THE MARC SORT PROGRAM

John C. RATHER: Specialist in Technical Processes Research, and
Jerry G. PENNINGTON: Information Systems Mathematician,
Library of Congress, Washington, D.C.

Describes the characteristics, performance, and potential of SKED (Sort-Key Edit), a generalized computer program for creating sort keys for MARC II records at the user's option. SKED and a modification of the IBM S/360 DOS tape sort/merge program form the basis for a comprehensive program for arranging catalog entries by computer.

THE ROLE OF SORTING IN THE MARC SYSTEM

Many present and potential uses of cataloging data in machine readable form require that the input sequence of the records be altered before output. The production of book catalogs, bibliographical lists, and similar output products benefits from an efficient means for arranging the records in a more sophisticated way than mere alphabetical order or, even worse, the collating sequence of a particular computer. Internal files, such as special tape indexes, also may require sequencing by sort keys that differ from the actual character strings in the records.

The demonstration of the feasibility of filing catalog entries by computer hinges on successfully performing two tasks: 1) analyzing the requirements of particular filing arrangements; and 2) programming the computer to perform the required operations. Actually, the two tasks are interdependent, because the nature of the filing analysis is strongly influenced by the ability of the computer to perform certain types of operations. The requirements for filing arrangement were considered at the genesis of the MARC project (1) and they materially affected the characteristics of the MARC II format (2,3). Structuring the format of a machine record is only part of the solution to the problem, however.

The first requirement for a program for library sorting is a set of generalized computer techniques for creating sort keys from MARC records at the user's option. These techniques will provide the foundation for further refinement of the sorting capability by developing algorithms to resolve specific problems in file arrangement. This article describes the characteristics, performance, and potential of a generalized program developed by the Information Systems Office and the Technical Processes Research Office of the Library of Congress.

The present approach to the computer sorting problem was based on the following assumptions:

1) The sort key must be generated on demand. For maximum flexibility and economy of storage, it should not be a permanent part of the machine readable record.
2) Data to be sorted must be processed (edited) for sorting by the machine. Input to a data field should be in the form required for cataloging purposes; it should not be contrived simply to satisfy the requirements of filing.
3) All data elements contributing to a sort key must be fully represented. To determine the position of an entry in a large file, the filing elements must be considered in turn until the discrimination point is reached. No element may be truncated to make room for another.
4) At least initially, a manufacturer's program should be used for sorting and merging the records with sort keys. Given the Library's present machine configuration, this means using IBM S/360 DOS tape sort/merge program.

These assumptions shaped the course that was followed.

The requirement that the sort key be generated on demand meant that a program had to be written to build sort keys specifically for records in the MARC II format. To allow maximum flexibility in specifying elements to be included in the sort key, the basic program was to be highly generalized, allowing any combination of fixed and variable field data to be included in the sort key. Since several data elements may have to be considered to determine the proper location of an item in a complex file, it is desirable to construct a single sort key containing as many characters of each element considered in turn as the length of the key will allow. Using a single sort key is more efficient than using separate keys for each element.

THE MARC II FORMAT

The MARC sort program was written to handle records in the processing format used by the Library of Congress. The differences between this format and the MARC II communications format (2,3,4) have been described by Avram and Droz (5). For the purposes of the present article, it is sufficient to give a brief outline of the structure of the format as it relates to computer sorting capability and to describe the salient features of the content designators that facilitate the creation of sort keys.

MARC records are undefined; that is, they vary in length, and information is not provided at the beginning of each record for use by software input/output macros. Since the manufacturer's program used for sorting MARC records cannot handle undefined records, preparation for sorting must include changing them from one type to the other. At the end of the sort/merge phase, they must be returned to an undefined state.

The maximum physical length of a MARC record is 2048 bytes. If a logical record (that is, the bibliographical data plus machine format data) requires more than 2048 bytes, it must be divided into two (or more) physical records. At present, the MARC sort program cannot handle continuation records of this type.

The basic structure of the format includes a leader, a record directory, and variable fields. Each variable field is identified by a unique tag in the directory. If necessary, the data in a field can be defined more precisely by indicators and subfield codes. They appear at the beginning of the field separated by delimiters. When no indicator is needed, the field begins with a delimiter. Tags, indicators, and subfield codes are used to specify what variable field data are to be included in the sort key, how the data are to be arranged, and what special treatment may be required. Although the full potential of these content designators has yet to be realized, they provide a basis for programming to achieve content-related filing arrangements; for example, placement of a single surname before other headings with the same character string.

CHARACTERISTICS OF THE MARC SORT PROGRAM

The MARC sort program has three components: 1) a sort-key edit program (SKED); 2) the sorting capability of the IBM S/360 DOS tape sort/merge program (TSRT); and 3) a merge routine written expressly for the MARC sort program.

The MARC sort program is activated by a set of control cards supplied by the user. These control cards specify the parameters to be observed in processing each record. Using this information, SKED reads each record, builds as many sort keys as are required to satisfy the parameters, duplicates the master record each time a different key is constructed, and records information about the sort key and the master record for possible later use. The output of SKED is an intermediate MARC sort file containing records with sort keys.

The second phase of the program involves TSRT, which also is controlled by parameter cards. The input is the intermediate MARC sort file. The TSRT program sorts the records according to their keys, using a standard collating sequence (that is, according to the order of the bit-configurations of the characters in the keys). The output can take either or both of two forms: 1) MARC format, in which the sort key is stripped

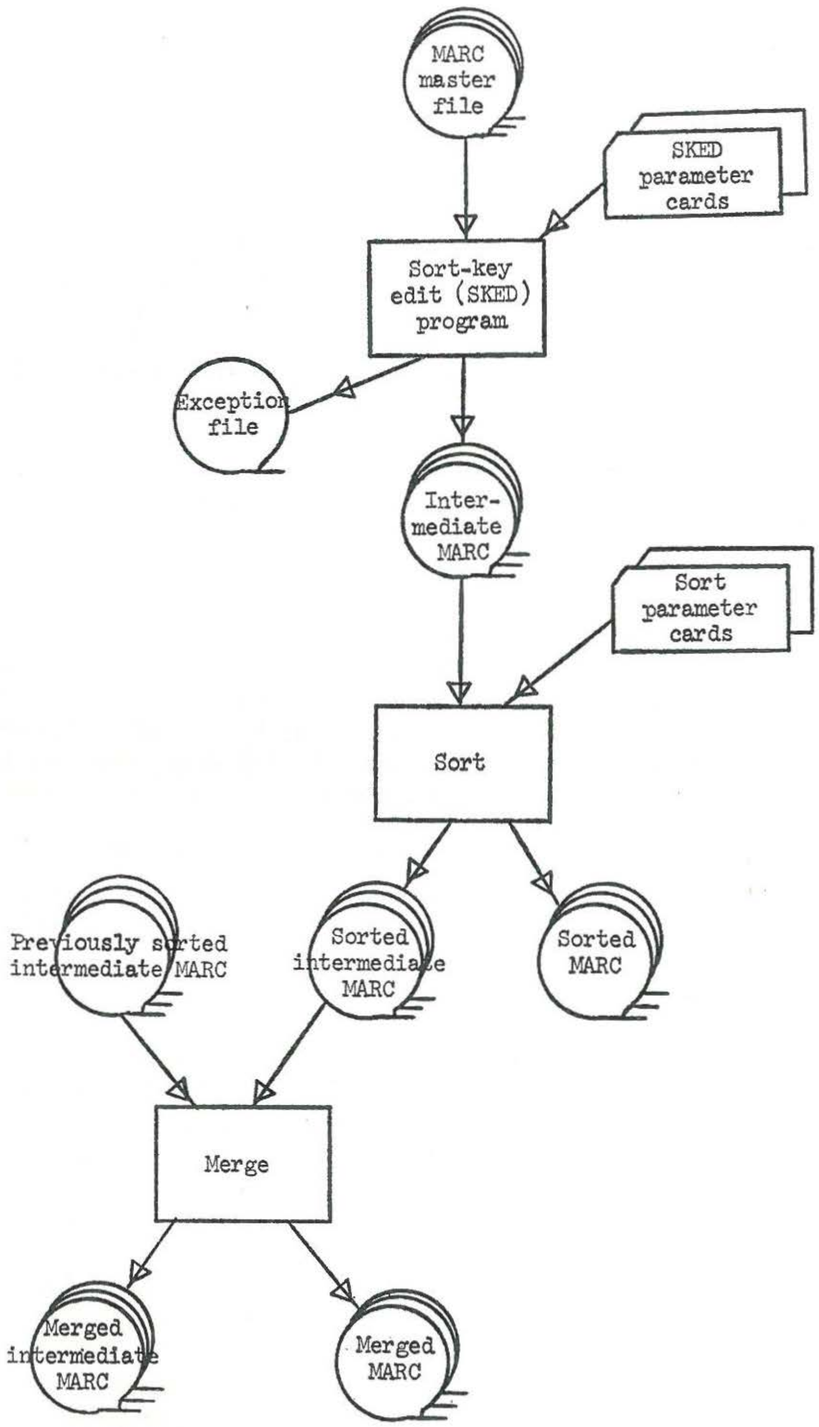

Fig. 1. MARC Sort Program Data Flow.

from each record and the format is returned to an undefined state; or 2) intermediate MARC sort format, which is identical with the input to the TSRT program (sort key remains with the record).

The merge routine written especially for the MARC sort program provides the capability to merge two or more files produced previously by TSRT in the intermediate MARC format and to output files in either or both of the above formats. It is necessary to provide a separate program for the merge function, since the manufacturer-supplied sort/merge package does not have the capability to merge intermediate MARC records while producing MARC II output. Figure 1 shows a simplified flow chart of the program.

Sort-Key Edit Program (SKED)

SKED builds sort keys according to the parameters specified at run time. In this process, it uses a table to translate the data to equalize upper- and lower-case letters, eliminate unwanted characters (e.g., diacritics, punctuation), and to provide a more desirable collating sequence during the sort phase. If the parameters result in more than one sort key for a record, the record is duplicated each time a new key is built.

The sort key is attached to the front of the MARC record when both are written in the intermediate MARC sort file. This is a variable-length, blocked file with a maximum block length of 4612 bytes (minimum blocking factor of 2). Figure 2 shows diagrammatically how a record looks after it has been processed by SKED.

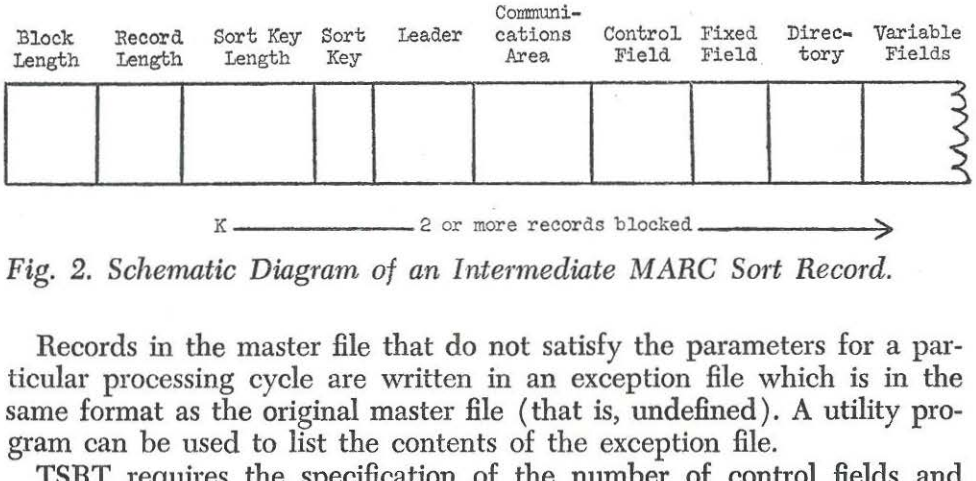

Fig. 2. Schematic Diagram of an Intermediate MARC Sort Record.

Records in the master file that do not satisfy the parameters for a particular processing cycle are written in an exception file which is in the same format as the original master file (that is, undefined). A utility program can be used to list the contents of the exception file.

TSRT requires the specification of the number of control fields and certain related information about each such field. As many as twelve control fields (each with a maximum length of 256 bytes) can be accommodated by the program. The current implementation of the MARC sort program uses a 256-byte key starting in position 9 of each record. (The first 8 bytes are used for variable record information). Any change in the length of the sort key must be reflected in the SKED source deck and on the control cards for TSRT. The specification of control fields shown on a TSRT control card must be changed as follows: If the length of the sort key is shortened, then the control field length specification must be reduced. If the sort key is lengthened, then the control field must be split into two or more control fields, as follows:

$$\text{Number of control fields} = \frac{\text{key length}}{256}$$

(If the quotient is a fraction, use the next higher integer.)

Parameters

The control cards for a SKED processing cycle allow the user to specify the following options:

1) Type of Field.
Both fixed and variable fields may be specified as parameters for a sort key. There is no restriction on the order in which they are given.

2) Specification of Fields.
Fields may be specified in several ways: a) exact form: a specific tag for a variable field (e.g., 650) or the address of a fixed field (the only option for this type of field); b) generalized form: NXX, NNX, XNN, NXN, where any digit may be substituted for N (e.g., 1XX); and c) as a range: NNN-XXX (e.g. 600-651).

3) Selection of Data from a Field.
The amount of data from a field to be processed can be determined in any of three ways:

a) Specifying the variable field tag without specifying particular subfield codes associated with it. This results in all data in the field being processed.
b) Specifying the number of characters to be processed. This must be done for fixed fields even if all data are desired. With either type of field, the data will be truncated if the number specified is smaller than the number of characters in the field.
c) Specifying the particular subfield codes associated with a variable field tag. This results in the sort key containing only the data from the specified subfields. For example, if the data in a 100 field were Smith, John, 1910- ed., failure to include subfield "e" (the designator for a relator like "ed.") in the specification of subfields would result in its being excluded from the sort key.

4) Alternate Selection.
Two or more parameters may be specified for the same position in the sort key with the provision that only the first to be found will be used. For example, if 240 (uniform title) and 245 (bibliographical title) are specified as alternate selections in that order and both occur in a record, preference is given to 240 and only it is used in the sort key.

5) Multiple Parametric Levels.
For efficient processing, mutually exclusively parameters can be listed in the instructions for the same processing cycle. The program affords a means of distinguishing between primary parameters that must always appear in the sort key and secondary parameters that cannot be combined with one another. The user also has the option of specifying that a sort key is to be generated using only the primary parameters. For example, if a book catalog were to contain main entries, added entries, and subjects, the tags for added entries and subjects would be specified as secondary parameters and the tags for the main entry, title, and imprint date as primary parameters. The sort key built for each added entry and each subject entry would always include the main entry (if present), title, and imprint date. This option can be by-passed if, for example, only a subject catalog is desired.

6) Sequence of Subfield Codes.
The subfield codes for a variable field may be specified not only to control the data to be included in the sort key but also to determine the order in which it appears. The following example shows how this works:

	Tag	*Subfield Codes*	*Data*
Record	100	abcd	Charles ‡ II ‡ King of Great Britain,‡ 1630-1685
SKED parameter	100	acbd	
Sort key			

```
CHARLES  KING  OF  GREAT  BRITAIN  II  1630  1685
```

7) Separator.
The user must specify the character that will separate each data element in the sort key but he has a choice of the character to be used. When the required characters from a given data element have been moved to the sort key, the selected separator is inserted to mark the end of the element. The separator is one of a set of specially contrived characters called superblanks that sort lower than any other character, including a blank. Use of the superblank permits the combination of different data elements in the same sort key because it prevents unlike data elements from sorting against one another, as shown below:

```
BALL JOHN● ARTHUR THE KING
```

```
BALL JOHN ARTHUR● CHESS
```

Without the superblank (shown here for convenience as a bullet) the second sort key would be placed before the first. Later it is expected that use of different superblanks within data elements will enable the sort/merge program to group related headings together.

8) Acceptance/Rejection Indicator.
At the beginning of the processing cycle, a decision can be made as to whether a MARC record should be processed if it does not include a particular parameter. If the rejection indicator is set, the record will be written to the exception file if that parameter does not occur. For example, if the parameters are 1XX (any main entry), 245 (bibliographical title), and imprint date (taken from the fixed field), a record without a main entry tag could be accepted but a record without a title rejected. This allows for title main entries while excluding imperfect records that may be present.

9) Duplication.
The parameters for variable fields to be included in the sort key may be satisfied by more than one combination of tags in the directory for a single MARC record. To provide for this occurrence, a duplication indicator may be set, thereby insuring that a sort key will be generated for each combination that satisfies the parameters. In addition the entire record will be duplicated so that it can accompany each sort key.

10) Number of Parameters.
The number of parameters designating data elements for the sort key is determined by the user at the beginning of the processing cycle. Any number up to twenty may be specified. It is unlikely that any given sort key will contain more than four or five parameters, but the ability to specify a much larger number allows for processing sort keys of several different types during the same cycle.

Translation Table

Before data characters are moved from a designated field in a MARC record to the sort key, they must be edited to insure that the key will include only characters that are relevant for sorting. This involves translation of the characters into the SKED character set: 1) to equate upper and lower case versions of the same alpha characters; 2) to translate the period, comma, and hyphen to the bit configuration of an ordinary blank; 3) to reduce other punctuation, diacritics, and special characters to a single bit configuration that cannot be moved to the sort key; and 4) to insure the proper machine collating sequence (blank, Ø - 9, A - Z). The SKED character set also provides bit configurations for superblanks (see above). The translation routine is written so that the character set can be changed without programming complications.

SKED also includes a feature that safeguards the sort key against the possibility of two consecutive blanks, as would be the case when a period and a blank occur in sequence, or when the data erroneously include two blanks when only one should occur. Before a character with a bit configuration equal to a blank is moved to the sort key, the program determines whether the last character moved has the same configuration. If it does, the second character is not moved.

Other Options

SKED has the optional capability of adding two variable data fields and their corresponding directory entries to each record. These entries follow the format for data in other MARC II variable fields.

1) 998 Entry.

When the SKED capability to duplicate records is used, it may be desirable to label one record of the set as the "master record". This technique or a modification of it might be used to generate a reference from a partial record to the full (master) record in a book catalog. When this option is selected there will be one, and only one, 998 field generated with each record. Information about the master record will be given by listing the tags used in the sort key to achieve a unique position of that record on file. For example, if a master record is sorted by main entry, then title (if different from main entry), and finally by the date of publication, the 998 field describing this master record should list the 1XX tag, followed by the 2XX tag, and finally by the address and length of the fixed field containing the publication date. The order of the tags in the 998 field is the same sequence used in the sort key of the master record.

2) 999 Entry.

When a book catalog is produced, it is desirable to show on the first line of an entry the element (e.g., title, subject) that determined the position of the entry in the arrangement. SKED supplies this information by creating a variable field (tagged 999) containing the initial element of the sort tag. If this option is chosen, an indicator can be set in the 999 field to show that the data in it should be printed as the first line of the bibliographic printout.

Sort/Merge

The TSRT program used by the MARC sort program is the standard IBM-supplied IBM System/360 basic operating system tape sort/merge program. Design specifications for this program satisfy the sorting and merging requirements of tape-oriented systems with at least 16K bytes of main storage. This program enables the user to sort files of random records, or merge multiple files of sequential records, into one sequential file. If any inherent sequencing exists within the input file, the program will take advantage of it. The intermediate MARC sort file produced by SKED is acceptable to TSRT.

As stated earlier, TSRT can accommodate up to twelve control fields for sorting. The MARC sort program requires only one control field at present. It is important to note that the TSRT comparison routines end as soon as a character in one control field is different from the corresponding character in another control field.

TSRT operates in four phases: assignment (Phase 0); internal-sort (Phase 1); external-sort (Phase 2); merge-only (Phase 3). If sorting is to be done, the assignment, internal-sort, and external-sort phases are executed. If only merging is to be done, the assignment and merge-only phases are executed.

TSRT provides various exits from the main line to enable a user to insert his own routines. Exit 22 in the external sort has been provided to delete records. In the MARC sort program phase this exit is used as an option for stripping the key and returning the records to standard MARC II format (undefined 2040 bytes maximum). The user exit intercepts each sorted record prior to output and converts it to an undefined state. The option is provided by addition of a "MODS" control card. A flow chart of TSRT appears as Figure 3.

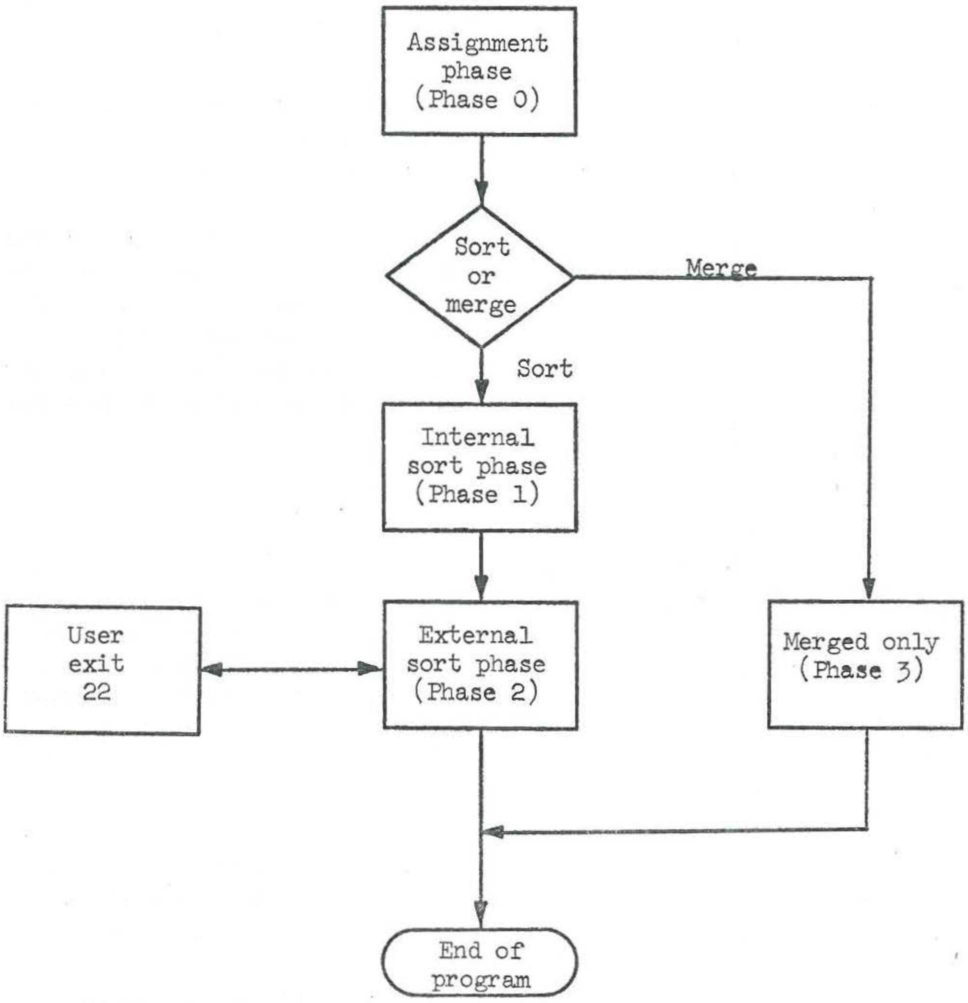
Assignment phase (Phase 0)
Sort or merge
Merge
Sort
Internal sort phase (Phase 1)
User exit 22
External sort phase (Phase 2)
Merged only (Phase 3)
End of program

Fig. 3. Flow Chart of the Sort/Merge Phases of the MARC Sort Program.

Four work tapes are used by this application of TSRT. A fifth drive is used for input and output.

The IBM sort package is not capable of writing undefined length records. Since the MARC record is in an undefined format, the output routines of TSRT cannot be used. Therefore, the following method is used to develop the MARC output file: 1) a separate file receives the sort output instead of the standard sort out-file; 2) the separate file is written by special coding in Exit 22 of TSRT; and 3) each record is written in such a way as to prevent the sort from also writing the record. The merge routine written especially for the MARC sort program will output either intermediate (with key) or MARC II (without key) format tapes or both.

THE PROGRAM IN ACTION

Processing Times

SKED was written in assembler language, using physical input/output control system and dynamic buffering assignments to achieve speed. The amount of time required to process a particular record is affected by the applicability of run-time parameters on the record. For example, if the user specifies twelve data elements from twelve different variable fields for inclusion in the sort key, the processing time will be greater than that required for a run with only one specification. Likewise, a run requesting duplication of each record for every added entry will require more computer time than another run that does not duplicate records at all.

Since the total processing time required for a file of *n* records will be the same as the time to process *n* files of one record each (disregarding I/O considerations) it is possible to project times using a run with various data conditions and numerous SKED parameters as a guide.

One such run on the IBM S/360 at the Library of Congress processed records at the rate of 2,400 per minute. Twelve parameters were specified and records were duplicated for certain subject entries. Except for time spent in changing tape reels, SKED can be expected to process records at the same rate regardless of the size of the file.

The processing time for TSRT is affected by the same characteristics that affect most sort programs. Some of these are as follows: 1) amount of memory available to the sort; 2) number of storage units (in LC's case, tape units are used); 3) types of storage unit (for magnetic tape—inter-record gap, density, and tape length); 4) block size for data; and 5) amount of bias in the input.

The only characteristic of SKED that seems to relate to the speed with which TSRT operates has to do with SKED's extended use of a single control field. For example, in many sorting systems, if records are to be arranged by main entry and within main entry by title and then by date, three control fields would be specified. One would be chosen for main entry; one would be chosen for title; and, one would be selected for the date. SKED places all of these within the same control field, separating them by a superblank. Since TSRT is required to discriminate only on the single control field, a smaller amount of processing time is needed than would be the case if several control fields were used.

Results

Although SKED does not have the ability to make the refined distinctions among headings required for sophisticated filing arrangements, it performs in a workmanlike way in producing alphabetical sequences that are unaffected by the presence of diacritical marks and vagaries in punctuation and spacing. Moreover, the collating sequence (blank, Ø - 9, A - Z) insures that short names will file before longer ones beginning with the same character string. The ability to truncate headings to remove relators (e.g., ed.) also insures the creation of a single sequence for authors whose names are sometimes qualified in this way.

The following consolidated example shows some of the arrangements produced by SKED. To simplify the presentation, generally only the first filing element is given. Other elements have been added if they are needed to show distinctions that were made by the program.

Abbott, Charles
ABC Company.
A'Beckett, Gilbert
Acadia University.
. . .
Alexander III, King of Albania
Alexander II, King of Bulgaria
Alexander I, King of Russia
. . .
Bradley, Bradley and Bradley, firm.
Bradley, Milton, 1836-1911.
Bradley (Milton) Company
. . .
Katz, Eric, ed.
  Sound about our ears.
Katz, Eric.
  Sound in space.
. . .
Lincoln, Abraham, Pres. U. S., 1809-1865.
Lincoln Co., Or.—Directories
Lincoln County coast directory
Lincoln, David.
Lincoln Highway
Lincoln, Marshall.
Lincoln, Mass.—History
Lincoln, Me.—Genealogy

London, Albert, joint author.
  [Mockridge, Norman.] (author not used in sort key)
  Inside the law.
London, Albert.
  London at night.
London at night.
London. Central Criminal Court.
London. County Council.
London, Declaration of, 1909.
London—Description
London (Diocese) Courts.
London, Jack.
  White fang. 1930.
London, Jack.
  White fang. 1950.
London, Jack White.
  Alaskan adventure.
London, Ont. Council
London, Ontario; a history
London. Ordinances.
London—Social conditions
. . .
Smith, John, 1900-
Smith, John, 1901-1965.
Smith, John Allan, 1900-
Smith, John, clockmaker.

ANTICIPATED DEVELOPMENTS

At the present stage of the development of the MARC sort program, SKED does not have the ability to treat data in a field according to their semantic content. It cannot, for example, treat a character string in a 100 field in a special way because it is a single surname as opposed to a forename or multiple surname. Nor does SKED include routines for treating abbreviations and digits as if spelled out, or for suppressing data in a given field in some cases but not in others.

The achievement of these capabilities will require: 1) development of a generalized technique for taking account of indicators in processing data in variable fields; 2) devising algorithms to handle particular filing situations related to the content of the field; and 3) placement of the algorithms within the framework of the SKED program.

The refinement of SKED is being undertaken in relation to the problem of maintaining the LC subject heading list in machine readable form. Techniques developed for this purpose will be applicable also to filing for book catalogs and other listings. The result should be a firm foundation for a comprehensive program for arranging bibliographic entries by computer.

AVAILABILITY OF THE PROGRAM

Since the MARC sort program should be useful to libraries that subscribe to the MARC Distribution Service, the package (consisting of SKED and the modified version of TSRT) has been filed with the IBM Program Information Department. Requests should be made through a local branch office of IBM and should cite the following number: 360D — 06.1.005.

REFERENCES

1. U. S. Library of Congress. Office of the Information Systems Specialist: *A Proposed Format for a Standardized Machine-Readable Record.* Prepared by Henriette D. Avram, Ruth S. Freitag, Kay D. Guiles. ISS Planning Memorandum, No. 3. (Washington, D. C.: 1965), p. 5.
2. U. S. Library of Congress. Information Systems Office. *The MARC II Formats A Communications Format for Bibliographic Data.* Prepared by Henriette D. Avram, John F. Knapp, and Lucia J. Rather. (Washington, D. C.: 1968), p. 33.
3. "Preliminary Guidelines for the Library of Congress, National Library of Medicine, and National Agricultural Library Implementation of the Proposed American Standard for a Format for Bibliographic Information Interchange on Magnetic Tape as Applied to Records Representing Monographic Materials in Textual Printed Form (Books)," *Journal of Library Automation,* 2 (June 1969), 68-83.
4. U. S. Library of Congress, Information Systems Office: *Subscriber's Guide to the MARC Distribution Service.* (Washington, D. C.: 1968).
5. Avram, Henriette D.; Droz, Julius R.: "MARC II and COBOL," *Journal of Library Automation,* 1 (December 1968), 261-272.