they may be placed anywhere within the item and extracted into register A for comparison. This is more time-consuming than the previous method, and becomes even more wasteful if the digits cannot be assembled at all, but must be extracted from different words.

Variations in the efficiency of processing also depend upon the location within the item of the word which contains the key digits. These result from the iterative process used to transfer the item from its input location to an output location, rather than from anything inherent in the method of comparison. A com-

bination of both efficiency considerations gives the following list of key digit locations in descending order of desirability.

1. Entire first word, or the digits assembled at the left of the word.
2. Entire last word, or the digits assembled at the left of the word.
3. Any other complete word, or the digits assembled at the left of the word.
4. Key digits assembled within the first word, but not at the left.
5. Key digits assembled within the last word, but not at the left.
6. Key digits assembled within any other word, but not at the left.
7. Key digits divided between two or more words.

The collation and merging programs will process numeric, alphabetic, or mixed numeric and alphabetic characters. Where mixed digits are to be used, however, it must be remembered that UNIVAC always considers numeric digits of smaller magnitude than alphabetic ones.

Data which will not exceed one tape length in quantity after processing, does not require the establishment of a complicated system of tape identification. A simple paper label externally attached (by the operator) to the final reel will adequately indicate the contents of the tape and the original data source for future reference. On the other hand, when large quantities of data are to be collated, or when items already collated are to be further processed without destroying their sequence, an efficient system for maintaining the serial order of the reels is necessary.

The method chosen must provide both internal and external labels for each tape. The external label, which is attached to the reel, need only indicate the contents of the tape and its position with relation to other tapes in the series. The internal label, which is part of the tape itself, must, in addition, supply sufficient information to make a rerun of the tape possible, if required.

In the tape identification system planned for the programs included in this report, each uncollated data tape is initially assigned a four-digit label, according to a master plan maintained by the person in charge. The label consists of two alphabetic characters chosen such that no pair is exactly duplicated, and the numeric digits 01. It is first introduced at the beginning of the Internal Collation of the given data tape, when it is typed into a specified memory location by the operator at the Supervisory Control keyboard. The label remains in this location throughout the collation process, undisturbed by changing instructions. At the beginning of the last External cycle, it is placed in the first or identification block on the output tape, and the collated data follows in the succeeding blocks. The same number is also placed on the reel by the operator. No additional identification is necessary at this time, as the master plan indicates the original data tape, from which the collated tape was derived.

During the first Merging cycle, the collated tapes are arbitrarily merged into strings of either two or three (depending upon whether two-way or three-way merging is used) and the resulting reels of each string are assigned new identification numbers, con-

sisting of another pair of alphabetic characters, accompanied by digits 01 and 02 (if two tapes) and 03 (if three tapes). Output tapes of successive cycles are treated similarly. Still another pair of characters are assigned to each string, with associated digits beginning with 01 for the first reel, and increasing by 01 for successive reels, until all the reels in the string have been uniquely numbered in a manner which distinguishes them from each other, and from those of every other string. Ultimately, the data is arranged in monotonic sequence on a single string of reels, and the reels are serially numbered to indicate the order in which they should be considered. All identification labels are alotted according to the master plan, and therefore, the necessary information with regard to those tapes comprising the input strings for a given output string will be immediately available.

Throughout Merging, the first block on every output tape continues to be an identification block, containing the tape identification label. Furthermore, this block now contains the labels of the input tapes involved, and enough additional information to locate the first pertinent item from each input tape concerned with the current tape. Information regarding the labels for input and output tapes is given from the Supervisory Control at the beginning of every string, and is kept current by a counting process within the program. Thus, the identification block on any tape contains enough information to identify the tape, and to rerun it if necessary. Agreement between the internal and external identification numbers is maintained by having the computer issue (by means of the

Supervisory Control printer) a paper label to be attached to the completed output tape. This label also contains the keyword of the first item.

The labelling system which has been described limits the number of tapes in a given string to a maximum of 99. In actual practice, it may be necessary to allow three digits for the numerical part of the tape label in order that longer strings may be handled. Additional changes in the system will be required if there is a possibility that the data will include more than one block of items containing the same key digits. The present method locates the first item on an output tape by identifying the input tapes, the current working location within each respective data storage block, and a keyword known to be equal to, or less than, the first on the output tape. If the same location within any of several successive blocks on a given input tape may contain the keyword, the specific block concerned should be designated by number. This would require additional programming to count the input blocks from each input tape as used.

4.0.5. Automatic Operations Performed During the Collation and Merging Programs.

To combine maximum computer efficiency with a minimum of human error, the collation and merging programs are designed to be almost entirely automatic. Supervisory Control input and output instructions are used only when necessary to increase the flexibility of the program, or to aid the operator in preventing the occurrence of error. After the initial setting, the switches

associated with these instructions may be ignored, as no changes are required during operation. If normal precautions are used to ascertain (1) that the correct input tapes are mounted on each of the specified UNISERVOs, (2) that correct disposition is made of output tapes as they are removed from the UNISERVOs specified, and (3) that the programmed interruptions are correctly interpreted, collation and merging should proceed rapidly and without error.

As now programmed, the Internal and External collation processes are interrupted only once, normally. Cessation of operations soon after completion of the initial read instructions permits the operator to insert a designated tape label (identification number) in the memory. When the word has been typed and released, computer operations are resumed, and continue without further intervention until the data tape has been completely collated. The following partial list indicates the type of automatic operations performed during this period of computation:

1. After the initial starting procedures have positioned the first block of instructions within the memory, all others are read from the instruction tape as needed and are positioned according to a preassigned plan. More than one group of instructions are required to complete the processing of a single tape, but all possible groups may not be required for the collation of a particular tape. The instruction tape is searched for the correct group of instructions when a new one is required.

2. Similarly, the original data tape is read into memory, one block at a time, as needed, and the collated data is written on an output tape, one block at a time, when ready.

3. The collation process as described earlier in this chapter embodies many techniques which are programmed to take place without the aid of the operator. These include: iteration of comparison and transfer routines as frequently as necessary, adjustment of counters for the collation of incomplete strings, alternation of tapes in order to obtain approximately equal groups, determination of end of tape, adjustment for incomplete block, determination of the number of blocks to be collated, and from this determination of the number of External cycles required to complete collation, determination of the number of units to be included in input and output strings, and provision for tape identification.

4. When collation is complete, all tapes are rewound in order that they may be reused, or removed from the servos. To eliminate confusion regarding the location of the collated output tape, the number of the servo containing the final reel is indicated through use of a Supervisory Control output instruction.

When only four UNISERVOs are available, the efficiency of the collation program is somewhat reduced. Servos used for instruction and data tapes during Internal Collation are required for output use in the External cycles which follow. These UNISERVOs are available only after the remaining instructions have been positioned in the memory and the two original tapes have been rewound and replaced. Computer operations proceed in parallel with rewinding, but are soon halted by the need for an output servo which is still interlocked. The delay continues until a new tape has been mounted on the servo, and the interlock is released.

With this one exception, there is no difference between the automatic operations included in the above program, and those previously described. The role of the operator is more demanding when the number of UNISERVOs is limited, but it is of importance primarily because of the time element involved. There is little concern that serious errors might be introduced at this time.

Supervisory Control input instructions are required at the beginning of every run of the Merging program, to indicate the number of tapes in each input string and their identification letters, as well as to provide the identification letters for the tapes in the output string. When these items have been positioned within the memory, Supervisory Control output instructions give the operator an opportunity to verify the input information. Because it is imperative that the serial order of sequenced tapes be maintained, additional Supervisory Control output instructions are employed at other points in the program to aid in checking the identification of incorrect input units, and to provide a printed label for each output tape.

The increase in processing time which accompanies the use of these techniques is justified by the resulting increase in overall flexibility and accuracy. On one hand, a single program can be used for the merging of varying numbers of tapes. On the other hand, the operator is able to verify the input identification information for each run, to locate the source of error when input tapes have been incorrectly mounted, and to correctly label each output tape. Furthermore, a routine established within the pro-

gram tests every new input tape to determine whether it is the next in sequence and provides an error indication if it is not. Without this additional verification procedure, an input tape might be mounted out of sequence and the resulting error would not be detected until many hours of computer time had been wasted.

When input tapes have been correctly mounted, the Merging program proceeds without further interruption until a new tape is brought into use, and proves not to be the next in its string. If no errors are made in mounting input tapes during an entire Merging run, processing is undisturbed until the output string is complete.

The automatic operations included in the Merging program are similar to those noted for collation with the following exceptions:

1. The instructions for Merging are fewer in number, and are not separated into different groups which must be positioned at intervals during the processing.

2. Not all of the collation techniques which result in automatic operations are included in the Merging program. On the other hand, the tape identification system maintained without operator aid during Merging is more sophisticated than that used in the collation programs.

3. Not only is the servo number printed when an output tape is available, but also the paper label which must be attached to the reel.

4. The Merging program includes a method of checking the accuracy with which the operator fulfills his duties, as explained above.

5. The Merging program is easily adjusted to variable length input strings.

A more extensive instruction tape containing an entire library of collation and merging programs for every item size desired might be employed to make this type of sequencing even more automatic. The initial instructions on such a tape would call for Supervisory Control to indicate the group of instructions desired and then would read past successive blocks until they had been positioned. Thus, a single instruction tape could economically replace several smaller ones. As the programs now stand, only minor adjustments in coding would be required to eliminate the rewinding of the instruction tape following collation and merging. The routine substituted would position the first block of the selected instructions again, for the next UNIVAC run.

The same efficiency considerations which make automatic operations desirable in the collation and merging programs, make them desirable when rerun problems must be solved. Rerun is a term applied to the operation of the UNIVAC for the purpose of reprocessing a data tape which has been incompletely or unsatisfactorily processed previously. Precautions may be taken against many types of failures or errors which may interrupt a program but not all can be entirely eliminated. Since it is not possible to resume operations from the point where discontinued, the development of economical reprocessing techniques is desirable.

The most unpredictable type of failure is a breakdown of the computer or any of its associated units. When this occurs,

operations are halted for repairs or replacements, and the output of the program in progress is useless. Similarly, an imperfect magnetic tape mounted on an output servo, will cause a cessation of computer operation when it is used as an input tape for another program. This is a two-fold problem, as both the current input and output tapes are valueless in this instance.

A third source of possible error in the processing of data is the failure of the operator to fulfill some part of his responsibility correctly. Even when UNIVAC operations are almost entirely automatic, the operator is required to set certain switches, type an occasional word of information into the computer, and to mount and dismount tapes as necessary. Not all of these operations can be checked within the program, and it is impractical to assume that errors will never be made.

Finally, the processing of data may break down because of incorrect programming. There may be small errors in coding, in the typing of instruction tapes, or in the original analysis of the program. An effort is made to eliminate errors of this type but it may not be completely successful.

The following general procedures are of use in any type of programming, either to guard against failures which may be foreseen, or to minimize the resulting wastefulness of those which may not. The last technique is the one which is of particular value in making the Merging programs automatic.

1. All programming is checked carefully and thoroughly before converting it into an instruction tape.

2. The original instruction tapes and data tapes are duplicated, in order that the necessary information for rerunning may be readily available, if one tape proves defective.

3. The UNITYPER provides a printed copy of all original tapes as they are made. This is used to check typographical errors which may have gone unnoticed in transcribing the data or instructions.

4. Before a program is used, it is tested with a special data tape which is known to contain all variations inherent in the actual data. By coding all Qm and Tm instructions in the program as Qnm and Tnm instructions, and depressing the appropriate Transfer Breakpoint Selector Buttons, the UNIVAC is stopped at each test point. At this time, the Supervisory Control indicates whether a transfer of control is to be made, and knowledge of the data being operated upon will approve or disapprove a transfer at this time. If the result of the test is not as expected, the programming must be revised.

5. A rim, manually inserted in the reels which are currently being used as input reels, prevents them from being used as output reels until the rim has been removed. Under these circumstances, it is impossible for the data tape to be destroyed by writing upon it, either through an unpredictable error in the computer's circuits, or because of a careless operator.

6. In any program involving large numbers of tapes which must be processed in serial order, it is practical to assign identification numbers to these tapes. After each new tape has

been mounted, a subroutine within the program may accept it or reject it, as the next in the series.

If the collation process is interrupted at some point for any of the reasons discussed, it does not necessarily mean that an entire program, involving many intermediate tapes must be repeated, from the beginning. However, if the processing has not progressed beyond Internal Collation, the simplest rerun technique is to begin again. If the breakdown occurs during External Collation, under the present manner of coding, all the Internal and External Collation procedure for the data tape in question must be repeated. The necessary information concerning tape identification and the various counters affecting control is not available at any intermediate point.

A slight modification in the present programming would improve this situation. If the last block written on an output tape at the conclusion of Internal Collation and of each cycle of External Collation were to contain the tape label, and the values of r,s,t,u,v,w,R, 3R (or 2R), U, and H as established for the next cycle, the largest difficulty would be eliminated. In addition, it would be necessary to indicate (on the Supervisory Control) the input servo numbers at the beginning of each cycle, and to revise the External program to rewind all output tapes to the leader before using them as input tapes. A special program would then be used immediately after the resumption of UNIVAC operations following a breakdown. In this program, the required information would be read from the designated input tape and placed in the

proper memory locations, the appropriate instructions for the next cycle would be determined and positioned, and finally, the necessary adjustments in the External program would be made. Upon transfer of control to the first instruction of the new cycle, the collation process would continue at the beginning of the cycle in progress at the time of breakdown.

Use is made of this general approach in planning an automatic rerun technique which will operate in conjunction with the Merging programs. The first, or identification block of every tape used in the process contains all the information necessary to reproduce that tape alone, if required, or to continue the process beginning with that tape if so desired. If the tape to be rerun is the first of a series, the use of a special routine to adjust controls in the normal Merging program is unnecessary. The rerun may be accomplished by starting the series in the normal fashion, and continuing until the tape(s) have been rerun as required.

Identification information for a merged tape is stored in memory locations designated by q_1, q_2, q_3 etc. during the Merging program. At the start of an output tape, certain of these locations indicate the input tapes in operation at the moment and the current working location within each of the respective data storage blocks. Another location stores either the first keyword on this output tape, or the last keyword on the previous output tape. This information is written in the identification

block of the output tape, and the merged data follows in the remaining blocks. If a breakdown occurs before or after completion of the tape, a special program reads the information from the identification block, calls for the necessary data tapes, advances each to the point where the specified working locations in the current block contains a keyword equal to or greater than the one indicated in the identification block, places the normal Merging instructions in the memory, makes the adjustments in instructions required, and transfers control to the starting location of these instructions. When a single tape in a completed series is unsatisfactory, rerun is required for one tape only, and is terminated when that tape is complete. If the Merging process is interrupted before completion of a string, the rerun routine is inserted and Merging can proceed to the end of the string.

CHAPTER 5

INTERNAL COLLATION

5.0.1. Introduction.

Internal Collation is the first step in the complete collation of a tape containing items arranged in random order. The Internal Collation program considers each successive block of data independently, and arranges the items within it in descending monotonic sequence. At the conclusion of the process, every block is completely collated, but the various blocks remain in random sequence with respect to each other.

Because of the nature of the program, and of the ones to follow, it is mandatory that all items be of uniform length, and that all blocks contain the same number of items. If the information which is to be collated consists of variable length items from original data tapes, the Internal Collation program must include the necessary routines for converting all items to a uniform size. If, as is more likely, the tapes to be collated are the output of a previous processing or editing run, these routines need not be included. In either case, if it is not certain that the last block on all input tapes is completely filled with data, the program must provide a routine for completing a partial last block. The artificial items which are inserted must not interfere with the sequencing of the true data.

The general Internal Collation method is one which closely resembles the basic two-way plan described in Chapter 3. The anal-

ogy is most obvious when a block containing two, four, or eight items is to be processed. The corresponding items sizes of 30, 15 or seven words are not the most typical, however, and they represent only a small portion of those available for use. On the other hand, if the items to be processed are ten words in length, each block contains only six items to be collated together, and the established pattern cannot be maintained after the first cycle. Similar deviations, either during the first, or in a subsequent cycle must be introduced whenever the quantity of data is not a perfect power of two.

To meet this situation, three modifications of the basic plan have been devised. The first of these arbitrarily expands the number of items being processed to a quantity which will fit the existing program. By the addition of words containing appropriate non-data symbols, six or seven items are considered as eight, 15 as 16, and so on. When the block is collated, the necessity for the additional items is terminated, and only the original ones are transferred to the output tape. No permanent change in the total quantity of data is therefore effected.

A second adaptation makes adjustments in computer controls, rather than in the number of items to be processed. Because $2^W = N$ is no longer an exact expression of the relationship between N, the number of units to be collated, and W, the number of cycles required to collate them, it is replaced by the expression $2^W \geq N > 2^{W-1}$. The difference between the number of items to be processed, and the next perfect power of two is adjusted by allow-

ing some of the strings to have less than the proper number of units. These are partial or incomplete strings.

During the first cycle, each input string consists of only one item. If an even number of items are to be collated, all the data is included in one of the two groups. If the number of items to be collated is odd, one item is excluded from both groups until a later cycle, when it becomes part of an incomplete string.

At the beginning of the second and all other cycles, the input data is again divided into two groups containing an equal number of strings. If the total number of strings is odd, part of the data is excluded from both groups during that cycle, but is again included in group two at some time prior to the completion of the last cycle. Whenever a string which has been omitted for one cycle re-enters the data group, it is an incomplete string. Its components are included among the components of the incomplete string which is present in every cycle thereafter until collation of the block is complete. For example, seven items are grouped in the following way for each of the three cycles required to complete collation:

Cycle 1:	Input:	Two groups of three items, seventh item not included in either group.
	Output:	Three strings of two items, one string of one item (partial string).
Cycle 2:	Input:	One group of four items, one group of three items (second string of group two incomplete).
	Output:	One string of four items, one incomplete string of three items.
Cycle 3:	Input:	One group of four items, one group of three items (group two string incomplete).
	Output:	One collated string of seven items.

The third modification deviates from the basic two-way method and eliminates the necessity to define groups, strings, or cycles. The correct sequence of the items is determined by direct comparison of every item with every other one.

The first of these adaptations has been discarded because it increases the number of comparisons unduly. The third has been discarded because it is an unsatisfactory method for processing items whose key digits are divided between two words. Only the second adaptation is a general method of Internal Collation which may be used for many different item sizes. Nevertheless, it is desirable to analyze every collation problem, in order to determine whether another method, or combination of methods may not accomplish internal processing more economically.

5.0.2. The Basic Two-Way Method of Internal Collation.

A completely standardized program for Internal Collation by any general two-way method does not exist. Each change in item size introduces variations in the number of items to be included in each cycle, and in the number of cycles required to complete Internal Collation. These variations, in turn, require changes in computer controls, which are reflected in flow chart and coding.

The tables which begin on page 5.1.1.1. indicate the recommended number of items to be included in each group in each cycle, for several different item sizes. When the manner of handling a particular item size has been determined, the end-of-cycle routines can be prepared, and the remainder of the program then follows an established pattern. The keyword of item one, group one,

is compared with the keyword of item one, group two, and the larger, with its satellite information is transferred to the first output location. The smaller with its satellite information is transferred to the second output location. Continuing in this manner, successive items in group one are compared with those in group two, always transferring the one of greatest magnitude first. When the last item in each group has been transferred, the output is a series of two-item strings, each properly collated, plus the item, if any, which was omitted from both groups. This completes the first cycle.

At the beginning of the second cycle, the data is divided into two groups with equal numbers of strings. The last string in group two may or may not be incomplete, depending upon the number of items to be collated, and on whether there was an excluded item in the first cycle. After collation, the output is a series of complete four-item strings plus a partial string of fewer items. At the end of each additional iteration of the routines which comprise a cycle, the number of items in a complete string is doubled. The final iteration, or cycle, yields a single output string containing all the items in descending monotonic sequence.

The length of the input string in each cycle is regulated by the quantity R, which is defined as the number of items from each group to be included in a single string. During successive cycles, R is equal to one, two, four, eight, etc.. The total number of items in the output string is controlled by the quantity 2R, which varies in a corresponding manner.

The number of items transferred from a group to the output string is controlled by two tests made after each data transfer. The first test determines whether the item just transferred was the last in its group. If the item just transferred was the final item, and this group is the first to be exhausted, the cycle is completed when the items remaining in the other group have been transferred. If this is the second group to be exhausted, the cycle is already complete, and adjustments for the next cycle are made. If the item transferred was not the last in its group, a second test determines whether R items from the current string in the group have been transferred. If so, and this is the first of the current pair of strings to be exhausted, the remaining items from the current string in the other group are transferred to complete the output string. If this string is the second of the current pair to be exhausted, the output string contains 2R items, and the next comparison of an item from each group begins a new output string. If the item transferred did not exhaust the current string in its group, processing is continued with the next item.

The routines entered at the conclusion of a cycle prepare controls for the operation of the next cycle, and transfer to reiterate the main comparison and transfer routines again. The number of cycles required is controlled by substituting alternate instructions at the end of each cycle. At the conclusion of the final cycle, the controls for the first cycle are reestablished in order to process the next block of data in exactly the same manner.

When one block has been completely collated, it is written on an output tape. The collated data must be divided into approximately equal groups for the External cycles which follow; therefore, the output tapes are rotated after each block is written. The first block is written on tape three, and the second on tape four. If three-way External Collation is to be used, the third block is written on tape five and the fourth on tape three again. If two-way External Collation is to be used, the third block is written on tape three, and the fourth on tape four. If it is not possible to divide the data into exactly equal groups, this procedure assures that the larger group will always be on the first tape (or the first and second tapes, if three-way collation will be used) and that the additional quantity will be no more than a single unit. This plan is of use in establishing the computer controls for External Collation.

After a collated block has been written on an output tape, another block of uncollated data is read from the data tape, and tested for the presence of the end-of-tape sentinel. If this block contains sentinel and data as well, it is a partial block. Words of ignore symbols are then substituted for the missing items, and the block is processed in the usual manner. After it has been written on the output tape, the closing routines of the program are completed. If the input block contains only sentinel and no data, every block has been collated, and the closing routines are entered directly.

The closing routines of the Internal program set the necessary counters for External Collation, determine the number of External cycles required, read a block of instructions for the first External cycle into memory, and a second into register I. Control is then transferred to the next program.

5.0.3. Procedures for Handling Items of Different Sizes.

Items which are one or two words in length are collated in the manner just described. Because memory space is not at a premium in this program, the locations designated to receive output data during one cycle are used as the input working locations for the next. Computing time is saved by the elimination of the necessity for transferring the output of one collation cycle back to a standard working location for the next.

Following each data comparison, an iterative routine is used to transfer the item with keyword of greater magnitude to the output location. For a two-word item, only three instructions are required to accomplish this operation. For a three-word item, the following lines of coding are required to fabricate the necessary instructions and to effect the transfer.

T_1	F	B (m)
T_2	E (x)	H (T_5)
T_3	A (c)	H (T_6)
T_4	X	C (T_7)
T_5	B (A)	C (G)
T_6	B (A+1)	C (G+1)
T_7	B (A+2)	C (G+2)

Legend: Memory location
A = current location in data block.
G = current location in output block.
$T_1 \ldots T_7$ = locations containing instructions for transfer routine.
c = location of the constant 000001 000001
m = location of instruction containing G.
x = location of instruction containing A.

Using V and W instructions where possible, a six-word item may be transferred in seven lines of coding also. A nine-word item, however, requires ten lines to complete the comparable operation.

Items which are three to nine words in length may be more economically collated if repeated data transfers are eliminated through the use of a function table. The memory locations of the item keywords are stored in a group of standard working locations designated as the input function table. The items included in each group during each cycle are specified in terms of function table locations, rather than in terms of data locations. Comparison instructions are fabricated, using the data storage locations contained in the current working positions of the function table and the two keywords are compared. The address of the keyword of greater magnitude is transferred to the first output function table location. The address of the keyword of smaller magnitude is transferred to the second output function table location.

Successive items are compared and sequenced in the same manner without disturbing the contents of the data storage block. When the first cycle is complete, adjustments are made for the second, and the contents of the output function table are transferred back to the input function table working locations.

In the cycles which follow, processing continues in the same manner. When the last cycle has been completed, the final output function table indicates, by keyword locations, the order in which the items should be arranged to obtain the desired monotonic se-

quence. At this time, transfer instructions are fabricated from the function table, and the items of data are placed in the correct order in the output block. After the output block has been written on the current output tape, the original input function table is replaced, and the next block of data is collated.

This application of the general method is more efficient because it eliminates many of the multiple transfer instructions required for each item of data. For example, the 14 instructions usually required to transfer a six-word item, are reduced to six instructions when input and output function tables are incorporated. Even though six additional instructions are required to fabricate comparison instructions from the function table, two instructions per item-transfer are eliminated. When the corresponding reduction in computing time is multiplied by the number of transfer per cycle, the number of cycles per block, and the number of blocks per tape, the total saving is small but important. The saving is increased when one string or group is exhausted and the comparison routine is not required. The saving is somewhat reduced by the time required to fabricate output instructions once per block.

The sample routines below indicate differences in the coding required for each of these two procedures.

Example I: Typical comparison and transfer instructions, not using a function table. Register F already contains the digit extractor.

Q_1	E (A)	K 000
Q_2	E (B)	T
T_1	F	B (m)
T_2	E (Q_1)	H (T_5)
T_3	A (c)	H (T_6)
T_4	X	C (T_7)
T_5	V (A)	W (G)
T_6	V (A+2)	W (G+2)
T_7	V (A+4)	W (G+4)

Legend: Memory location
A = current location in group one data storage block.
B = current location in group two data storage block.
G = current location in output storage block.
Q_1, Q_2 = locations containing comparison instructions.
$T_1 \ldots T_7$ = locations containing transfer instructions.
c = location of constant 000002 00000
m = location of instruction containing G.

Example II: Typical comparison and transfer instructions, using the function table adaptation of the general method. Register F already contains the digit extractor.

Input function table L: $L_o = A_o$
$L_1 = A_6$
$L_2 = A_{12}$
$L_3 = A_{18}$
$L_4 = A_{24}$
$L_5 = A_{30}$
$L_6 = A_{36}$
$L_7 = A_{42}$
$L_8 = A_{48}$
$L_9 = A_{54}$

Q_1	B (L_o)	A (q_4)
Q_2	C (Q_4)	B (q_5)
Q_3	A (L_5)	C (Q_5)
Q_4	E (A_o)	K 000.
Q_5	E (A_{30})	T
T_1	F	B (y)
T_2	E (Q_1)	C (T_3)
T_3	B (L_o)	C (Ly_o)

Legend: Memory location
$A_o \ldots A_{39}$ = storage location for one block of six-word items. Keyword is first word of each item.
$Ly_o \ldots Ly_9$ = storage location for output function table.
$Q_1 \ldots Q_5$ = locations containing comparison instructions.
$T_1 \ldots T_3$ = locations containing transfer instructions.
q_4 = location containing E 000 K 000.
q_5 = location containing E 000 T
y = location of instruction containing current posi-

Items which are ten words in length may be collated in the same way. However, a procedure permitting greater economy is available for this particular item size. The block to be collated is first divided into two groups of three items each. Direct comparisons within each group are made, until two strings of three items in sequence are obtained. The items of the two strings are then merged in a single string of six items, using a normal two-way comparison. In this way, the processing is reduced to two cycles of operation. The instructions for the first cycle are extensive, in order to provide for every possible sequence of events, but memory space in Internal Collation is expendable.

The economy introduced by this third variation on the general method, is due to the reduction in the number of cycles of operation. No additional saving in computing time can result from the incorporation of function table procedures as well. When ten-word items are to be transferred within the memory Y and Z instructions, rather than multiple transfer instructions, are employed. The manipulation of the function table is therefore less efficient in this case.

The three programs included in this chapter of the collation report have been programmed according to the efficiency considerations just discussed. When items of other sizes are to be collated, it is expected that additional combinations of methods may be developed, which will further increase computer efficiency in a particular instance.

5.0.4. Detailed Description of Internal Collation Program 940-4 (Section 5.1.3).

The initial read operation places one block of instructions from tape number one in memory locations 000 through 059. At the start of the program, the remaining instructions and constants are read and positioned in the designated memory locations. To prepare output tapes three, four, and five for use with backward read instructions, two sentinel blocks are written on tapes three and five, and three sentinel blocks are written on tape four. The tape label or identification number, is received into memory from the Supervisory Control keyboard and is stored in a location where it will not be disturbed throughout this program or the next. Finally, the counter employed to count data blocks as they are processed, is set to zero.

The iterative collation process is initiated with the reading of a block of data into memory locations A_0 through A_{59}. The last location, A_{59}, is tested for the presence of sentinel to determine whether the current block is the last. If sentinel is present in A_{59}, control is transferred to a routine which tests other positions in the block for the end-of-tape indication. If sentinel is not present in A_{59}, the current block is a complete block of data, and the main collation routine is entered.

The original function table containing the locations of the ten data keywords is transferred to the input function table working location, L, and the controls for the first cycle are established. The keyword locations are divided into two groups of five "items" each. The function table locations of the last address in

group one and group two, L_4 and L_5 respectively, are designated as J and K. The function table locations of the current (first) "item" in the two groups, L_0 and L_3 respectively, are correspondingly designated as j and k. The current location, y, in the output function table, Ly, is set to the first location, Ly_0. The counter, f, which records the number of blocks collated, is increased by one. The counters a and b, which count the number of items from the current input strings in group one and group two respectively, are set equal to zero. The limit with which each is compared, R, is set equal to one.

The actual data comparisons are accomplished by fabricating instructions from the data storage locations in the function table. The locations contained in the current function table working positions are added to basic comparison instructions containing no memory locations. The routine which is thus created compares the contents of $A_{(k)}$ with $A_{(j)}$ to determine the one of greatest magnitude. If the keyword in the location contained in j is greater than, or equal to, the keyword in the location contained in k, the location in j is transferred to y. The output location y is then advanced in preparation for the next "item". Correspondingly, the input location j is increased by one, and compared with its limit J, to determine whether group one has been exhausted. If the group has been exhausted, the program is adjusted to complete the current output string with the "items" remaining in group two.

If group one has not been exhausted, the counter, a, is increased by one, and compared with R. During the first cycle, R is one, and therefore, the string is completed after the transfer of

a single "item". Adjustments are made in the program to indicate that the current output string will be completed with an "item" from group two. The routine which will transfer the group two "item" to the output function table is then entered.

When the address of k has been transferred to y, the output location is again increased. The counter k is increased by one and compared with K, to determine whether group two has been exhausted. Group two cannot be exhausted at this time, as group one has not yet been exhausted, and incomplete strings, if any, occur in the second group. When tests show that group two has not been exhausted, the counter b is increased by one, and compared with R. Only one "item" from this group is needed to complete the current output string, and therefore, the end of the input string has been reached. The routines adjusted at the conclusion of the group one string are now restored to their original status, in preparation for the processing of a new string.

If the first comparison shows the keyword in the address contained in k to be of greater magnitude than the one in the address contained in j, the series of events is similar, but the address in k is transferred to the output function table first. The address in j is then transferred directly to the second position in the output table. After the two transfers take place, regardless of the order in which they occur, control is transferred to the routine which resets a and b to zero and initiates the comparisons for a new string.

Four more iterations of these routines are required. During the last iteration, when tests indicate that one of the two groups has been exhausted, adjustments are made in the program in order that the end-of-cycle routines may be entered when the other group has been exhausted. When the second of the two groups has been exhausted, the original connectors are reset, and control is transferred to the routines which establish the controls for the next cycle.

In summary of the above processing operations, an analysis of the variable connectors associated with the program may be of use. The connectors which apply to alternate paths in the processing of group one are numbered J6, J7 and J8. The parallel routines, which apply to the processing of group two, are numbered K6, K7, and K8. Initially, ^{o}J6a, ^{o}J7a, ^{o}J8a, ^{o}K6a, ^{o}K7a, and ^{o}K8a are all set.

When an "item" from group one is transferred, and that "item" is neither the last in its group, nor the last in the current string in that group, and the corresponding string in group two has not yet been exhausted, ^{o}J6a is entered. Control is immediately returned to the main routine to compare the next "item" in group one with the current "item" in group two. When an "item" from group two is transferred under analagous circumstances, ^{o}K6a returns control to compare the next "item" from group two with the current "item" from group one. However, when one group or string has already been exhausted and an "item" is transferred from the remaining group or string, ^{o}J6b or ^{o}K6b is entered. Connector J6b returns control to the routine which will transfer the next group

one item to the output string. Connector K6b returns control to the routine which will transfer the next group two item to the output location.

When the current input string in group one is exhausted, and the current string in group two has not yet been exhausted, the routine J7a sets ^{o}K7b and ^{o}K6b, in order that the output string may be completed with the items remaining in the current string in group two. Similarly, ^{o}K7a is entered when the string in group two is the first of the current pair to be exhausted. This routine sets ^{o}J7b and ^{o}J6b in order to complete the current output string with the remaining items in the current string in group one. When the group one string is the second to be exhausted, the J7b routine is entered, and ^{o}J6a and ^{o}J7a are reset for the beginning of a new string. When the group two string is the second to be exhausted, the K7b routine is entered, and ^{o}K6a and ^{o}K6b are reset for the beginning of a new string.

Finally, the J8a routine is entered when the item last transferred is the last in group one, and group one is the first group to be exhausted. Connectors K8b and K6b are set, in order to complete the current output string with the items remaining in group two. The J8b routine is entered when the item transferred is the last in its group, and that group is the second to be exhausted. Connectors J8a and J8b are reset and control is transferred to the routines which establish the controls for the next cycle of operation. The routines K8a and K8b are entered when group two ends under similar circumstances.

In preparation for the second cycle, the output function table Ly is transferred to the standard working location for the input function table L. The number of items to be transferred from each input string to the output string is doubled by substituting 2R for R. A test is made to determine whether the new R is equal to 4. Because R can only equal two at this time, the controls for the second cycle are set. Limits J and K are set to L_3 and L_7 respectively, k_o is set equal to L_4, and j_o is set equal to its original value, L_o. Control is then transferred back to the routines which reset a and b equal to zero and initiate the comparison routines for cycle two.

In the second cycle, only four strings of two items each are collated; one string of two items is entirely disregarded. At the conclusion of the processing, Ly is transferred to L, and R is replaced by 2R. At this time 2R is equal to four, and the routines which establish controls for the third cycle are entered. Because the same eight items are collated in the third cycle, j_o and k_o are reset to L_o and L_4 respectively. An alternate end-of-cycle routine is then designated before control is transferred to begin comparisons again.

The third cycle proceeds in the same manner as cycles one and two. There are four items in each input string, and eight in the output string. When the eight items are in a monotonic sequence, the third cycle is complete, and preparations are made for the fourth cycle. At this time, the string of two items, which has been disregarded for two cycles must be collated with the complete string of eight items. To do this, Ly is transferred to L,

2R replaces R, J is set to equal L_7, and k_o is set to equal L_8. As before, j_o is reset to L_o. Because this will be the last cycle of collation for this block, an alternate routine is provided in order that the correct concluding routines may follow when collation has been completed.

The distinguishing characteristic of the fourth cycle is the appearance of the partial string in group two. The collation process is not disturbed by the incomplete string, because tests will show the group to be exhausted at the conclusion of the transfer of the second item. Group one items are transferred, according to the correct manner of sequencing, until this group is also exhausted.

At the conclusion of the fourth cycle, the output function table Ly contains the addresses of the ten keywords, in order according to their contents. Data transfer instructions are fabricated to transfer the original items of data to the output block in the sequence thus indicated. These data transfer instructions are fabricated in the same way that comparison instructions were fabricated earlier. When all the items have been transferred to the output location, routine P1 is entered in order to write the collated block on the first of the three output tapes. Routine P2 is then designated for the next block. Finally, the controls for a new first cycle are established and the initial routine is reentered.

When the block of data just collated was first placed in the data storage locations, a second was left in register I. This

block is now placed in the data storage location, and another block is read into register I for use when the current one has been processed. The four collation cycles are then repeated with the new data. After the collated block has been written on the tape specified by P2, routine P3 is substituted in the program.

Processing continues in this mannen After each block has been collated, the print routines are cyclically rotated, in order to divide the data into three approximately equal groups for External Collation.

When a sentinel in the last position of a data block indicates that the end of the tape is at hand, the first location in the block is tested for sentinel. If A_0 contains sentinel, all the data has been processed, and the SJ routine is entered. If sentinel is not found in A_0, successive items in the block are tested. When the last data item has been located, ignore symbols are inserted for the missing items, and the program is adjusted to enter the SJ routine immediately following collation. The block is collated in the normal fashion.

When collated, the ignore symbols, which have the lowest pulse combination, are regarded as the items of least magnitude, and appear at the end of the descending sequence in the block. It is necessary to insert these symbols, because the collation programs cannot provide for the processing of a partial block of data.

When the last block of data has been collated, the routines which follow make the necessary preparations for External Colla-

tion. The counters u,v,w,r,s,t, and H are all set equal to zero. The constant p, an integral power of three, is set to its initial value of one. The counter R is set to equal one, 3R to equal three and the original data tape is rewound. The number of External cycles required to complete collation of the data is automatically determined from the counter f, which recorded the number of blocks collated during Internal Collation. To accomplish this, H is increased, and p is replaced by 3p. A test is made to determine whether p is greater than f-1. If so, H indicates the number of External cycles required. If p is not greater than f-1, H is increased, 3p again replaces p, and the test is made once more. The routine is reiterated until p is greater than f-1.

The next operation determines whether H is odd or even. If tests prove it to be odd, the counter U is set to zero to indicate this fact. One block of instructions for the ascending series of three-way External Collation is then placed in memory locations 000 through 059, and control is transferred to location 000 to begin the new program. If tests prove H to be even, U is set to equal one. The instruction tape is then advanced until the special descending series instructions are available.

A counter is established to count the number of blocks passed. Each time a block is read, the counter is increased and compared with its designated limit. When counter and limit are equal, the special series instructions have been positioned. Control is then transferred to location 000 to begin the new cycle.

5.0.5. Status of the Data at the Conclusion of Internal Collation.

At the conclusion of Internal Collation, the entire quantity of data has been divided into approximately equal groups of one-block strings. Each group has been placed on a separate tape. The items within each block are in descending monotonic sequence, but the contents of the various blocks are random with respect to each other. If there are two groups, and the data is not evenly divisible by two, the first group is longer and contains only one extra block. If the data has been divided into three groups, the first group may contain an extra block, or both first and second groups may contain an extra block.

The counters necessary for the control of input and output strings during the first External cycle have been initially set and stored in memory locations which will not be disturbed by subsequent instructions. The number of cycles required to complete the collation of the data has been determined. If an odd number of cycles is required, the first block of ascending series instructions is in the memory and the second block is in register I. If the number of cycles required is even, one block of the special cycle instructions has been positioned in the memory, and the second block is in register I. The last instruction of the Internal Collation program transfers control to memory location 000, and initiates the sequence of instructions which are discussed in Chapter 6.

CHAPTER 6
EXTERNAL COLLATION

6.0.1. Introduction.

The initial input data for the External Collation programs is the output of the Internal Collation program which precedes them. If the External programs are to employ the three-way method, the entire quantity of data consists of three approximately equal groups, each on a separate input tape. If the External programs are to employ the two-way method, the data consists of two approximately equal groups, each on a separate input tape. If one group is larger than the other(s), it must be the first group, and it contains only one extra block of data. If three-way collation is to be used, both first and second input groups may contain an extra block of data.

The data is in order within each block, but the various blocks are random with respect to each other. As the tapes have been prepared for use with backward read instructions, the first block from each group to be read into the memory will be the last placed on that tape during the Internal Collation process. In turn, the last block read into the memory from each group will be the first from that group to have been internally collated. When subsequent read instructions in the External program are carried out, sentinel blocks present on each tape will indicate that the data has been exhausted.

The number of sentinel blocks on the input tapes varies with the collation method to be used. If three-way collation is to be employed, tape interlock cannot be completely elimin-

ated, and a single auxiliary storage location for each group will be required. In this case two sentinel blocks per tape are present to insure that the input mechanism operates correctly when continuous-read instructions are given. A third sentinel block on the second input tape provides for the possibility that there may be only one block of data to be collated. (See section 6.0.2)

If the two-way method of collation is to be employed, each input tape contains three blocks of sentinel indication. The associated External program then will use two auxiliary storage locations for each input group, and the customary continuous-read instructions. No additional sentinel indication is required in the event that there is only one block of data present.

The counters and constants required for the correct operation of the External processing are stored within the memory in locations which are segregated from those used for data or instructions during the Internal or External Collation programs. These include: H, R, r, s, t, u, v, w, and the tape label (identification number), for three-way collation. For two-way collation, the list is similar, but r and w are not included. The counters u,v, and w, will be used to count the units transferred from an input string; the counters r, s, and t, will be used to count the units transferred to an output string. All have an initial value of zero, established in the Internal program. As these are the counters which will adjust for the occurrence of incomplete strings, it is necessary to set them prior to the beginning of the cycle in which they are used. The constant R, the number of units to be collated from each input string, for

the current cycle, is also preset. The constant H is the number of cycles required to complete collation. It is determined in the closing routines of the Internal program, and must continue to be available throughout External Collation. It is the constant which controls the number of iterations of the External cycles.

The method of collation to be used depends upon various considerations. If there are no more than four or five UNISERVOs available, only the two-way method can be used. A maximum of eleven External cycles will be required to collate a complete tape under these circumstances. If collation is to proceed by the three-way mathod used in several of the programs of this report, the number of cycles required to collate a complete tape will be reduced to seven, but the UNISERVO requirements will be increased to eight. By special programming, the same method might be used with seven servos. If there are ten servos available, collation may be completed in five cycles, using the four-way method. The decision as to the method to be used will have been made by the programmer well in advance of Internal processing.

6.0.2. The Method of External Collation.

When the three-way method of External Collation is to be used, and an odd number of cycles are required to complete collation, the closing instructions of the Internal program position the first block of the ascending sequence instructions in memory locations 000 through 059, and transfer control to 000.

The instructions which follow position the remaining instructions, and test to determine whether the current cycle is the last, and whether the special descending program has been used.

If the special program has been used, and the ascending sequence instructions are to be used for an "even" cycle, rather than an "odd" one, adjustments in the designation of input and output tapes are made. If the current ascending cycle is the last, the final output tape is rewound to the leader, and adjustments are made which will conclude External Collation at the end of this cycle. If the current cycle is not the last, the three output tapes are prepared for use with backward read instructions by writing two sentinel blocks on each. The first blocks of data to be collated are then positioned in the memory.

Two storage locations are assigned to the data from each input tape. Group one is stored in blocks A and D, group two is stored in blocks B and E, and group three is stored in blocks C and F. The data destined for location F is left in register I at this time. A test is then made to determine whether there is a sentinel indication in the data block C. If there is no data in C, the program is adjusted to collate the data in groups one and two only. This situation cannot arise in any cycle other than the last. It will arise in the first ascending External cycle, if the total quantity of data is only two blocks.

In an occasional instance, it may be possible for the tape which is to be Externally collated to contain only one block. This block is on the first of the three input tapes, and the other two then contain nothing but end-of-tape sentinels. No routine in

the Internal or External programs has been provided to care for this situation.

The manner in which the program proceeds from this point, causes the true data in input block A to be collated with the contents of the sentinel block in storage location B. The contents of the sentinel block, except for the first location, may be anything at all, and therefore, the resulting sequence is far from correct. If this situation can exist, either of several adjustments in programming must be made to take care of it.

A test may be added to the closing routine of the Internal Collation program, to determine whether more than one block has been processed. If there is only one block of output data, the routine may then reverse the sequence from descending to ascending order without using the External Collation instructions. On the other hand, the External program may be modified to determine the absence of data on tape B, just as the absence of data on tape C is now determined and to adjust the routines accordingly. The third alternative adjusts the contents of the extra sentinel block which is placed on the second output tape in Internal Collation. When this tape becomes an input tape for External Collation, the special sentinel block is the first block to be placed in the memory location B. If the last item in the block is a complete sentinel item having a higher pulse combination than any item of true data, collation may proceed without error. A sentinel block will be transferred to the output string as if it were a block of data, but this is of no importance. Furthermore, as the second input tape contains three sentinel blocks instead of two, no complica-

tions arise from the use of continuous-read instructions. Of the three alternatives, the first modification of the programming would probably prove to be the most efficient, and there is adequate memory space in the Internal program to store the additional instructions.

Throughout the main routine of the three-way ascending cycle, the items in the current string from input group one are compared with the items in the current string from the other two input groups. Comparisons begin with the item of least magnitude in each string, i.e. the last item in the block which is first positioned in the memory. The item which is the least of the three is transferred to the current output location. After the data has been transferred, tests are made to determine whether the end of the current output and input blocks has been reached. When the end of an input block has been reached, the data in register I is transferred to the designated storage location, another block from the current group is read into register I, and the appropriate auxiliary block is transferred to the working location just exhausted. This sequence is not halted by tape interlock unless the previous read instruction is still in operation. Approximately two hundred computational instructions can be executed in the period required to read one block from an input tape and place the data in register I. If two read instructions are given within a shorter period, the second is interlocked until the first is completed.

The new block of data is next tested for the presence of sentinel. If sentinel is not present, this is a full block of

data, and the group has not yet been exhausted. End-of-group and end-of-tape coincide in this program. The next test determines whether the current string has been exhausted, and the new block is part of a new string. If the current block is not the first in a new input string, processing continues with the item in the last location as before. If the current block is part of a new input string, the program is adjusted to complete the current output string with the items remaining in the strings of the other two groups.

By testing for end-of-group before testing for end-of-string, one test per group is eliminated, as the two limits coincide when strings are complete. More important, if the test for end-of-string were not accompanied by a test for end-of-group, an incomplete last string would never be discovered. If the order of the tests were to be reversed, the sequence of events to follow could not be arranged so conveniently.

When all of the items in the three input strings of R blocks each have been collated in this fashion, the resulting output string consists of 3R blocks of items in ascending sequence. As each output block is completed, it is written on the current output tape. When the output string is complete, an alternate output tape is designated, and another string is collated. At the conclusion of the second output string, the output tape is again alternated, and a third one is designated. The third output string is written on this tape. The fourth string is written on the first output tape again. The output tapes are continuously alternated in this fashion.

When the last complete string has been written on an output tape, this fact is indicated by the ending of at least one of the three input tapes. If all three tapes end, there is no extra data on any input tape, and therefore, no incomplete output string is formed. If only one tape ends, it must be the third, and additional data is then present on the first and second input tapes. The routines which follow collate the data from these two strings in the normal two-way fashion. The resulting incomplete output string is then written on the current output tape, and the closing routines of the program are entered. If two input tapes end when the last complete output string is collated, there is an additional input string in only the first group. The program is then adjusted to allow this data to be transferred to the current output tape, processing it only sufficiently to reverse the order from descending to ascending sequence.

If the current cycle is the last, there is only a single output tape. The concluding routines place the necessary sentinel blocks at the end of this tape, rewind all input and output tapes, print information on the Supervisory Control printer in order to identify the output tape and label it correctly, and then stop the program. If the current cycle is not the last, the closing routines determine the values of the counters which count the blocks in the output strings. If all have been reset to zero, no incomplete output string has resulted from the collation of this cycle, and the counters r, s, t, u, v, and w are all equal to zero for the next cycle. If either r, s, or t has a value other than zero

at the conclusion of the current cycle, an incomplete input string will be collated in the following cycle. To adjust the program to process this string as though complete, the corresponding input string block counter and the first output string block counter for the next cycle are preset to the number of missing blocks. The quantities R and 3R are each tripled, and the first block of the descending series instructions are positioned in the memory. The second block of instructions is placed in register I. Control is then transferred to the first location of the new instructions.

The processing of three-way collation cycles in descending sequence is very similar to the processing of the cycles in ascending sequence, but the order of the final data is reversed. It is assumed that the desired final order of the data will never be in descending sequence, therefore no descending cycle can ever be the last. This simplifies the initial routines of the descending program as no adjustments need be made to conclude the entire External process with the current cycle. Adjustments in the designation of input and output tapes are again required if the special descending series program has been used.

In the descending series routines, the item of greatest magnitude in group one is compared with the item of greatest magnitude in each of the other two groups. The item of greatest magnitude is then transferred to the output block. This initiates the first string in descending order. Other strings are processed similarly. All the techniques employed in the ascending series are also employed in the descending programs. The two are therefore almost completely parallel.

At the conclusion of the descending series, one block of another group of ascending series instructions is positioned in the memory, and a second is placed in register I. When control is transferred to the starting location of the new instructions, the next ascending cycle is begun. The cycles continue to alternate in this fashion. Because the instructions for each cycle are so lengthy, only those for one series can be placed in the memory at a time. Therefore, although the instructions for all odd cycles are the same, and those for all even cycles are the same, it is necessary to read new instructions into the memory for each new cycle.

The instruction tape contains the Internal Collation instructions first. The next seven instruction groups alternate four sets of ascending series instructions with three sets of descending series instructions. The instructions for the special cycle follow last, as it is expected that these will be used infrequently.

When two-way External Collation is employed, the operation of the cycles closely parallels that of three-way External Collation. Several important differences do exist, however. The first of these lies in the total number of cycles required to complete collation. Eleven cycles are required by the two-way method, in contrast to seven cycles required by the three-way method. Secondly, the reduction in the number of instructions needed for each two-way cycle makes possible the storage of both ascending and descending series instructions in the memory. The two-way instruction tape therefore consists of only three groups of instruc-

tions. Those for Internal Collation are followed by the combined ascending-descending External Collation instructions and the special cycle instructions are the final group.

When two-way collation is employed, tape interlock may be completely eliminated. Two auxiliary storage blocks for each of the two input groups are maintained. As the data in each of the current working locations is exhausted, it is replaced with that from one of the associated storage locations. The two storage locations assigned to each data block are used alternately for this purpose. A new block of input data is read only when an output block is written on the output tape. To do this, the address of the current block in register I and the source of the next block to be read, must be known. The address of the latter must also be available when needed.

In general, the sequence of events is as follows. When an output block ends, the data in register I is positioned according to the address in Z, and a new block is read into register I from T_K. The address of the new block in I, currently stored in a location designated as Z + 1, is then transferred to Z. When an input block is next exhausted, the contents of one of the associated auxiliary storage blocks is transferred to the working location. To refill the emptied storage block the associated tape is designated as T_K and the storage location of the empty auxiliary block becomes the next Z + 1. When another output block is completed, the block in register I is positioned according to the address in Z, and the new read instruction is executed.

To use this technique with continuous-read instructions, each input tape must contain three sentinel blocks. In the event that there is only a single block of data, no additional sentinel indication is required. However, if this situation may exist, some program adjustments are required. These are similar to the adjustments required when the same situation arises in three-way collation.

When tests indicate that there is an extra string of data on the first input tape during two-way External Collation, the program is adjusted to reverse the order of these items, and to place them on the current output tape. The output string block counters for the following cycle are then adjusted in the same manner as the corresponding ones are adjusted in three-way External Collation. If there is no extra data on the first tape, the final output string is complete, and there is no incomplete string to be processed in the next cycle. When all the data has been placed on a single output tape, the two-way program is concluded.

6.0.3. Detailed Description of External Collation Programs.

Program 952-1 (Section 6.1.1). Two-Way External Collation with Tape Interlock Eliminated; - Ascending and Descending Series.

When the two-way ascending External Collation program follows immediately after the Internal Collation program, the following facts are known and may be asserted. The counters x and y, which count the items transferred from input blocks A and B respectively, are both equal to zero. The counter m, which counts the items transferred to the output block G, is also equal to zero. The

counters s and t, which count blocks transferred to an output string, and the counters u and v, which count blocks transferred from an input string, are equal to zero. The constant R, the limit of counters u and v, is equal to one at this time, and 2R is therefore equal to two. The input data is on tapes three and four, and the output data will be placed on tapes five and six. The first output string will be placed on tape five, as determined by the setting of connector P1. The address of the block which will be left in register I by the first tape-read instructions is the storage block E. The first input tape to be read following the initial positioning of the data will be tape four, and it is therefore the designated T_K. The address of the block to be read from T_K, (the first Z-1), is the storage block F.

In addition, the following paths of operation are set. Connectors L1a and L2a insure that the data stored in auxiliary locations is processed in the correct order; the contents of storage block C will be transferred to the A working location when A has been exhausted, and the contents of the storage block D will be transferred to the B working location when B has been exhausted. The connector Q1 establishes the initial comparison between the items of the two groups for the ascending series, and the connectors A1a, A2a, B1a, and B2a return control to this comparison routine following the transfer of an item to the output location. The connectors RA0, RB0, SA0, and SB0 provide for the correct adjustment of the program when a string or group first ends. Connector 7a insures the proper alternation of the output

routines. Connector 14a provides for the alternation of ascending and descending cycles when the special descending series has not been used. Finally, connector 15a is set to make adjustments in counters s and t for the next External cycle.

It is possible to assert all these conditions on the flow chart of program 952-1, although many apply only to the first External cycle, and then only if that cycle is the normal ascending cycle. When the ascending series routines are entered from the special descending cycle, and when they are entered following the descending series routines, the point of entry is connector 1. The assertions associated with the initial routines do not apply under these circumstances.

When the remainder of the necessary constants and instructions have been positioned in the memory, a test is made to determine whether the current cycle is the last (°1). If the current cycle is not the last, three sentinel blocks are written on each of the output tapes, and five blocks of data are positioned within the computer. Two of these are read from tape A and are stored in data locations A and C. Two blocks are read from data tape B and stored in locations B and D. The last block is another block from tape A, which is left in register I. Control is then transferred to the comparison routine.

If the current cycle is the last, a single final output tape is prepared. The tape designated in the currently set P routine is rewound to the leader, and the tape identification block is written on it. Connector P5 is then set to eliminate the end-of-output-string tests, and connector 15b is set in order to con-

clude the program when the last of the data has been collated. Control is then returned to the routine which initially positions the input data.

The comparison routine for the ascending series program is Q1. The key digits of the last item in block B are compared with the key digits of the last item in block A. Three comparisons are required to accomplish this, as the data in this program has a split key.

Following the comparison(s), the item of lesser magnitude is transferred to the output location. Tests are then made to determine whether the end of the output and input blocks has been reached. If the end of the output block has been reached, the current P routine is entered. If the end of the output block has not been reached, the counter m is increased, and the test for the end of the input block is made.

If the item just transferred was an item from the A block, counter x is compared against its limit, N. If x and N are equal, the input block has been exhausted, and the L1 routine is entered, in order to transfer the contents of an auxiliary block to the A working location. If x does not equal N, the input block is not exhausted, and the comparison routine is reentered in order to compare the next items.

If the item just transferred was an item from the current B block, the counter y is compared with N. When y and N are equal, the L2 routine is entered for the purpose of positioning another block of group two data. If y and N are not equal, the comparison routine is entered as before.

The "main routine" is repeated in this fashion until an output block is completed. In the P routine which follows, the contents of location G are written on the output tape, and a test is made to determine whether the end of the output string has been reached. If the string is not yet complete, the counter for that string is increased, and control is transferred to the routine which reads another block of data. To eliminate tape interlock, it is desirable to read a block only when a block has been written, although the decision as to the tape from which it will be read and the address of the new block of data, is made in another routine.

The block in register I is placed in location C, D, E, or F, depending upon the address stored in Z, and the new block of data is read from T_K. The address previously stored in location Z + 1 is transferred to Z, for use when the next read instruction is given and the block now stored in register I must be positioned. The counter m is then reset to zero, and the end-of-input-block routine is entered as usual.

When all of the data in input block A has been exhausted, it is replaced by the data in one of the auxiliary locations. These are alternated, so that the sequence of the input blocks is not disturbed. Because the first auxiliary location is the C block, it is the data from this block which is transferred when A is first exhausted. In addition, the program is adjusted to read next from T_A, to place the new block in location C, and to replace the A block when next exhausted with the data now in E.

The new data in A is then tested for sentinel indication. If there is sentinel in A, group one is exhausted, and additional adjustments in the program are required (°SA). If the current A block is not a sentinel block, the counter x is reset to zero, and a test is made to determine whether the current input string has been exhausted. If u and R are equal, the data in A is the first block of a new string, and program adjustments are made in the RA routine which will complete the output string with the items remaining in the other input string. If u and R are not equal, u is increased and control is transferred to the comparison routine as before.

When the data transferred to the A location was that stored in location E, the program is adjusted to read next from T_A just as before, but the address of that block will be E, and the next block to be transferred to the A location will be the one now stored in C. The tests for end-of-group and end-of-string are made as described above.

When input block B has been exhausted, the procedure is very similar. If connector L2a is set, the data in B is replaced with the data in D. Arrangements are then made to transfer from block F when B is next exhausted, and to read from tape B when another read instruction is executed. The address of the data in register I at that time will be the location just emptied, D. The new B block is then tested to determine whether the group has been exhausted. If it has, the SB routine is entered. If group B is not exhausted, the counter v is tested to determine whether the current input string has been exhausted. If the new block

of data is part of a new string, the routine RB is entered. If the current input string has not been exhausted, v is increased, and control is transferred to the comparison routine as before. The sequence of operations in routine L2b is completely analogous.

Eventually, an output string will prove to be complete. The associated counter, either s or t, is reset to zero, and the alternate P routine is designated for the next output string. Routines P1 and P2 alternate at the conclusion of each output string. This insures that the output data will be divided into two approximately equal groups of strings for the next cycle. If °P5 is set, no test is made, as only one output tape is required for this cycle. Following the alternation of the P routines, the routine which reads a new input block is entered, just as after the writing of any output block.

When the first input string is exhausted, routine RA0 or routine RB0 is entered, depending upon the group concerned. Adjustments are made in the program, in order to complete the current output string with the items remaining in the other input string without returning to the comparison routine. To do this, connectors A1b and A2b or B1b and B2b are set, depending upon the string which remains to be transferred.

When an output string is completed, the remaining input string must also be exhausted. Either routine RA1 or RB1 will be entered, depending upon the connector which had been set. These routines readjust the program to compare the items which will begin the next new string.

Similarly, when the first input tape ends, either routine

SAO or SBO will be entered. Each one makes adjustments in the program to permit the output string to be completed with the items remaining in the input string of the other group. If it is the A tape which ends first, it is known that the B tape must certainly end when the current string in that group ends, as there can be extra data on tape A only. If it is the B tape which ends first, the end of the current string in the A group may also end the group or it may not. For this reason, the adjustments made in the routine SBO, must provide for the end of another string in A as well as for the end of the A tape.

When the end of tape B is followed by the end of a string in A, the remaining group one data is transferred to the new output string. The only processing required for this data is the usual reversal of sequence.

When the end of the second input tape indicates that all of the data has been collated, the concluding routines of the cycle are entered. If the current cycle is the last, it is necessary to write three sentinel blocks on the output tape, to rewind all tapes and then to stop. If additional cycles are required to complete the processing, counters s and t are examined to determine whether an incomplete string was formed. The status of the counters s, t, u, and v for the next cycle is then determined, R and 2R are doubled, and routine 14a is entered in order to make adjustments for the descending cycle.

To prepare for the descending series program, the counters x, y, and m are reset to zero. Connectors A1a, A2a, B1a, B2a, L1a, L2a, RAO, RBO, SAO, and SBO are reestablished and T_K, Z and Z + 1

are set to their previous initial values. The input tapes for the descending series are tapes five and six and the output tapes are tapes three and four. The alternating print routines are P3 and P4, and connector P3 is set. Connector 7c is set to insure the alternation of P3 and P4, and connector 14b is set to insure the return to the ascending series instructions when the next cycle is complete. Connector Q2 is set for the comparison routine which will permit the transfer of the item of greater magnitude. Control is then transferred to the routine which tests to determine whether the current cycle is the last.

The operations of the descending cycle are similar to those of the ascending cycle, except that the final sequence of the data is in descending order, and the designation of input and output tapes is reversed. Whereas °7a and °7b previously alternated P1 and P2, °7c and °7d now alternate P3 and P4.

The program for the special descending cycle is used only when an even number of cycles are required to complete collation. The procedure employs all of the same techniques already described for the ascending cycle. At completion of the special cycle, the instructions for the ascending and descending cycles are placed in memory, and adjustments are made to reverse the designation of input and output tapes. This is accomplished through the use of connectors 7c, 14c, and P3, and by changing the initial designation of T_A, T_B, T_P, and T_K. No other changes are necessary.

Program 953-1 (Section 6.1.5).
Three-Way External Collation
Ascending Series

The conditions which exist at the outset are asserted as follows: The counters x, y, and z, which count the number of items collated from the input blocks A, B, and C, respectively, have been set to zero. The counter m, which counts the number of items in the output block G is also equal to zero. The counters s and t, which count blocks transferred to a string on output tapes Y and Z, respectively, are equal to zero, but the corresponding counter for tape X cannot be asserted at this time. The counters u, v, and w, which count blocks transferred from a string on input tapes A, B, and C, respectively, are also not asserted, because any one may have a non-zero value in the event that there is an incomplete string in one of the input groups.

The variable connectors, RAO, RBO, and RCO are set, in order to correctly adjust the program when one of the input strings ends. The connectors SAO, SBO, and SCO are also set, in order to correctly adjust the program when one of the input groups (tapes) ends. The connectors Q1, A1a, A2a, B1a, B2a, C1a, and C2a are all set, in order to establish and maintain the three-way comparison of items from each of the input groups. Assuming that this is not the last cycle, connectors 2a and 3a provide for the necessary sentinel blocks on the three output tapes. Under the same circumstances, connector 9a provides for the tests for an incomplete string at the end of the cycle. Connector P1 is set to allow the first output string to be written on tape X.

Depending upon the processing which immediately precedes the current ascending cycle, additional facts concerning the counters and constants may be known. When the ascending series follows Internal Collation, R is equal to one, 3R is equal to three, and r, s, t, u, v, w, are equal to zero. If the ascending program follows the special series program, the only additional information available is that R is equal to three, and 3R is equal to nine. If the ascending series is entered from the descending series program, nothing more is certainly known.

The initial instructions of the ascending series program position the remainder of the necessary instructions and constants. A test is then made to determine whether the current cycle is the last one. If it is not, a test is made to determine the correct output servos for the collated data. If the current cycle is the last, alternate connectors 2b and 3b are set before making the same test.

When the counter U is shown to be equal to one, the special descending series program has been used. The output tapes for any ascending cycle prior to the last are then on UNISERVOs three, four and five, and the input tapes are on servos six, seven and eight. On the other hand, if the counter U is equal to zero, tapes three, four and five are input tapes, and tapes six, seven and eight are output tapes. In either case, the routine which follows positions the input data in memory, and writes two sentinels on each of the output tapes.

When the current cycle is the last, only the first of the three output tapes is required. The identification block is

written on this final tape, after it has been rewound to the leader. An alternate connector 9 is then substituted, to provide for the conclusion of the program at the conclusion of the cycle.

Comparisons are then made between the items of the three input groups. The key digits of the last item in block B are first compared with the key digits of the last item in block A. When the B item is equal to, or less than, the A item, a comparison is made between the B item and the key digits of the last item in the C block. If the A item is less than the B item, a comparison is made between the C and A items. The item of least magnitude is transferred to the first location in the output block.

Following each transfer, a test is made to determine whether the output block has been completely filled. If the block is not complete, the item counter, m, is increased, and control is transferred to connector 8 to determine whether the last item in the input block has been processed. If the output block is complete, the output data is written on the designated tape, T_X.

A test is then made to determine whether the current string on this tape is complete. If the string is not complete, the associated block counter, r, is increased, and control is transferred to connector 8 as before. If the string is complete, r is reset to zero and the program is adjusted to place the next output string on T_Y. Control is then transferred to the end-of-input-block routine.

If the item just transferred to the output block was an item from the A block, the routine in connector 8 tests to determine

whether the A block has been exhausted. If the item counter for this input block $(x)_x$ has not reached its maximum, it is increased, and control is returned to the comparison routine again. If the end of the A block has been reached, x is reset to zero, and the L routine is entered in order to read a new data block from tape A into register I.

As one block of data from tape C was left in register I in the initial routines, it must be placed in memory location F before another read instruction can be executed. To this end, connector Lc is already set. If the A block has just been exhausted, connector Llc is the routine which is entered.

A new block of data is then read from tape A into register I, and connector La is set for the next routine. This insures that the new block from tape A will be placed in auxiliary location D when the next read instruction is executed. The data currently in D is now transferred to the A working location, for processing. If the new block contains only a sentinel indicator, tape A, and hence group one, is exhausted. The current SA routine is then entered. If the new block does not contain a sentinel indication, group one is not exhausted, and a test is made to determine whether the current string in group one has been exhausted. If the counter u,which is used for this purpose,has reached its limit, the current A string is exhausted, and the current RA routine is entered. When the counter u has not reached its limit, the current group one string is not exhausted, and the counter is increased before control is returned to the comparison routine.

If the item just transferred to the output block was an item from the B block, the procedure is essentially the same, except that the B block is tested. The counter associated with items transferred from B is y. When the B block is exhausted, the L2 routine is entered. This routine reads another block from tape B, sets connector Lb, and transfers the contents of the current E block to the B working location. If the B block does not contain sentinel, the counter v is tested to determine whether the current B string is exhausted. If it is not exhausted, v is increased, and the comparison routine is entered again.

If the item just transferred to the output location was a C item, the C block is tested to determine whether it has been exhausted. The counter associated with the C block is z. When the C block is exhausted, the L3 routine is entered. Another block is read from tape C, connector Lc is set, and the contents of the working location F are transferred to the C location. The same end-of-group and end-of-string tests are made on the new block of data in C, and if the string has not been exhausted, the counter w is increased. Control is then transferred to the comparison routine just as before.

The succeeding comparison is made between the next item in the block from which the previous transfer was made, and the other two items used in the former comparison. Again, the item of least magnitude is transferred to the output location, and the tests for end-of-input-block and end-of-output-block are made. The sequence of events remains the same as long as there are items to be compared from each of the input strings.

When one of the input strings is exhausted, an end-of-string routine is entered. This is either RAO, RBO, or RCO, depending upon the group to which the string belonged. The instructions which follow adjust the program to continue the processing with a normal two-way comparison between the items of the two remaining groups. This is primarily a matter of adjusting the Q routine to Q2, Q3 or Q4. Because it has been possible to make these adjustments within the framework of the original coding of Q1, a kind of shorthand has been used to designate the necessary changes. The original lines of coding are specified as Ao, Bo, Co. When one of the groups is eliminated from future comparisons, this fact is indicated by A=0, B=0, or C=0. The associated changes in the line of coding which uses the B location are called B=B1 for Q2, and B=B2 for Q3.

Eventually, a second input string ends. Depending upon the identity of the two groups, and the order in which they have been exhausted, the second end-of-string routine may be any of the following: RA1, RA2, RB1, RB2, RC1, or RC2. Instructions set the connector for the next end-of-string routine, and adjust the program to complete the output string with the remaining items in the third input string. One of the following pairs of connectors is set for this purpose: A1b, A2b, or B1b, B2b, or C1b, C2b. Wherever the outcome of a test previously transferred control to connector Q, it now transfers control to connector 5, or 6, or 7.

Before tests indicate that the third input string has been exhausted, an output string is completed. The associated output

string block counter, either r, s, or t, is then reset to zero, and another output tape is designated for the next string. The three output tapes are cyclically rotated after an output string is complete, in order to divide the data into three approximately equal groups for the next cycle.

Following the ending of the output block, and the completion of the output string, tests indicate that the third input group is exhausted. Depending upon the identity of the group, the routine entered will be either RA3, RB3, or RC3. The instructions which follow readjust the program to compare items from three input strings once more, and then transfer to the three-way Q routine.

The chart which begins on page 6.1.4.1 of Chapter 6 lists all the possible combinations of string endings which have been briefly suggested above. In addition, there are analogous combinations of group endings (the SA routines), and other combinations which arise from an extra string of data on one or more of the input tapes. The source of each sequence of endings, and the program adjustments which must accompany them are carefully delineated in section 6.1.4. and therefore, will not be discussed further here. The necessity for providing for all possible combinations of events makes the coding of three-way collation very extensive. The processing is comparatively short in any particular instance, however, as only one sequence of events can lead to the completion of a given output string.

When all three input groups have ended, the final end-of-group routine is either SA7, SB7, or SC7. In any case, the con-

cluding routines of the program follow immediately afterward. If this cycle is the last, routine 9b or 9c is entered in order to rewind the final and all intermediate tapes, and to indicate to the person in charge (via a Supervisory Control output instruction) the final output servo, and the label for the completed tape. If the current cycle is not the last, the routine which follows tests the counters r, s, and t to determine whether an incomplete output string has been formed. If all three counters have been reset to zero, counters u, v, and w may also be set to zero for the next cycle. If either r, s, or t do not equal zero, the associated input counter, and the first output counter for the next cycle are set equal to the number of missing blocks. All other input and output counters may then be set to zero for the next cycle.

The only remaining operations in the ascending cycle concern the setting of R and 3R for the next program, and the positioning of the first block of descending series instructions. A second block of the new instructions is left in register I.

As previously indicated, the descending series and special descending series programs are very similar to the ascending series, except in the sequence of the output data. Both are simpler than the ascending series program, because neither can be the last cycle of operation. The special cycle has several unique features, but, aside from the different order in which items are considered, these variations arise from the fact that the special cycle if used at all, is always the first External cycle, and therefore, certain additional facts are known at the outset.

6.0.4. Status of the Data at the Conclusion of External Collation.

At the conclusion of the last External cycle, the data on the output tape is a single continuous sequence of items in ascending order. The first block on the tape is an identification block which contains the tape label. There are three sentinel blocks at the end. Collation is complete, if there is only one tape to be processed.

If there are many data tapes to be collated, each is sequenced in the manner described in this chapter. All the individually collated tapes are then merged to form a single string of tapes which are in sequence with each other. This is accomplished by the merging process which is discussed in Chapter 7.

CHAPTER 7

MERGING

7.0.1. Introduction.

The purpose of this chapter is to explain the procedure called Merging, which is the third and final step in the collation process Internal Collation, the first step, receives data in random order and arranges this data in monotonic sequence in block lengths. A complete description of the various methods available is given in Chapter 5. External Collation, the second step, receives the data which is in block lengths, and arranges it in monotonic sequence in tape lengths. Chapter 6 explains in detail the processes used to accomplish this. Merging, the third step, arranges the data, which is in order by tape lengths, into one continuous monotonic sequence

The Merging process is straight forward. Simple comparisons of the data are made, and since an ascending sequence is desired, the quantity of least magnitude is selected first, the quantity of next largest magnitude second, and so on, until all the data has been examined and arranged. The following is an example of three groups of data to be merged; A, B, and C.

A		B		C	
a_1	1	b_1	0	c_1	2
a_2	4	b_2	2	c_2	3
a_3	6	b_3	3	c_3	4

If a comparison is made between the quantities stored in a_1 and b_1, and a_1 is greater than b_1, a second comparison is made between c_1 and b_1. If c_1 is greater than b_1, b_1 is the quantity of least magnitude and is selected first. Then b_2 is compared with a_1 and c_1 and the process is repeated until all the quantities have been arranged.

Programs have been prepared for both two-way and three-way merging. The method used depends partially on the number of UNISERVOs available, and partially on the number of tapes to be merged. Two-way merging requires the use of six UNISERVOs, and three-way requires eight.

7.0.2. Tape Identification.

As previously discussed, a system of tape identification has been devised such that each tape will contain enough information about itself to distinguish it from any other tape. The purpose of the system is to facilitate problem re-run and filing.

A master chart, which designates the current tape identification for each input and output tape, serves as a control for this system. The operator follows this chart during the merging process when mounting each input tape. At the end of the problem the output tapes are filed sequentially as indicated on the chart. The tape identification system has been set up to handle a maximum of 99 tapes.

The master chart shown in Figure 1 of section 7.0.7 indicates one method of labelling a group of 54 tapes. Each tape has a label

containing a two-character alphabetic code plus a two-digit numeric code, e.g. CB01. Prior to the first cycle of Merging, the label of any tape lies in the range AA01.....CB01. The tape label is typed into the memory from the Supervisory Control at the beginning of the Internal Collation program. In the last cycle of External Collation this tape label is placed in the second memory location of the first block on the output tape. At the end of the External Collation process, the operator must attach the printed tape label to the tape reel. The identification number, which has been typed into the memory and printed on the Supervisory Control printer at the beginning of Internal Collation, provides the material for the printed label.

As the program now stands, computer or operator errors which occur prior to merging require the re-running of the data tape. In Merging, however, the identification system has been set up so that any tape may be re-run. To make this possible, it is necessary to have certain information easily available. The first block on the tape is the identification block as before, but it now contains the information necessary for re-run.

Figure 2 of section 7.0.7 represents a typical identification block for three-way merging. It contains the three current input tape identification numbers (q_2, q_3, and q_4), the current output tape identification number (q_1), the three final input tape identification numbers for the current strings (q_5, q_6, and q_7), the keyword of the last item on the previous output tape (q_8), the three memory positions containing the next keywords to be compared (q_9, q_{10}, and q_{11}), and the keyword of the first item on the current

output tape (q_{12}). Since merging is done in ascending series, the keyword of the last item on the previous output tape (q_8) will always be less than, or equal to, the keyword of the next item to be compared. Since there may be several items containing the same keyword, it is necessary to know which memory positions were being used at the beginning of the tape. These factors plus the three input tape numbers are the only requirements for re-running a tape. The only tape that cannot be re-run under the present system is one which contains more than one block of equal key digits. The addition of a block counter routine would eliminate this problem.

There is only one major difference between the identification systems used in the two-way and three-way merging programs. In two-way merging, the initial comparison test precedes the writing of the identification block on the output tape. The keyword of the first item on the tape is known and therefore may be used in the re-run routine as the basis for locating the first relevant item on each input tape. Three-way merging takes a different course, and the identification block is written prior to the comparison. For this reason, the last word on the previous output tape is used as the basis for positioning the input tapes. The routine on page 7.1.2.34, which prepares the three-way merging re-run, uses this system.

7.0.3. Strings and Cycles.

Although the definitions of string and cycle differ in Internal Collation, External Collation, and Merging, there is a logical relationship between all three.

The unit of measure for a string in Internal Collation is an item. A string is formed by comparing items from one group with items from another group, and transferring them in a monotonic sequence into a single series. The unit of measure for a string in External Collation is a block. A string is formed by comparing items in a block on one input tape with items in a block on another input tape, and transferring them in a monotonic sequence onto a single tape. The unit of measure for a string in Merging is a tape. A string is formed by comparing items from one series of input tapes with items from another series of input tapes, and transferring them in a monotonic sequence onto a series of output tapes. At the conclusion of the merging process this group of tapes will be in order within itself.

A cycle in Internal Collation is the period in which the number of items in each string is doubled. A cycle in two-way External Collation is the period in which the number of blocks in each string is doubled. A cycle in two-way Merging is the period in which the number of tapes in a string is doubled, and in three-way Merging it is the period in which the number of tapes in a string is tripled. At the beginning of cycle one for example, the strings are each one tape long. That is, each completely collated tape to enter the Merging routine is a string in itself. Figure 3 of section 7.0.7 shows the length of the input strings for consecutive cycles in the merging of 99 tapes.

7.0.4. Tape Interlock.

Tape interlock is the interruption of all computational operations for the period needed to complete the tape operation cur-

rently in progress, and to initiate the one required by the current instruction. This has been discussed in detail in section 4.0.3.

All programs cannot be coded to completely eliminate tape interlock, because the memory space in the computer is limited to 1000 locations. For example, the three-way Merging program uses almost all the available memory space to minimize tape interlock time. In order to completely eliminate interlock, at least six more blocks of memory space are required. However, two-way Merging, which requires less space, has been coded to completely eliminate tape interlock.

7.0.5. Elimination of Block Fill-in.

Because the collation routines are unable to handle incomplete blocks of data, the Internal Collation program includes a routine which fills in the partial block with suitable words for the missing items. A pulse combination of lesser value than any keyword in the data is used if the final arrangement is to be in ascending sequence. A pulse combination of larger value than any keyword in the data is used if the final arrangement is to be in descending sequence. The programs already coded assume an ascending sequence and use an ignore symbol as the fill-in since its pulse combination is of least magnitude. During the collation process, the keywords of the filled in items are grouped at the beginning of the string in ascending collation and at the end of the string in descending collation.

At the end of External Collation, all ignore symbols appear at the beginning of the first data block on each tape. The first

data block on any collated tape may contain a maximum of n-1 items of ignore symbols. If each of three tapes contain n-1 items of ignore symbols, and they are merged together, two complete blocks of ignore symbols will result. In two-way merging, one complete block of ignore symbols may result. The merging programs have been coded to discard all complete blocks of ignores at the beginning of each string throughout all cycles prior to the last (Figure 4). A routine, 962-4, on page 7.1.1.26, has been programmed to show the method of discarding all the remaining ignore items during the final cycle of two-way merging.

7.0.6. Detailed Description of Three-way Merging: Program 963-3 (Section 7.1.2).

The instruction tape for Three-way Merging is on UNISERVO one, and after it has been read into UNIVAC the tape is rewound. This leaves UNISERVO one free and it is subsequently used along with UNISERVO two for the alternating output tapes. The initial input tapes are on UNISERVOs six, seven, and eight, and they alternate with tapes on UNISERVOs three, four, and five respectively. It is general practice to refer to the actual tapes by number rather than to the UNISERVO each tape occupies. For ease of explanation the remainder of this report will follow this procedure.

At the beginning of each Merging cycle, the operator must type two words on the Supervisory Control. The first of these words contains the final identification numbers for the three strings of input tapes to be merged in the current cycle (α_n, β_n, γ_n). The second word contains the initial output tape identification number for the current cycle (δ_1). These quantities are

found on the master chart (Fig. 1) which serves as a guide throughout the merging process. All other identification data is generated within the computer. Each initial tape will have the same code letters as the final tape in its series, therefore, the digits 01 are affixed to these code letters, giving the correct identification numbers (α_{n-n+1}, β_{n-n+1}, and γ_{n-n+1}). When this process has been completed and the transfers to the identification block completed (Fig. 2), all the identification information so far obtained (q_1... q_7) is typed out on the printer associated with the Supervisory Control. This allows the operator to make a visual check with the master chart to be sure that the correct tape identifications were typed into UNIVAC.

After this check is made, the data tapes are read in. The identification numbers which were put on the tape in the second word of the first block during External Collation are checked with the tape identification numbers in q_2, q_3, and q_4. This is done so that any incorrectly labelled or incorrectly mounted reel will be detected. When such an error is found, a routine is entered which rewinds this tape, and prints out on the Supervisory Control printer both the correct and incorrect tape identification numbers, and the UNISERVO number. This enables the operator to relabel the incorrect reel if necessary and mount another tape on the designated UNISERVO.

When the correct data tapes have been read in, the remaining identification locations must be filled. The initial memory locations of x, y, and z are sent to q_9, q_{10}, and q_{11} and the contents of q_8 and q_{12} are cleared to zero. The next process is to write the identification block on output tape two. All identification

blocks are written at the rate of twenty characters per inch. This pulse density is used because a printer can only receive data at this rate. If the label on a tape reel should be lost, the identification block may be printed on a UNIPRINTER and a new label made.

The next step is to read in the data from tapes six, seven, and eight and fill blocks A, B, and C. In order to partially eliminate tape interlock, extra data storage blocks D, E, and F, are kept in the memory. Blocks D and E are filled from tapes six and seven, and the block from tape eight, is held in register I (Fig.5) for future transfer to block F. When one of the data blocks empties the contents of rI are transferred to F, a block from the designated tape is read into rI, the contents of the associated auxiliary block are transferred to the exhausted data block, and computation proceeds. See Figure 6 of section 7.0.7.

The main comparison routine follows the reading in of data, and the items are transferred to the output block in ascending monotonic sequence. After each transfer of data, there follows a test for the end of the input block. If the end has not been reached, the item counter is increased by unity and a test for the end of the output block is made.

This basic sequence is repeated until the output block is filled, at which time, a test for a complete block of ignore symbols is executed. Should there be a block of ignore symbols, this block is not written on the output tape, because all complete blocks of ignores are discarded. As soon as the output block is found to contain a true item, the test for ignore symbols is dropped from the program for the remainder of the cycle, and the output block

is written on tape two.

Each output block is counted, and a test for the maximum allowable blocks on a tape is made. If the limit has not been reached, the block counter is increased by unity. The first word in the first data output block is transferred to position q_{12} in the identification block. This word will be used for part of the external tape label. The routine which executes this operation is dropped temporarily from the program.

When the end of the output tape has been reached, three sentinel blocks are written on the completed tape and a tape label is typed out on the Supervisory Control printer. Since two tapes have been allocated for output data, the alternate output tape is now activated. In addition, the block counter is cleared to zero, and the routine which transfers the first word to q_{12} is again included. The following changes are made in the identification block. The output tape number is increased by unity, the memory locations of the next three quantities to be compared are transferred to q_9, q_{10}, and q_{11}, the keyword of the last item on the previous output tape is sent to q_8, and q_{12} is cleared to zero. This block, which contains enough information to enable a re-run procedure to be effected if necessary, is now written on the current output tape. If all three input strings are still in use, the routine then transfers back to the main comparison sequence.

When an input block has been completely transferred, the following operations occur: the contents of rI are sent to the proper auxiliary block, the associated reserve data is transferred to the exhausted input block, the correct tape is read into rI (Fig. 6),

and the input block item counter is set back to zero. A test for the end of the input tape follows, and if the end has not been reached, the routine transfers control to the test for the end of the output block. If the input tape has ended, it is rewound and the routine which tests for the end of string is entered.

If the end of string has not been reached, the input tape identification number is increased by unity. The identification number of the next tape in the series is checked. This tape has previously been mounted on the alternate UNISERVO allocated to this string. If the wrong tape has been mounted, either through the carelessness of the operator, or because the external label is incorrect, a routine is entered which rewinds this tape, and prints on the Supervisory Control printer both the correct and incorrect tape numbers, and the UNISERVO number. The operator then mounts another tape on the UNISERVO, and the checking process is repeated. When the test is satisfactorily passed, the second block on the tape is read into the designated data block and control is transferred to test for the end of the output block.

When the end of one string has been reached; controls are set to eliminate this string from the comparison sequence, the identification location for the input tape in this string is cleared to zero, and the tape just ended is rewound. In addition, the end of string routines for the remaining two strings are altered. Control is then transferred as before to the test for the end of the output block.

When the second string ends, the tape identification location is cleared to zero as before, and the tape just ended is rewound.

At this time, control may be changed to handle the remaining tapes in the last string. Since there is only one string left, the main comparison routine is no longer needed. The number of items remaining in the current input block equals the number of items necessary to fill the current output block. When the input block empties, and sentinel is detected in the auxiliary block, the last two string have ended in the same output block. The closing routines are entered after the final output block is written out.

However, if two strings end, and sentinel is not detected in the auxiliary block of the third, at least one block still remains to be transferred from the third string. In this case, controls have been set to read in a block, test this block for sentinel and if no sentinel is present, immediately write the block out. This process is repeated until all data blocks have been written on the output tape. When sentinel is detected and the final string ended, the last block has already been written out and the write sequence must be by-passed. In the final routines three sentinel blocks are written on the output tape, the tape label is printed out, and the tape is rewound. The merging process is now completed.

An output tape may have been filled, and the alternate tape activated, before the end of the last string has been detected. Because the write operation precedes the read, the last data block and three sentinel blocks have been written on the previous output tape, and it would be useless to write a block of sentinel on a new output tape. Therefore, the program is immediately concluded following the detection of a new output tape.

Figure 1. TAPE IDENTIFICATION MASTER CHART

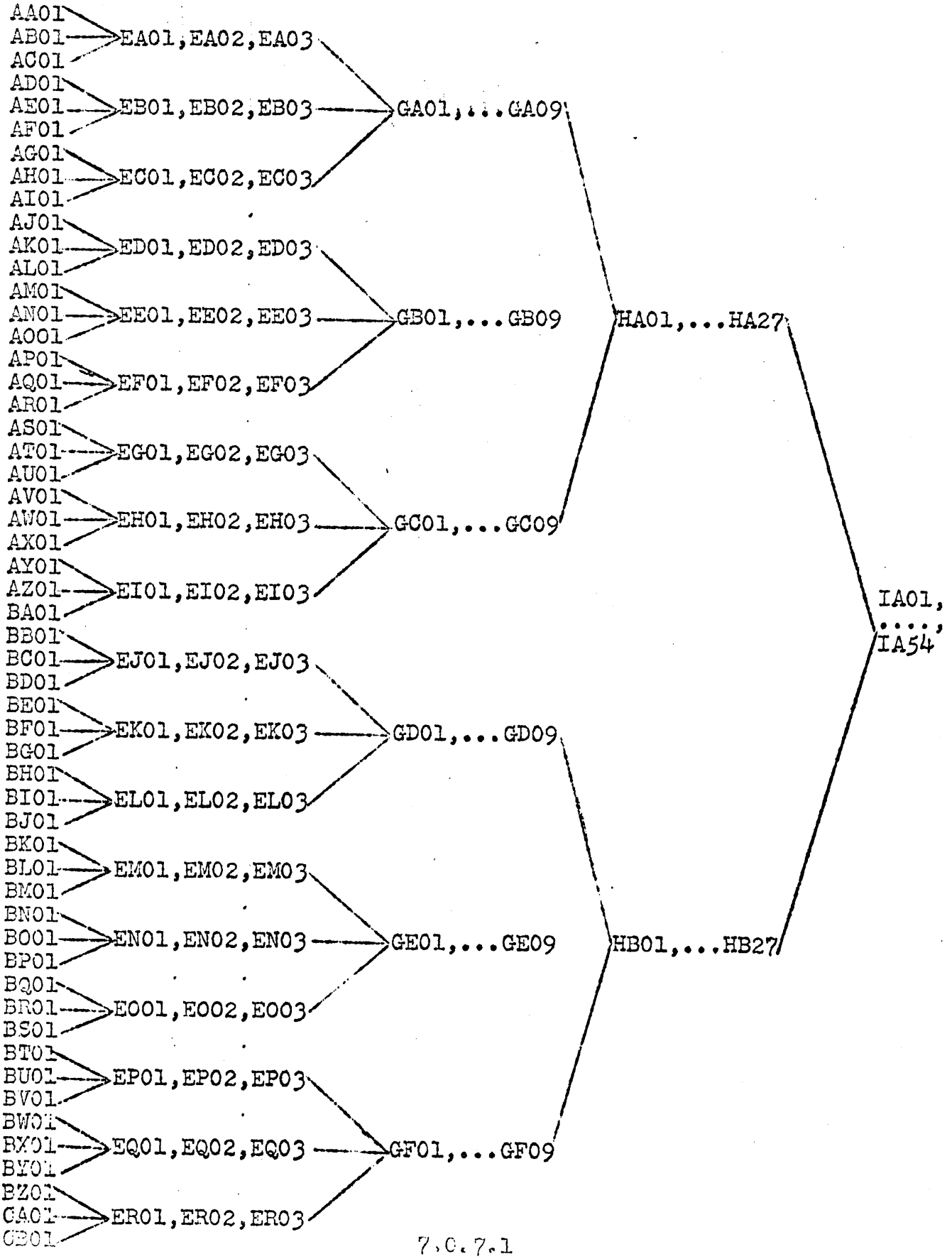


Figure 2. A TYPICAL IDENTIFICATION BLOCK

q_0	Sentinel
q_1	Current Output identification no.
q_2	Current A Input identification no.
q_3	Current B Input identification no.
q_4	Current C Input identification no.
q_5	Final A Input identification no.
q_6	Final B Input identification no.
q_7	Final C Input identification no.
q_8	Keyword of last item on previous tape.
q_9	Current location in A block.
q_{10}	Current location in B block.
q_{11}	Current location in C block.
q_{12}	Keyword of first item on current output tape.

q_0	ZZZZZZ
q_1	HA16
q_2	GA06
q_3	GB07
q_4	GC04
q_5	GA09
q_6	GB09
q_7	GC09
q_8	Smith, J.G.
q_9	532
q_{10}	664
q_{11}	784
q_{12}	Smith, K.B.

Figure 3. INPUT STRING LENGTHS FOR MERGING 99 TAPES

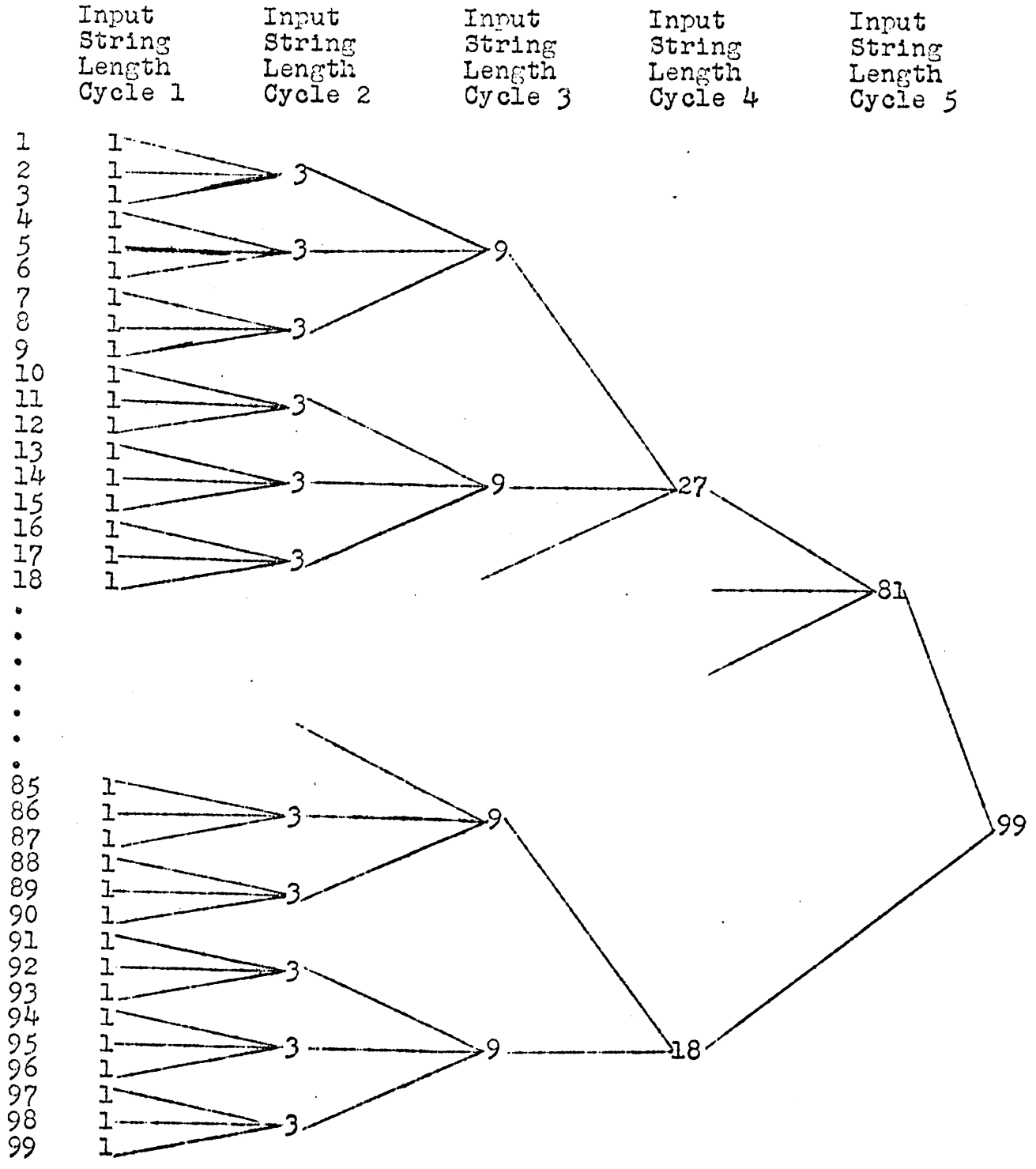


Figure 4. BLOCKS OF IGNORE SYMBOLS TO BE DISCARDED IN MERGING

10 Word Item

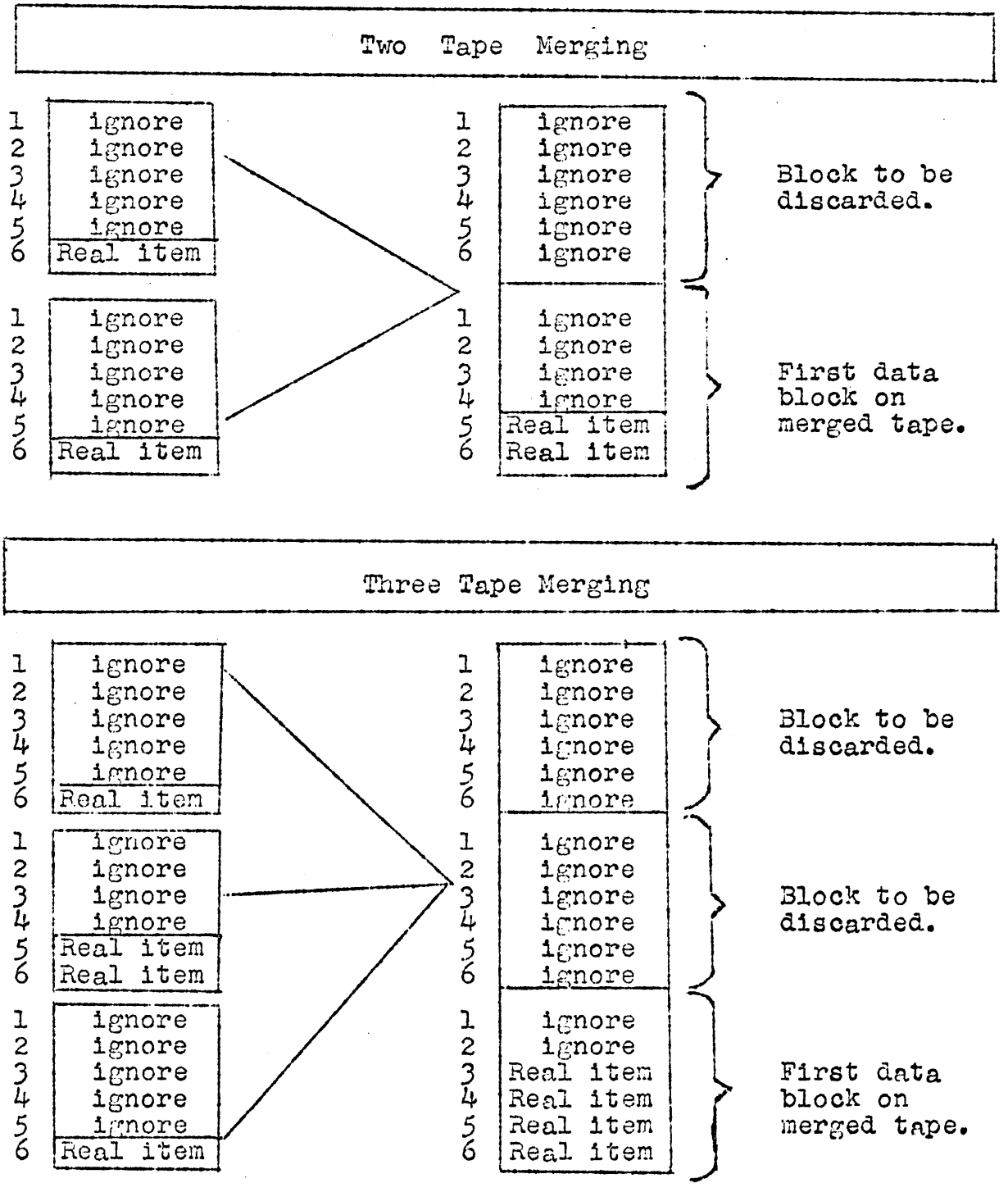
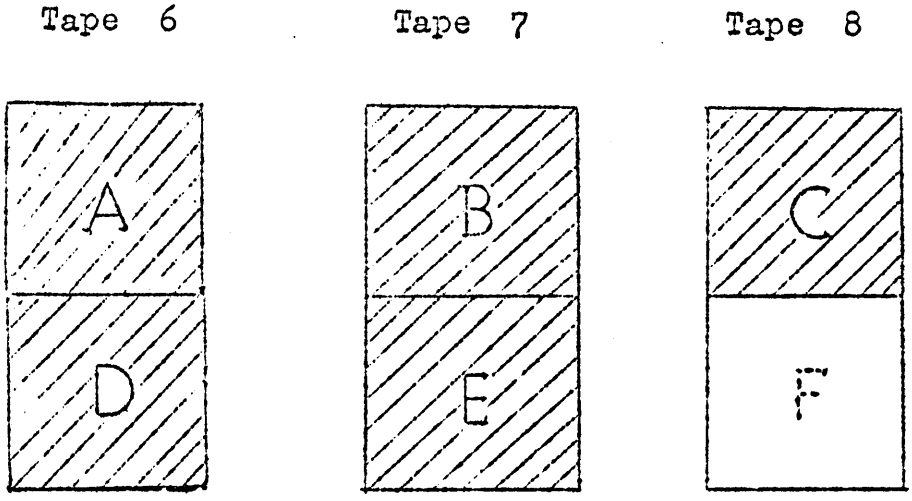


Figure 5.

INITIAL STATUS
OF DATA STORAGE BLOCKS



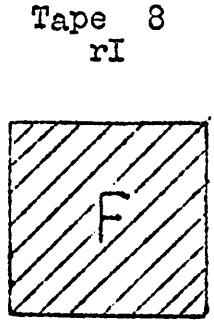
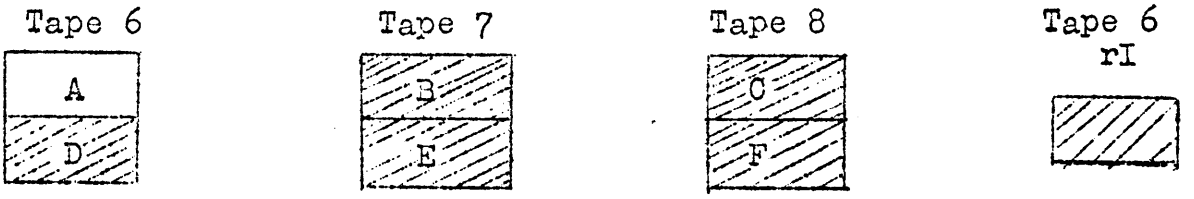
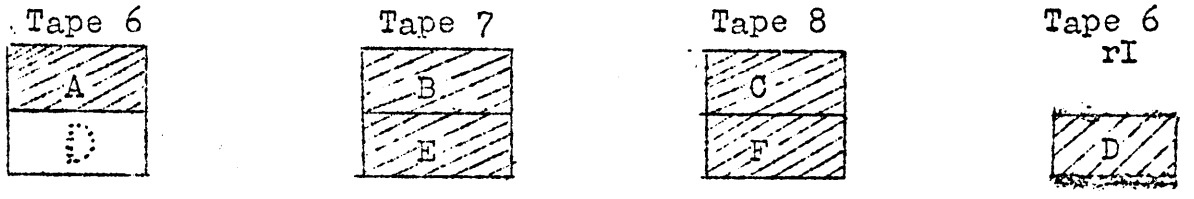


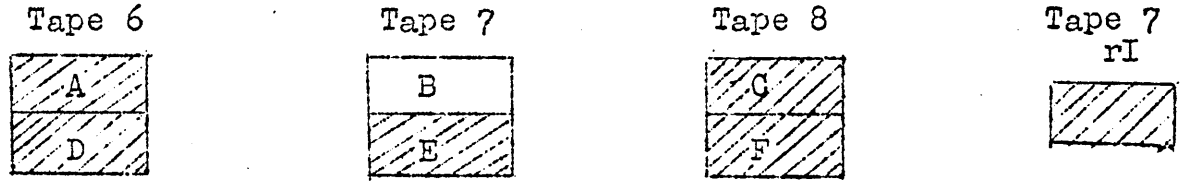
Figure 6. PROCESS OF EMPTYING AND REFILLING DATA BLOCKS



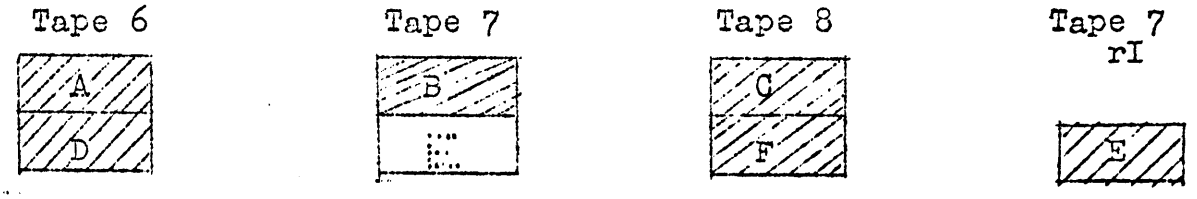
1. Block A empties
2. rI to auxiliary block F
3. Tape 6 to rI



1. Auxiliary block D to A
2. rI contains data to be transferred to block D when next data block empties



1. Block B empties
2. rI to auxiliary block D
3. Tape 7 to rI



1. Auxiliary block E to B
2. rI contains data to be transferred to block E when next data block empties

CHAPTER 8

TIMING OF COLLATION ROUTINES

8.0.1. Introduction.

This chapter presents, by means of formulae and charts, the times required to completely collate tape lengths of items of given sizes.

Section 8.0.2. defines the symbols used in the formulae.

Section 8.0.3. gives the assumptions on which the timing studies were based.

Sections 8.0.4,, 8.0.5., and 8.0.6., are analyses of the methods used and the formulae derived in timing Internal Collation, External Collation, and Merging respectively.

Section 8.0.7. summarizes the times given in the previous sections and shows the total collation times required.

8.0.2. Definition of Symbols.

Σ' The number of instruction words required to transfer one item from one location in the memory to another. The value of Σ' is one for a ten-word item (Ym, Zm), a two-word item (Vm, Wm), and a one-word item (Bm, Cm).

In the generalized flow chart form for External Collation and Merging, the symbol Σ represents the number of words per item.

C The number of Merging runs needed to arrange a given quantity of individually ordered tapes into a continuous string of ordered tapes. One run is required for each intermediate output string, as well as for the final output string.

H The number of External Collation cycles required to arrange in monotonic sequence the information on a single tape. The number of blocks on a tape and the method of External Collation (two-way, three-way, or four-way) determine the value of H.

I The amount of interlock time per block in seconds. Interlock time is the period during which computation is halted while a block of information is read into rI.

M The total number of blocks of information on a tape.

N The total number of items per block.

t The amount of computer time, in seconds, required to execute one data transfer.

T The number of tapes, or fractions of tape, containing unordered information.

W The number of Internal Collation cycles required to arrange a block of information in monotonic sequence. The value of W is dependent upon the number of items per block.

8.0.3. Assumptions in the Timing of Programs.

The coded routines in this report were analyzed and the times tabulated according to the computer times of operation listed in Code C-10. Where a choice of coding techniques existed, the more efficient technique was used in the timing analysis, and the formulae were derived on this basis.

The time allocated to Supervisory Control operations, computer breakdown, and tape changes,have not been accounted for in the timing equations.

The External Collation, and Merging routines, lend themselves quite easily to a generalized equation. Internal Collation, however, which requires individual programming for each item size, necessitates a separate calculation for each of these item sizes.

All the programs included in this timing study require an extraction of digits from a single word to obtain the key information. Weighting factors were applied to the repeated sequences a routine might employ, in order to produce a maximum, rather than

an average, computer time in the formulae.

The extraction of key digits from the items to be compared is not the most economical method to use in collation. The sequence of instructions required is F(m), E(m), K 000, E(m), and T(m). The most efficient method is available when the key digits are located at the extreme left in the word. The sequence of instructions necessary in this instance is only B(m), L(m), and T(m).

The table below lists the percentage of time which is saved by this alteration.

Words per Item	% of Time Saved in Three-Way	% of Time Saved in Two-Way
2	5.5	6.0
3	4.0	4.5
4	4.5	5.0
5	3.0	3.5
6	3.5	4.0
8	3.0	3.5
10	3.5	4.5

The key digits may occur at the right of one word and be continued in the next. If this is the case the time spent in the comparison sub-routine is increased. The percentage increase in the total time due to split key digits is listed in the table below.

Words per Item	% Increase for Three-Way	% Increase for Two-Way
2	5.0	3.0
3	4.0	2.0
4	4.0	2.0
5	3.0	1.5
6	3.5	2.0
8	3.0	1.5
10	3.5	2.0

8.0.4. Timing of Internal Collation.

Routine 940-4, which begins on page 5.1.3.1, collates by means of a function table. In this routine the input block is transferred to the output block, item by item. The transfer orders are fabricated for each item and, after each transfer, a test is made to determine when the final item has been transferred. This procedure is correct but very time-consuming. A more efficient method is to fabricate the transfer orders for the entire block. This was done in timing the routine, and comparison showed a time saving of ten percent.

Coding from 940-4

```
064  L  182  U  082
     *********
080  000000  000000

081 [V  000  W (900)]
082 [B (510) 06 000]
083  A  081  H  087
084  A  144  H  088
085  X  000  H  089
086  X  000  E  080
087 [V (xxx) W (900)]
088 [V ( +2) W (902)]
089 [V ( +4) W (904)]
090  C  081  B  082
091  A  142  Q (→ °P)
092  C  082  U  082
```

Subroutine Timed

```
064  F  080  U  081
     *********
080  000000  000111

081  B  116  E  510
082  H  116  A  144
083  H  117  X  000
084  H  118  X  000
085  E  511  H  119
086  X  000  H  120
087  X  000  H  121
088  X  000  E  512
     *********
116  W  898  V (xxx)
117 [W  900  V ( +2)]
118 [W  902  V ( +4)]
119 [W  904  V (xxx)]
120 [W  906  V ( +2)]
     *********
146  W  958  U (→ °P)
```

Previously, it was stated that a general equation is not applicable to Internal Collation. For this reason, no formula is presented. The chart below shows the times required to carry out Internal Collation for given item sizes.

Words per Item	No. of Items per Block	Time per block in Seconds	No. of Items per Tape	Time per Tape in Hours
2	30	1.62	60,000	.911
3	20	1.25	40,000	.707
4	15	.830	30,000	.468
5	12	.684	24,000	.386
6	10	.535	20,000	.302
8	7	.322	14,000	.182
10	6	.085	12,000	.048

A method of saving time has been proposed for those routines which handle blocks containing a number of items not equal to a perfect power of two. The routine for collating a six-word item may be used as an example. At the end of the second cycle there are two strings of four items each and one string of two items. In the present program the two four-item strings are collated to form a string of eight items. This string is then collated with the two-item string. The proposed change is to collate the second four-item string with the two-item string to form a string of six items which is subsequently collated with the remaining four-item string. Should this proposal be adopted, a saving of five percent of the Internal Collation time would be effected.

8.0.5. Timing of External Collation.

The routines for both two-way and three-way External Collation have been timed and formulae derived. Since both routines follow the same general pattern the formulae are basically similar. The two-way External Collation routine has been programmed to completely eliminate tape interlock. The three-way External Collation routine, however, cannot be programmed to do this because of insufficient memory space. This is the only significant difference between the two routines, and the equations derived for them differ only on this account.

The equation for two-way External Collation is:

$$\text{Time in hours} = .000283\ T\left[H\left(2.1 + M\left[.0339 + N\left(.00936 + \Sigma'\left[.00073 + t\right]\right)\right]\right) + 7.3 + M\left(.00461 - .00655(1/2)^{H}\right) - N\left(.00907 + \Sigma'\left[.00073 + t\right]\right)\right]$$

Where:

$$2^{H-1} < M \leq 2^{H}$$

t = .00089 seconds for a one-word transfer

t = .00097 seconds for a two-word transfer

t = .00130 seconds for a ten-word transfer

and,

$\left[.00073 + t\right]$ = time, in seconds, to fabricate and transfer an item. Should $\Sigma' = 1$, the term appears as $\left[.00073 + t - .00024\right]$ or, $\left[.00049 + t\right]$

$\left(.00936 + \Sigma'\left[.00073 + t\right]\right)$ = time, in seconds, to compare, transfer, and count one item.

$\left[.0339 + N\left(.00936 + \Sigma' \left[.00073 + t\right]\right)\right]$ = time, in seconds, to transfer one block from the input to the output.

$\left(2.1 + M \left[.0339 + N \left(.00936 + \Sigma'\left[.00073 + t\right]\right)\right.\right.$ = time, in seconds, to complete one External Collation cycle.

7.3 = total time, in seconds, for the initial reading of instructions, the writing of the label block on the final output tape, and the completion of the final output tape.

$M\left(.00461 - .00655(1/2)^H\right)$ = time, in seconds, for executing the end of string routines.

$-N\left(.00907 + \Sigma' \left[.00073 + t\right]\right)$ = time, in seconds, to write the first data block on the final output tape. (This time is already included in the time to write the label block on the final output tape).

.000283 = factor to change seconds to hours and allow for the periodic memory check.

The equation for three-way External Collation is:

Time in hours =

$$.000283\ T\left[H\left(3.0 + M\left[.0241 + N\left(.01083 + \Sigma'\left[.00073 + t\right]\right) + I\right)\right.\right.$$
$$\left.\left. + 6.4 + .00927\ M\left[1-(1/3)^H\right] - N\left(.01045 + \Sigma'\left[.00073 + t\right]\right)\right]$$

Where:

I = interlock time

All other terms are defined as in the two-way External Collation formula.

The times for tape interlock were obtained from the results of a routine computed on the BINAC. The table below shows the percentage increase per block caused by tape interlock.

Words per Item	% Increase in Time per Block
2	3
3	3
4	3
5	4
6	5
8	8
10	19

The derivation of a single term common to both formulae is presented as an example. The term chosen is .00073 Σ' which is the amount of time, in seconds, to fabricate the instructions for transferring an item. The coding from which this term was derived is shown below.

100	B	xxx	Place in rA the location of the input item to be transferred next.
	F	xxx	Place in rF the instruction extractor.
101	E	xxx	Extract the output location around the contents of rA to form the transfer order.
	H	(m)	Hold this transfer order in rA and also in m.
102	A	xxx	Add a constant to the transfer order.
	H	(m+1)	Hold this transfer order in rA and also in m.
103	X	000	Add the same constant to the transfer order.
	H	(m+2)	Hold this transfer order in rA and also in m.

The times required to execute these lines of coding are:

line 100 = .00089 seconds

line 101 = .00089 seconds

line 102 = .00097 seconds

line 103 = .00073 seconds

Line 103 would be repeated ($\Sigma'-2$) times for all items requiring more than two transfers. Therefore the times in the formula may be expressed as:

$$.00089 + .00089 + .00097 + .00073(\Sigma'-2).$$

Combining these terms, the result is:

$$.00275 + .00073\ \Sigma' - .00146 \quad \text{or,}$$

$$.00129 + .00073\ \Sigma' \text{ seconds.}$$

These two terms must be multiplied by NMHT to obtain the total time required by the computer to fabricate the necessary transfer instructions.

It has been stated previously that a special case arises for $\Sigma' = 1$. The lines of coding required are 100 and 101 giving a total time of .00178 seconds. Since the formulae have been derived for $\Sigma' > 1$, a quantity, .00024, must be subtracted for the special case of $\Sigma' = 1$. This quantity is derived as follows;

$$.00089 + .00089 + .00097 + .00073\ (1-2)$$

$$.00178 + .00097 - .00073$$

$$.00178 + .00024 \text{ seconds.}$$

The table below shows the time in hours for Externally collating one complete tape.

Words per Item	Two-Way (Hr.per Tape)	Three-Way (Hr.per Tape)
2	2.23	1.61
3	1.99	1.39
4	1.41	.978
5	1.52	1.04
6	1.12	.770
8	.923	.636
10	.636	.474

8.0.6. Timing of Merging.

The routines for two-way and three-way Merging have been timed, and formulae derived for both. The tape interlock factor is present in Merging as it was in External Collation, and again causes an additional term in the three-way formula.

Both programs are designed to produce only one string of merged tapes per run. When a large number of tapes must be merged, many runs are required.

A minimum number of input tapes per run produces the best conditions for computer operating time.

Each full output tape resulting from either Merging program is designated as one tape pass. Formulae have been derived to obtain the fewest possible tape passes, thereby gaining optimum efficiency.

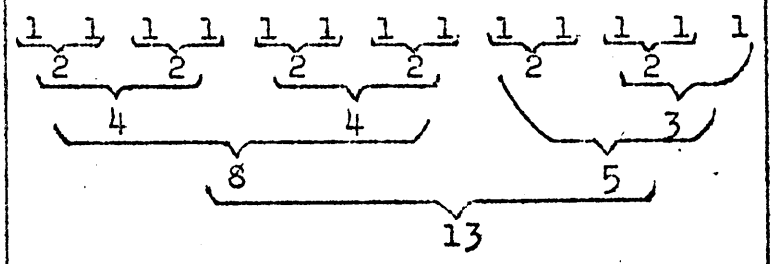
$T \frac{\log T}{\log 2}$ = Optimum number of tape passes for two-way Merging.

$T \frac{\log T}{\log 3}$ = Optimum number of tape passes for three-way Merging

If these formulae do not produce integral values the next highest integers are used. The optimum cannot always be achieved as is shown in the second of the two examples given. The first example is the merging of 13 tapes by the two-way method.

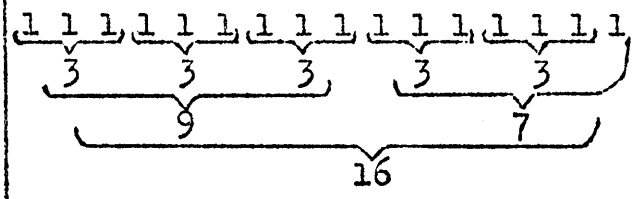
$$T = \frac{\log T}{\log 2} = 48.1 = 49$$

Six strings of two tapes each are formed on the first six runs. This process involves twelve tape passes. There are now six strings of two tapes each and one string of one tape. The one-tape string is now merged with any of the two-tape strings to form a single three-tape string and complete the seventh run. There have now been three more tape passes. In runs eight and nine, four of the remaining five two-tape strings are combined to form two strings of four tapes each. This requires eight more tape passes, making a total of 23. There are now four strings; one two-tape string, one three-tape string, and two four-tape strings. In the tenth run the two-tape string is merged with the three-tape string to form a single string of five tapes. The total number of tape passes is now 28. In the eleventh run, the two strings of four tapes each are merged into one string of eight tapes. This increases the number of tape passes to 36, and leaves two strings, one of five tapes and the other of eight tapes. In the twelfth and final run the two remaining strings are merged to form the required single string of 13 tapes. This requires 13 tape passes and increases the total to 49, which was the optimum number.

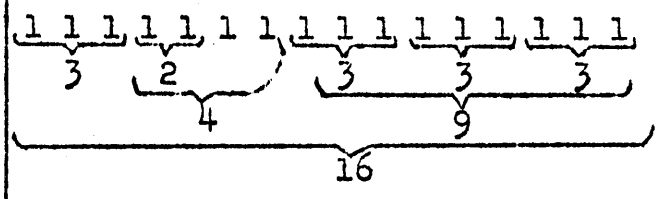
Length of Strings (No. of Tapes)	No. of Tape Passes
	12 11 13 13 Total = 49

The second example is the merging of 16 tapes by the three-way method:

$$T = \frac{\log T}{\log 3} = 40.4 = 41$$

Length of Strings (No. of Tapes)	No. of Tape Passes
	15 16 16 Total = 47

The total number of tape passes is too large in this case. It may be noted that only two strings are present for the final run and two-way Merging is used. It is more efficient to use two-way Merging in an earlier run, as shown below.

Length of String (No. of Tapes)	No. of Tape Passes
	14 13 16 Total = 43

The number of tape passes is still too large but this system appears to be the best available for merging 16 tapes.

The general equation for two-way Merging and the explanation of its terms are:

Time in hours =

$$.000283\left[T\,\frac{\log T}{\log 2}\left(4.16 + M\left[.0251 + N\left(.00931 + \Sigma'\left[.00073 + t\right]\right)\right]\right) + 3.411C - NC\left(.0068 + \Sigma'\left[.00073 + t\right]\right)\right]$$

Where:

t = .00089 seconds for a one-word transfer

t = .00097 seconds for a two-word transfer

t = .00130 seconds for a ten-word transfer

and,

$[.00073 + t]$ = time, in seconds, to fabricate and transfer an item. Should $\Sigma' = 1$, the term appears as $[.00073 + t - .00024]$ or, $[.00049 + t]$

$\left(.00931 + \Sigma'\left[.00073 + t\right]\right)$ = time, in seconds, to compare, transfer, and count one item.

$\left[.0251 + N\left(.00931 + \Sigma'\left[.00073 + t\right]\right)\right]$ = time, in seconds, to transfer one block from the input to the output.

$\left(4.16 + M\left[.0251 + N\left(.00931 + \Sigma'\left[.00073 + t\right]\right)\right]\right)$ = time, in seconds, to complete one tape pass.

$T\,\frac{\log T}{\log 2}$ = the number of tape passes.

3.411 = the initial read time.

$-N\left(.0068 + \Sigma' \left[.00073 + \bar{t}\right)\right.$ = time, in seconds, to write out the first data block. (This time is already included in the time to write the label block on the final output tape.)

.000283 = factor to change seconds to hours and allow for the periodic memory check.

The values of $T \frac{\log T}{\log 2}$ and C are obtained from the chart below.

Total No. of Tapes in Final String	$T \frac{\log T}{\log 2}$	C
2	2	1
3	5	2
4	8	3
5	12	4
6	16	5
7	20	6
8	24	7
9	29	8
10	34	9
11	39	10
12	44	11

The equation for three-way Merging is similar to that for two-way Merging, and the explanations of the terms are the same. The three-way Merging routine presented in this manual does not include an extract order in the main comparison sequence. There-

fore, in establishing the formula, extract orders were added to the existing coding. The equation is:

Time in hours =

$$.000283\left[T\frac{\log T}{\log 3}\left(4.2 + M\left[.0255 + N\left(.00917 + \Sigma'\left[.00073 + \overline{t}\right)\right] + I\right]\right)\right.$$
$$\left. -C\left[1.66 + N\left(.00917 + \Sigma'\left[.00073 + \overline{t}\right)\right]\right]\right]$$

The tape interlock term is present in the three-way Merging formula as it is in the three-way External Collation formula. The total time per block is obtained by increasing the computer time for handling one block from input to output to include interlock time. The percentage increases are shown in the table below.

Words per Item	% Increase In Time per Block
2	3
3	3
4	4
5	5
6	6
8	9
10	21

The values of $T \frac{\log T}{\log 3}$ and C are obtained from the chart below.

Total No. of Tapes in Final String	$T \frac{\log T}{\log 3}$	C
2	2	1
3	3	1
4	6	2
5	8	2
6	11	3
7	13	3
8	16	4
9	18	4
10	22	5
11	25	5
12	29	6

The total time required to accomplish one tape pass in the Merging routines is shown in the chart which follows.

Words per Item	Two-Way (Seconds per Tape Pass)	Three-Way (Seconds per Tape Pass)
2	700.6	758.8
3	621.2	664.8
4	435.7	471.0
5	472.2	511.4
6	342.6	375.4
8	279.9	313.0
10	187.6	235.6

8.0.7. Total Collation Time for One to Twelve Tapes.

The number of hours required to collate and merge from one to twelve tape lengths of information has been computed and is shown in the following charts. If a combination of two-way and three-way collation is used, the time will lie somewhere between the two quantities shown.

I = thousands of items; time is given in hours.

Words per Item	1 Tape			2 Tapes			3 Tapes			4 Tapes		
	I	2-way	3-way	I	2-way	3-way	I	2-way	3-way	I	2-way	3-way
2	60	3.2	2.5	120	6.7	5.5	180	10.4	8.2	240	15.2	11.4
3	40	2.7	2.1	80	5.8	4.6	120	9.0	6.9	160	12.2	9.5
4	30	1.9	1.5	60	4.0	3.2	90	6.3	4.8	120	8.5	6.6
5	24	1.9	1.4	48	4.1	3.2	72	6.4	4.7	96	8.7	6.6
6	20	1.4	1.1	40	3.1	2.4	60	4.8	3.5	80	6.5	4.9
8	14	1.1	0.8	28	2.4	1.8	42	3.7	2.7	56	5.1	3.8
10	12	0.7	0.5	24	1.5	1.2	36	2.3	1.8	40	3.2	2.5

Words per Item	5 Tapes			6 Tapes			7 Tapes			8 Tapes		
	I	2-way	3-way	I	2-way	3-way	I	2-way	3-way	I	2-way	3-way
2	300	18.1	14.3	360	22.0	17.5	420	26.0	20.4	480	29.9	23.6
3	200	15.6	12.0	240	19.0	14.6	280	22.4	17.1	320	25.8	19.8
4	150	10.9	8.3	180	13.2	10.1	210	15.7	11.9	240	18.0	13.7
5	120	11.1	8.3	144	13.6	10.2	168	15.0	11.9	192	18.5	13.7
6	100	8.3	6.2	120	10.1	7.6	140	11.9	8.9	160	13.7	10.3
8	70	6.5	4.8	84	7.9	5.9	98	9.3	6.9	112	10.7	8.0
10	60	4.1	3.1	72	5.0	3.8	84	6.1	4.5	96	6.8	5.3

Words per Item	9 Tapes			10 Tapes			11 Tapes			12 Tapes		
	I	2-way	3-way	I	2-way	3-way	I	2-way	3-way	I	2-way	3-way
2	540	34.0	26.6	600	38.0	30.0	660	43.0	33.0	720	47.0	37.0
3	360	29.4	22.3	400	33.0	25.1	440	37.0	27.8	480	40.0	31.0
4	270	20.5	15.4	300	23.0	17.4	330	25.5	19.2	360	28.0	21.2
5	216	21.0	15.4	240	23.6	17.4	264	26.2	19.3	288	28.8	21.3
6	180	15.6	11.6	200	17.5	13.1	220	19.4	14.5	240	21.3	16.0
8	126	12.2	9.0	140	13.7	10.1	154	15.2	11.2	168	16.7	12.4
10	108	7.7	5.9	120	8.7	6.7	132	9.6	7.4	144	10.5	8.2

When the above values are graphed the resulting curves are nearly linear. The time required to collate more than twelve tapes of data may then be extrapolated, using a correction factor to compensate for the non-linearity.

Collation times determined from the tables below incorporate extrapolated values. To calculate the total time required to collate a given number of items by the three-way method, the following formula is used.

Time in hours = (no. of items) X (next higher power of three) X (time per item).

The time per item per tape pass is given for items of various sizes in the second of the two tables. The number of tape passes required for the items to be collated is the corresponding power of three. This value can be found in the first table below. If the number of items to be collated falls between any two entries in this table, the next higher power of three is used.

Time in hrs. = (no. of items) X (next higher power of three)
X (time per item)

Power of Three	Number of Items
10	59,100
11	177,000
12	532,000
13	1,600,000
14	4,780,000
15	14,350,000
16	43,050,000
17	129,000,000
18	388,000,000
19	1,160,000,000
20	3,490,000,000

Words per Item	Time (Hrs.) per Item
2	.000 004 12
3	.000 005 48
4	.000 005 17
5	.000 006 50
6	.000 005 88
8	.000 006 93
10	.000 005 33

The tables presented so far have shown the total time required to completely collate items of various sizes. The charts which follow show the percentage of the total time applicable to each of the three processes required in collation.

Words per Item	One Tape		
	% Internal	% External	% Merging
2	36	64	00
3	34	66	00
4	32	68	00
5	27	73	00
6	28	72	00
8	22	78	00
10	9	91	00

Words per Item	Ten Tapes		
	% Internal	% External	% Merging
2	31	54	15
3	28	56	16
4	27	57	16
5	22	61	17
6	23	60	17
8	18	64	18
10	7	72	21

Words per Item	One Hundred Tapes		
	% Internal	% External	% Merging
2	27	48	25
3	25	49	26
4	24	49	27
5	19	52	29
6	20	51	29
8	16	54	30
10	6	60	34

The above percentages are for three-way collation. For the two-way method the amount of time for External Collation and Merging routines is increased while that for Internal is constant. Therefore, the percentage of time for Internal Collation is effectively decreased while the others are increased.

CHAPTER 9

APPENDIX

9.0.1. Flow Chart Notation.

Every flow chart included in this report indicates the manner in which some phase of the collation problem has been solved. The coded routine associated with each is the extension of this solution to a series of UNIVAC instructions. For a complete overall picture of the processing methods employed in any given instance, the flow chart should be studied first. Then, when control requirements are clearly understood, techniques used in coding are more easily mastered.

The initial step in the analysis of any problem should be the development of a flow chart. An experienced programmer assigned to this task can demonstrate, by diagram, the manner of obtaining a correct solution by computer methods and, at the same time, indicate the most efficient sequence of UNIVAC operations. The possibility of introducing logical errors when the individual routines are finally coded, is thereby reduced.

The flow charts of the collation programs include most of the symbols which are described and explained in MP-2. Although the latter publication was released before most of the present report had been written, the final notation was established during the time when the collation routines were being prepared. The existing discrepancies are primarily in the earlier programs. A few have been carried on into later programs for consistency.

In so far as possible, analogous routines in related collation programs have the same connector numbers. The variations which occur arise from differences in the processing required. Programs which are unrelated, but which are parallel in nature, have similar if not identical connector designations. The lettered routines, such as Q, L, and P are common to many.

The use of the R and S connectors to indicate the end-of-string and end-of-group routines is a further use of lettered connectors. Because these routines are parallel for all the input groups, the parallel notation has been introduced as well. It is not general practice, however, to number the first of a group of variable connectors with zero, as in RA0 or SB0. Initially, this was done to assure that the tenth variation on a connector of this type might still be numbered with a single digit. In later programs, the procedure was continued to maintain consistency.

Throughout the collation programs, an attempt was made to avoid the use of any alphabetic character to represent more than one constant or counter. Although the effort was not completely successful, duplication has been minimized. In general, constant values have been designated by capital letters, and counters and other variables have been designated by lower case letters. The number of words per item, which varies from program to program, is usually represented by Σ .

Using a notation of this sort, it is possible to construct a flow chart which is independent of item size, unless the number of items to be processed is an important factor in determining

method. In the Internal Collation programs, the establishment of groups, strings, and even cycles is closely linked with the number of items to be processed. A generalized flow chart, therefore, cannot be used. In External Collation and Merging, however, item size does not affect the processing in the same way, and the generalized flow chart is equally applicable for items of two, ten, or any other number of words. For this reason, the conversion of existing programs to those of other item sizes can be accomplished with a minimum of difficulty.

9.0.2 Glossary of Flow Chart Symbols

Constants and Counters

A — External Collation: the storage location for one block of data from the first data input tape; and also, the current working location in that block.

Internal Collation: the storage location for the first data group, and also, the current location in that group.

B — External Collation: the storage location for one block of data from the second data input tape; and also, the current working location in that block.

Internal Collation: the storage location for the second data group, and also, the current location in that group.

Internal Collation Subroutine: the current position in the first group of 32 two-word items.

C — Three-way External Collation: the storage location for one block of data from the third data input tape; and also, the current working location in that block.

Two-way External Collation: the first auxiliary storage location for data from the first data input tape.

Internal Collation: a temporary storage location.

Internal Collation Subroutine: the current position in the second group of 32 two-word items.

D — Three-way External Collation: the first auxiliary storage location for the first data input tape.

Two-way External Collation: the first auxiliary storage location for the second data input tape.

Internal Collation: a temporary storage location.

E — Three-way External Collation: the first auxiliary storage location for the second data input tape.

Two-way External Collation: the second auxiliary storage location for the first data input tape.

Internal Collation: a temporary storage location.

F Three-way External Collation: the first auxiliary storage location for the third data input tape.

Two-way External Collation: the second auxiliary storage location for the second data input tape.

Internal Collation: a temporary storage location.

G The storage location for one block of output data; and also, the current position in that block.

H The number of External cycles required to complete the collation of a data tape.

J Internal Collation: the function table address of the last item to be considered in group one. Also, the storage location for one block of data from the data input tape.

K Internal Collation: the function table address of the last item to be considered in group two. Also, the storage location for one block of output data.

L The input function table. A sequence of key digit storage locations which indicates the order in which the items to be collated are to be considered during the current cycle of operation.

M The maximum number of blocks to be found on a complete tape.

N The number of items in one block.

R The number of units from each input group which are to be collated together to form an output string. In Internal Collation, the unit is an item; in External Collation, the unit is a block.

S The sentinel which indicates the end of a tape.

S.C. The abbreviation for Supervisory Control.

T The symbol for tape.

U — A counter which is used to indicate whether or not the special descending series instructions have been used; U = 1 when the special cycle has been used, U = 0 when the special cycle has not been used.

V — A complete word of ignore symbols.

X — Two-way External Collation: either of the two auxiliary blocks allocated to the first data input tape.

Internal Collation: the number of items from group one to be collated during the current cycle. Also, the first item in both groups (differentiated by subscript).

Y — Three-way External Collation, special series: the address of the data currently stored in register I.

Two-way External Collation: either of the two auxiliary blocks allocated to the second data input tape.

Internal Collation: the number of items from group two to be collated during the current cycle. Also, the second item in both group one or group two (differentiated by subscript).

Z — Two-way method with tape interlock eliminated: the address of the data currently stored in register I.

Internal Collation: the third item in both input groups (differentiated by subscript).

Z+1 — Two-way method, with tape interlock eliminated: the address of the data which will next be stored in register I.

a — Merging: a constant used to alternate the servos allocated to the input tapes of group one.

Internal Collation: the satellite digits associated with the first item in both input groups (differentiated by subscript).

Internal Collation: the counter used to tabulate the number of items to be collated from group one.

b Merging: a constant used to alternate the servos allocated to the input tapes of group two.

Internal Collation: the satellite digits associated with the second item in both input groups (differentiated by subscript).

Internal Collation: the counter used to tabulate the number of items to be collated from group two.

c Merging: a constant used to alternate the servos allocated to the input tapes of group three.

Internal Collation: the satellite digits associated with the third item in both input groups (differentiated by subscript).

f A counter used to count the number of blocks on an input or output tape.

g A temporary storage location for the current $^{o}12b$.

i The current variation on connector 12b.

j The function table address of the group one item which is currently being collated. Also, a counter used to tabulate items transferred from the input block J.

k The function table address of the group two item which is currently being collated.

m A counter used to tabulate items transferred to the current output block.

n Internal Collation Subroutine: the number of items in a group to be compared for a single string in the current cycle, and also, the number of items transferred from a string in either group (differentiated by subscript).

p A power of two, used to aid in determining H for two-way External Collation. A power of three, used to aid in determining H for three-way External Collation.

q The storage location for one word of tape identification.

r Three-way External Collation: a counter used to tabulate the blocks which have been transferred to the current string on the first output tape.

s Three-way External Collation: a counter used to tabulate the blocks which have been transferred to the current string on the second output tape.

Two-way External Collation: a counter used to tabulate the blocks which have been transferred to the current string on the first output tape.

t Three-way External Collation: a counter used to tabulate the blocks which have been transferred to the current string on the third output tape.

Two-way External Collation: a counter used to tabulate the blocks which have been transferred to the current string on the second output tape.

u A counter used to tabulate the blocks of the current string in group one, which have been processed.

v A counter used to tabulate the blocks of the current string in group two which have been processed.

w A counter used to tabulate the blocks of the current string in group three which have been processed.

x A counter used to tabulate the items from input block A which have been transferred to the output block.

y A counter used to tabulate the items from input block B which have been transferred to the output block.

z A counter used to tabulate the items from input block C which have been transferred to output block G.

α The alphabetic characters associated with the tape identification of the first input string.

β The alphabetic characters associated with the tape identification of the second input string.

γ The alphabetic characters associated with the tape identification of the third input string.

δ The alphabetic characters assigned to the identification of tapes in the output string.

Σ The number of words in one item.

Subscripts

A Pertaining to the first data group, as in T_A.

B Pertaining to the second data group, as in T_B.

C Pertaining to the third data group, as in T_C.

K or k Pertaining to the current input group, as in T_K.

Pertaining to the keyword of the first item, as in G_K.

N Pertaining to the keyword of the Nth item, as in G_N.

P or p Pertaining to the current output tape, as in T_P.

X Pertaining to the first output tape, as in T_X.

Y Pertaining to the second output tape, as in T_Y.

Z Pertaining to the third output tape, as in T_Z.

a Pertaining to the first group of key digits in an item with split key, as in B_a.

b Pertaining to the second group of key digits in an item with split key, as in B_b.

ℓ A table of key digit addresses, in sequence according to the order of the items in the data storage block, as in L_ℓ.

n The numeric digits of the label associated with the last tape in a string.

y Pertaining to the output function table, as in L_y.

The subscript is also used as a counter to tabulate items transferred from the input block to the output block according to the sequence specified by L_y.

Connectors

L The routine which is entered when one block of input data has been completely processed, and the contents of an auxiliary storage location are to be transferred to the working location.

P The routine which is entered when the current output block is complete, and that block is to be written on an output tape.

Q The routine which is entered in order to compare the key digits of the current items from the input groups.

R The routine which is entered when the current input string is exhausted.

S The routine which is entered when the current input tape is exhausted.

T The routine which is entered when both input tapes have been exhausted in routine 952-2.

W The routine which is entered in order to make adjustments in the Merging program preparatory to the rerun of a merged tape.