LC/MARC ON MOLDS; AN EXPERIMENT IN COMPUTER-BASED, INTERACTIVE BIBLIOGRAPHIC STORAGE, SEARCH, RETRIEVAL, AND PROCESSING

Pauline ATHERTON, Associate Professor, School of Library Science, and Karen B. MILLER, Research Associate, Syracuse University, Syracuse, New York

A project at Syracuse University utilizing MOLDS, a generalized computer-based interactive retrieval program, with a portion of the Library of Congress MARC Pilot Project tapes as a data base. The system, written in FORTRAN, was used in both a batch and an on-line mode. It formed part of a computer laboratory for library science students during 1968-1969. This report describes the system and its components and points out its advantages and disadvantages.

INTRODUCTION

The somewhat intimidating title of this report becomes less so when translated from jargon into more familiar phrases. The LC/MARC ON MOLDS experimental project conducted at Syracuse University School of Library Science utilizes a computer: 1) to store bibliographic reference (library catalog) data, 2) to search the data for items that meet a searcher's criteria, 3) to retrieve items the searcher wishes retrieved, and 4) to process or manipulate items as required. A dialog or interaction between man and his data, via the machine, is established when a searcher makes a request in a query language and the computer responds immediately to the request.

The LC/MARC ON MOLDS system consists of two major components. The first is the data base, which is a slightly modified subset of the Library of Congress MARC Pilot Project records (1).

The second component is the computer programming system written in FORTRAN known as MOLDS (acronym for Management On-Line

Data System). MOLDS provides the computer routines required to store and maintain the data base, and the query language (also known generally as MOLDS) that a searcher uses to interact with his data stored in the computer.

The LC/MARC ON MOLDS system was originally implemented in April 1968 on the IBM 360/50 at the Syracuse University Computing Center. This system is part of an experiment to determine how on-line interactive retrieval systems could be used to greatest advantage in the information gathering process. The MOLDS system, developed in 1966 by the Syracuse University Research Corporation (2) for management purposes, was readily available for use in the research reported in this paper. MOLDS has been used with several data bases, including the MARC records.

The system has not been made available to a large user population. Preliminary work with the system and a few demonstrations to students have already provided considerable insight into the desirable and undesirable features in both the MARC data base and the MOLDS query language, an insight that has already resulted in both data-base and query-language modification.

Work with the system on the computer at Syracuse University has raised many crucial questions extending beyond the original research plan about system and data base design—questions for which there are as yet no answers. Even at its early stage of experimentation the work should be of interest to librarians because of its use of the MARC Pilot Project records and its use of an available retrieval program with features suitable for reference retrieval.

To the authors' knowledge, this is the first computer-based project in which the Library of Congress MARC records were used in an interactive retrieval environment.

The query language (MOLDS) was not specifically designed for reference retrieval, but its design features make its use for this purpose quite feasible. It differs from the usual interactive system designed for bibliographic reference retrieval and therefore deserves attention for comparative purposes. MOLDS gives a user the ability to process as well as retrieve data, something very few search and retrieval systems are designed to do.

The contribution of LC/MARC ON MOLDS to the world of information retrieval, promising though it appears, cannot be assessed until all experiments are run. This report on its features, both good and bad, is offered in order to make those concerned with the design and application of interactive systems aware of its unique aspects and potential. Hopefully, this work will contribute another ingredient to the synthesis of ideas and methods that will bring the state of the art ever closer to the optimum and ideal.

Table I. Some Features of Interactive Retrieval Systems (circa 1968)

System Name	*# Docs. in Data Base*	*Data Base Structure*	*Access Points*
1. AUDACIOUS (AIP)	2330	Tree structure threaded list	UDC descriptions Euratom key words
2. BOLD (SDC)	6000	Threaded list	ASTIA Subject category index terms accession numbers
3. COLEX MICRO (SDC)	2000	Inverted index tree structure index	descriptor, author, subject } qualified by { subject, country, type document, subject area, date
4. GRINS (Lehigh U) >	1000	Serial document inverted index	index terms
5. MULTILIST	Varies	Threaded list Tree structure directory	any chosen key term to fit application (eg. author, subject, date, title words, subject headings)
6. MARC/ MOLDS	2000	Cell-matrix	any discrete data block
7. NASA/RECON	270,000	?	subject, author, corporate source, report #, contract # } qualified by { date
8. TIP (MIT) >	25,000	List structure	author(s) location(where work done) citation identification (i.v-p.) article title (entire, keyword) citation index bibliographic coupling
9. SUNY BIOMED COMM. NETWORK >	20,000	Inverted index	author, title, subject } qualified by { date, lang.

*Each command is a subroutine. Commands are tailored to application.

Access to Authority Files On-Line	*Related Terms or Cross Refs Given*	*# Commands In Query Language*	*Computer Instruction In Language Use*	*Computer Aided Query Formulation (Conversation)*	*Root Word Search*	*Communication Link*
UDC Schedules	Yes	11	Optional	Limited	Yes	CRT
Subject category list, index term file	Yes	14 (11 light pen)	Optional	Yes	Yes	CRT with light pen
No	No	(conversation)	Optional	Yes	Yes	Teletype
Index term	Yes	(conversation)	No	Yes	Yes	Teletype
No	No	*	No	No		
Optional	Optional	35	No	No	No	CRT
No	Yes	16 function keys	Optional	Yes	No	CRT
No	No	9 Also various MAC commands	No	No	Yes	Teletype
No	No	10 (?)	No	Yes	No	IBM 2740 console

BACKGROUND

A number of interactive retrieval systems have been designed and implemented within the last few years. The features and potential of LC/MARC ON MOLDS are best viewed in relation to what has been done in the field up to now. To gain some perspective, the major features of data base structures and query languages of other interactive systems are summarized in Table 1. This table presents those features of most interest to librarians who may wish to compare searching on a computer with searching in the card catalog or other bibliographic reference tools. References 3-12 document sources for the data in this table.

MOLDS DATA BASE STRUCTURE

The general structure of the data base with which MOLDS operates is, in comparison with the threaded lists and inverted indexes found in many retrieval systems, extremely simple and unsophisticated. The data base can be composed of from one to ten distinct files of 1000 records each. A record is equal to the bibliographic description on a card in a library catalog. Each record may be up to 300 computer words (1200 characters) long and may be subdivided into 80 blocks. Originally, there was a 200-word (800-character) limitation on record size, but this has now been expanded. The total file size (limit of 10,000 records) is adequate for testing purposes, but expansion beyond the present limitations is planned in order to make the system more practical for actual use.

The structure of a file is essentially a simple matrix. Each row contains all the elements of a single complete record; each column contains all like discrete items of all the records in the file. The columns are called blocks in the MOLDS system, *block* and *field* being used synonymously in this report. For example, a library catalog card for one publication would be a record in a file composed of library catalog records. The main entries in the file constitute a block and the dates of publication constitute another block. Figure 1 illustrates the data base structure, as of 1968. In this illustration the maximum number of files is 10 (1000 records each) and the maximum number of blocks 80.

Each file and each block in a file is given a name and/or number. A user can reference or call up any file or data block within a file by using its name or number in a MOLDS query language command. There are as many access points to a file as there are blocks in that file. This is in contrast to a conventional card catalog, for example, where the only access points are filing entries: main entry, title, subject(s), added entries, series, and analytics.

No specific provision is made within the MOLDS system for the storage of authority files, cross reference lists, or other intermediate keys to the records. Such files are not absolutely necessary for effective operation of the system since every block can be accessed and can serve as its own authority file. For more efficient system operation, however, it is intended

to explore the possibility of creating authority files as part of the data base, beginning with portions of the Seventh Edition of the Library of Congress *List of Subject Headings.*

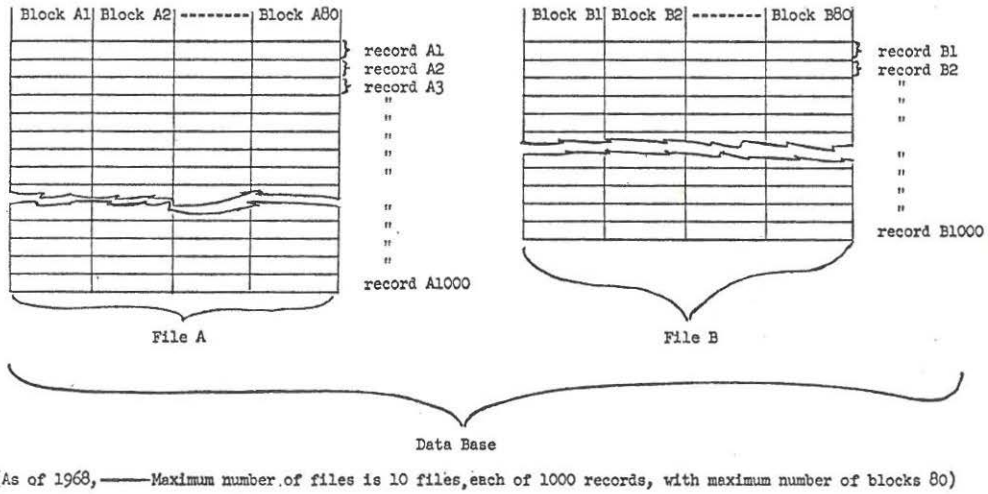

Figure 1. Section of General MOLDS Data Base Structure.

Provision is made for temporary user storage areas in which the user places the results of his retrieval and processing operations. Data in the user area is retained only during the session in which it is created. Although it cannot be saved for use at a later date, all or part of it can be printed out on the on-line printer for the user's later reference.

While the general structure of the data base is formalized within the MOLDS system, the content and specific organization of a particular data base is determined by its originator. This feature, plus the simplicity of MOLDS' own structure, introduces a great deal of flexibility into the data base and the use that can be made of it. The originator of the data base may designate as a block any discrete data item he wishes. If the user population is dissatisfied with results using one content and arrangement of blocks, the base can be reformatted and restructured in a fairly simple maintenance run. No problems of linking records or modifying authority lists arise, as neither is part of the system. The first version of the LC/MARC data base has in fact been modified by addition of three blocks and division of one block in half to form two blocks, giving access to smaller units of data.

The LC/MARC Data Base in MOLDS Format

Library of Congress MARC Pilot Project tapes containing some 40,000 records of English language books cataloged in 1966-67 became available for this project in the Fall of 1967. Because of the MOLDS data base limitations, a subset of these catalog records was selected for use with MOLDS. The original plan was to have each file in the data base consist of as complete a set as possible of all MARC Pilot Project records from a single Library of Congress classification schedule. The candidate for the first file was class *R* (Medicine) which contained just under 1000 records. Later MOLDS files were formed for two other LC classes: *T* (Technology) and Z (Bibliography and Library Science). In mid-1969 two stratified sample files of the MARC data base were created, one in the humanities, another in the social sciences. In all, Syracuse has a MARC/MOLDS data base of 10,000 records.

The record format of the MARC tape was first analyzed to determine which fields should be included in the data base, and which might be omitted. The criterion for selection was probable usefulness to searchers of the data base, a conception that should undoubtedly be modified as searches are monitored. Appropriate changes would not be difficult.

Toward the end of January 1969, a programming project was begun which entailed the design and implementation of a computer program to perform format conversion of the Library of Congress MARC I bibliographic file to satisfy MOLDS data base requirements. The project represented a three man-month effort and was completed by June 1969.

The data-base converter program represents an attempt to provide a user-oriented facility for creating a MOLDS data base from MARC information. Essentially, the user of the program describes each MOLDS file to be produced by specifying:

1) the number of (fixed) fields per MOLDS record;
2) the name and size (in characters) of each field in the MOLDS record;
3) the name of the MARC I field from which the data are to be taken;
4) selection criteria according to which MARC I records are to be chosen for conversion;
5) for any MARC I field, a data conversion procedure to be applied prior to transferring the information to the appropriate MOLDS field;
6) whether or not diacritical codes should be stripped from the MARC I field prior to transferring the information to the MOLDS field;
7) whether or not character translation from lower-case to upper-case codes should be performed on the data prior to transfer from the MARC I to the MOLDS field.

Although the program has not yet been refined to the extent originally intended, nevertheless it contains all the features indicated above and has

been used to create ten MOLDS files since its completion. The program is written in PL/I and more fully documented in a report available from the National Auxiliary Publication Service of ASIS.

MOLDS requires fixed-field input for its data base, but many of the fields or data blocks on the MARC tape are variable in length. Therefore, the field lengths of 200 records in the class *R* (Medicine) subset were examined to determine the maximum size which would produce a MOLDS record within the original 200 computer-word (800-character) limitation and still retain all the desired data. This limitation was easily expanded to 300 words, allowing addition of new fields and expansion of existing fields as new MARC/MOLDS files were generated. A record whose original variable length was 500 characters or less expanded to about 800 characters when converted to fixed-field form. In the first data base only records of 500 characters or less were considered for inclusion, which gave a total of 620 records in the first MARC/MOLDS file. By mid-1969 this data base was greatly enlarged using the program described above.

The names of the present MARC/MOLDS files are: SS01, SS02, SS03, SS04, SS0Z, and SS0H. The first files generated were called MARC and MARZ.

The MARC/MOLDS format now in use is given in Table 2. The additions made to the original format are noted. MARC/MOLDS block names can be used instead of block numbers; for ease of searching both name and number are given in the table. The MOLDS block number corresponds to MARC Pilot Project field tags whenever possible. After this second revision had been completed, MARC II (13) format with new field tags appeared. Interestingly, there were remarkably few differences.

Creating an information retrieval system from other data bases can present some major headaches. During the first test session with the MARC/MOLDS data base, it was discouraging to find that successful retrieval operations could not be performed on such vital items as subject or main entry (blocks MAIN and SUBA, respectively). The problem lay in the fact that the lower-case character codes employed on the MARC tape had not been converted to the all-upper-case-codes required by MOLDS. Once discovered, the problem was easily remedied. Other problems were not so easy to solve.

The MARC data base had been received in a "raw form", i.e., there were typographical errors in the original tapes and irregular spacing; and incorrect punctuation, spelling and abbreviations. There was no way to detect these errors, and the retrieval program would only work on direct matches of query and document information elements. The MOLDS language (to be discussed subsequently) required a good deal of standardization and regularity of the records to take full and effective advantage of its retrieval capabilities.

Table 2. MARC/MOLDS Data Base Format

Description MARC Fixed Fields:	*Field Names Molds Block Name*	*Blk No.*	*Chars. In Block*	*MARC I Fixed Field Position or Tag No.*	*Data Element Information Values or Explanation*
LC card no.	LDNØ	80	11	9-19	
Type of main entry	TYPE	81	1	21	A-G
Form of work	FØRM	82	1	22	M|S
Bibliographies indicator	BIB	83	1	23	X|b
Illustrations "	ILLU	84	1	24	"
Maps "	MAP	85	1	25	"
Conferences "	CØNF	86	1	26	"
Juvenile "	JUV	87	1	27	"
Languages	LANG	88	4/4	29-36	Both languages
Language 1	LAN1	1	4	29-32	
Language 2	LAN2	2	4	33-36	
Publication dates	DATE	89	4/4	38-45	Both dates
Height in cm.	HITE	90	2	59-60	
Uniform tracing indicator	UNIF	91	1	66	X|b
Series tracing indicator	SERT	92	1	69	"
Place of publication code	PLCD	18	4	46-49	
Publisher code	PUCD	19	4	50-53	
LC call no.	LCNØ	98	20	90	
Dewey class no.	DEW1	99	20	92	
Dewey class no. (edited)	DEW2	39	8	92	ooDDD.DD e.g. 00351.2352
LC class no. (edited)	LCCL	97	8	90	
Main Entry	MAIN	10	68	10	
Title Statement	TITL	20	80	20	
Subtitle Statement	STIT	21	80	20	
Edition Statement	EDIT	25	12	25	
Place } imprint statement	PLCE	30	28	30	
Publisher } imprint statement	PUBL	31	28	30	
Collation	CØLL	40	48	40	
Series note	SERS	50	44	50/51	
Note	NØTA	60	44	60	
Note	NØTB	61	44	60	
Subject tracing	SUBA	68	48	70	
Subject tracing	SUBB	69	48	70	
Subject tracing	SUBC	70	48	70	

Personal Author Tracing	PAUA	71	40	71
Personal Author Tracing	PAUB	72	40	71
Corporate Author Tracing	CØRP	73	1	72
LC card suffix	LCFF	94	3	94
Total MARC/MOLDS Characters			848	

THE MOLDS SYSTEM

Functionally, the MOLDS system consists of utility routines to store a data base, a well-defined query language, a language interpreter, and a set of logical procedures which allow the user to operate on a data base.

The MOLDS system is a set of FORTRAN IV subroutines which perform the maintenance functions, interpret the commands in the query language and perform the desired logical procedures.

The subroutines render the system modular and open. It is therefore relatively easy for a programmer skilled in FORTRAN IV to add, modify and delete commands and functions as required. This feature of the system is quite desirable. User feedback invariably points up weaknesses in the language or suggests useful features which might be incorporated. MOLDS was continually modified in response to user requirements, and each modification was implemented within a short time without requiring major programming changes throughout the system. The system has already grown since it was first implemented with the MARC data base, and commands have been added or modified as required.

Hardware Configuration

MARC/MOLDS was run at Syracuse University Computing Center on an IBM 360/50 computer. Originally, the on-line mode required full dedication of the computer during execution. The MOLDS system requires some 150,000 bytes of main memory and a disk storage unit to hold the entire data base, as well as intermediate data generated by the user. The MOLDS system has been implemented on other computers (2).

Interaction with the system in the on-line version was carried on through an IBM 2260 Display Station consisting of a keyboard and CRT (cathode ray tube) display screen. Although two or more consoles have not as yet been operated simultaneously, the system is intended to be time-shared.

Effort was made to alter the system to operate in a 50,000 (50K) upper partition, so that it could be accessible at all times rather than on a scheduled basis. This involved reorganizing the program into an overlay structure in which the basic or root segments are resident in a fixed portion of memory throughout execution, while the remainder of the program is divided into a set of smaller segments which can overlay each other, being brought into memory only when needed. This task

required a careful analysis of each subroutine for its dependence upon others, breaking the program into mutually exclusive segments, while ensuring that any given set of segments which occupied memory simultaneously did not exceed 50K bytes of storage. Many of the larger segments which had to be further subdivided required considerable reprogramming.

The first attempt at executing the new overlay version failed. Due to a general lack of experience with the 2260 Display Units, it had not been anticipated that system software would not allow the console to be accessed from outside of the root segment, and the 2260 software package had been placed in an overlay area. As a result the original overlay configuration had to be altered. The console input/ouput (I/0) package was moved into the root segment, increasing its size by several hundred bytes and similarly decreasing the amount of storage available for the overlay portions. Therefore, it was necessary to develop yet another configuration to conform to these new storage limitations.

While the necessary changes were being made, the Computing Center began operating a limited time-sharing system which itself required full dedication of the 360/50 machine. Projected dates for returning to normal computer operations within a multi-partition environment were far enough in the future to suggest the efficacy of creating a new version of MOLDS which could function off line, with cards and printer instead of the 2260 consoles.

In this batch, or off-line, mode MOLDS jobs could be submitted through the regular queue and run by Computer Center staff during batch processing time. With the on-line source program as a starting point, all references to 2260's were replaced with card reader and printer statements and the MOLDS language instructions deleted which depended on the console for their use. After all changes had been made and compilation was completed successfully, the off-line MOLDS was exercised against a sample data base until it was satisfactorily debugged.

Since it was known that the Computing Center would eventually return to partitioned operation, it was next undertaken to overlay the off-line MOLDS into a 50K partition. This was accomplished with little difficulty since the problems encountered in working with the on-line version were largely due to the consoles. The end result of the entire task, therefore, was an off-line MOLDS which could operate either in core or in overlay structure at the discretion of the user.

THE MOLDS QUERY LANGUAGE

The MOLDS query language includes some 34 distinct commands which must be entirely formulated by the user according to precise syntactical rules. The large number of commands is in part a reflection of the fact that this system provides the user with the ability to perform more operations of a greater variety on a data base than other interactive infor-

mation retrieval systems. It provides for retrieval of records from the data base according to data value descriptors, processing of data values by arithmetic and logical operations, sorting of retrieval records, and display of retrieval records in full or in part.

Operationally, the MOLDS system regards a file of records as a set of parallel lists of blocks (Figure 1). With the MARC data base, these blocks were the 38 fields of catalog data (such as Dewey class number, title, author, etc.). The commands in the MOLDS query language are geared to list processing operations. In general, most of the MOLDS commands will result in the formation of lists which are either identical in format to the original file, or are an independent list of alpha or numeric constants not subdivided into blocks.

Despite its surface complexity, the query language was designed specifically for users with absolutely no computer experience. The fixed format commands are easy to learn and use, even for the novice in computer based systems. They are mnemonic enough so that a little use soon brings an easy familiarity with them.

Commands in the MOLDS Query Language

There are six categories of commands in the language: retrieval, processing, display, storage, utility, and language augmentation. The commands are listed below with a brief explanation of each.

Retrieval Commands:

FIND:	Forms a temporary subfile consisting of records from the data base for which the value in a specified block is equal, not equal, greater, greater or equal, less, less or equal to an input value.
EXTRACT	Forms a temporary subfile consisting of records from an argument subfile for which the value in a specified block is equal, not equal, greater, greater or equal, less, less or equal to an input value.
FETCH	Forms a temporary file which duplicates an existing file in the data base (added to original MOLDS commands during this project).
DEFINE	Forms a temporary subfile from two argument subfiles based on logical relationships AND, OR, NOT.
CHAIN	Forms a temporary subfile consisting of records from an argument subfile for which the value in a specified block is equal to any of the values in a specified block from a second argument subfile.
SELECT	Forms a temporary subfile consisting of records from an argument subfile for which the value in a specified block is equal to any of the values in an argument list.

These six retrieval commands allow the user to extract selected data from the data base. Selection is based on 1) a simple algebraic relationship (e.g., equal, not equal, greater than, etc.) between block values and a value specified by the user in the command (*value* may be alphanumeric or numeric), or 2) a simple logical relationship (e.g., and, or, not) between block values in two lists.

All retrievals from MOLDS files are based on exact-match correspondences between input descriptors and data values as they occur in records. Each file is treated as distinct regardless of the fact that for the MARC/MOLDS data base the second file may simply be a continuation of the first, etc.

Any block in a file may be used as an argument in a retrieval process. Thus, the usual range of access points (author, title, subject, classification number) is considerably extended to include such unorthodox access points as juvenile literature, language, illustrations, and bibliographies. For example, one can retrieve all documents on a given subject or subjects which are juvenile books with bibliographies and illustrations published by a given publisher in 1966. The user can define his search limits with a degree of specificity not found in most interactive systems. However, the price he must pay is exactness in specifying the values used as retrieval criteria.

The system will not retrieve on root words or key letter combinations, although such capability could be added. The block values must, therefore, be consistent and the user must have a precise knowledge of what they may be. This knowledge can be gained by examining the values and having them printed out as needed. (MOLDS does have the capability of selecting unique values from a list, ordering them, and printing them out at any time during system operation.

Processing Commands:

COUNT	Counts the number of records in an argument subfile or items in an argument list.
ORDER (REVERSE)	Arranges the records of an argument subfile in ascending (descending) order according to the values in a specified block or similarly sorts the values in an argument list. May be applied to alphabetic, numeric, and chronological data.
MAXIMUM (MINIMUM)	Selects the record containing the maximum (minimum) value in a specified block from an argument subfile, or the maximum (minimum) value in an argument list. May be applied to numeric or chronological data.
TOTAL	Calculates the sum of the values in a specified block of an argument subfile or of a list of numbers.
AVERAGE	Calculates the average of the values in a specified block of an argument subfile or of a list of numbers.

MEDIAN	Calculates the median of the values in a specified block of an argument subfile or of a list of numbers.
VARIANCE	Calculates the variance (standard deviation squared) of the values in a specified block of an argument subfile or of a list of numbers.
SQUAREROOT	Calculates the square root of each value in a block of an argument subfile or of a list of numbers.
DIFFERENCE	Calculates successive differences in the values of a specified block in an argument subfile or of a list of numbers.
ADD (SUBTRACT MULTIPLY DIVIDE)	Adds (subtracts, multiplies, divides) the values from a specified block from an argument file (or list) to the corresponding values from a specified block from a second argument file (or list).
FIRSTELEMENT	Selects the first record from an argument subfile or list.
REDUCE	Deletes the first record from an argument subfile or list.
COMPRESS	Forms a temporary list composed of all the unique values in a specified block of an argument subfile or in an argument list.

The eighteen Processing commands allow the user to manipulate the data in the lists he has retrieved. He may count the number of elements in a list, arrange them in ascending or descending order, form the sum, average, variance, median and square root of a list of numbers; add, subtract, multiply, and divide one list by another, and select all unique elements from a list.

The ability to process data as well as retrieve it may be unique to MOLDS as compared to other interactive systems, and gives the language a useful added power.

Display Commands

DISPLAY	Outputs on the CRT (cathode ray tube) each complete record in an argument subfile (Added to original MOLDS commands during this project).
SHOW	Outputs in columnar fashion on the CRT selected blocks from up to three argument subfiles or lists (Deleted in batch or off-line mode).
PRINT	Outputs in columnar fashion on the printer selected blocks from up to three argument subfiles or lists (Added to original MOLDS commands during this project).

The three Display commands allow the user to display entire documents, or display selected books of information or records in columnar format. In

the on-line version of MOLDS this may be done on the CRT, or a printout made of selected blocks or lists of documents on the high speed printer. There is much flexibility and versatility in output format which is completely determined by the user. The command, SHOW, is not used in the batch mode of MOLDS.

Storage Commands:

SET	Stores a single numeric value.
STORE	Stores an alphabetic, chronological, or numeric list of arbitrary length.

The two Storage commands allow the user to insert independent lists of constants into the storage area. Such lists do not become part of the data base, but are used in conjunction with retrieval and processing commands.

Utility Commands:

CLEAR	Deletes from storage a temporary subfile or list created during the session.
DELETE	Deletes from storage all temporary subfiles or lists created during the session.
DUMP	Displays on the CRT in tabular fashion the names, file origins, and number of items in each subfile and list created by the user during the session (deleted in batch or off-line mode).
RECALL	Displays on the CRT the command which resulted in the creation of a specified temporary subfile or list (added to original MOLDS commands during this project).
LIST	Produces printed copy of all commands issued during the session. May be used with STOP at end of search (added to original MOLDS commands during this project).

The five Utility commands allow the user to perform housekeeping operations, such as the clearing of storage areas, reinitialization of the system, and termination of execution. The command DUMP is not used in the batch mode of MOLDS.

Language Augmentation Command:

PROGRAM	Allows the user to create new commands consisting of a sequence of basic commands and to store them for future sessions.

The language augmentation command PROGRAM, is one of the most important features of the language. It allows the user to create new commands tailormade to his own needs. This is shown in the first MOLDS search query which follows.

SEARCH REQUEST FORMULATION IN MARC/MOLDS

MOLDS Search Query—Example 1 (Batch Mode)

```
PROGRAM  TALLY  A/
COUNT  B  A/
PRINT  B//
END

FIND  ZNY  SSOZ/PLCD/E/NYNY/
TALLY  ZNY/
PRINT  ZNY/PLCE/PLCD/PUCD//
FIND  P67  SSOZ/DATE/E/1967/
TALLY  P67/
DEFINE  NY67  ZNY/AND/P67/
TALLY  NY67/
AVERAGE  AVHT  NY67/HITE/
PRINT  AVHT/
STOP
```

The above example shows an off-line or batch-mode search. This sequence of commands would be keypunched and submitted as a job deck in the regular queue and run by the computer center staff, the searcher receiving the results as a printout from the high speed printer. SSOZ is the name of one of the MARC/MOLDS files. This particular interaction shows the use of the operator PROGRAM to augment the language in the subsequent search by adding TALLY to the list of commands.

The following example shows a search query which is a sequence of some typical MOLDS commands along with an explanation of the effect of each. Each command has three parts. The first part (FIND, DEFINE, etc.) is the imperative which tells what operation is to be performed. The second part (BIBL, ENGL, BOTH, etc.) is the label of the place in storage where the result of the operation is to be stored. This label is made up by the user when he gives a command. The third part of the command is the operand. In some cases the operand gives the criteria for retrieval (as in FIND, DEFINE). It always gives the name or label of the list to be operated on, and in some cases specifies a particular block of that list.

The request shown in this example was handled by MOLDS to retrieve, display, and process all English language books on printing, or typesetting, or type founding which have bibliographies. The sequence illustrates the flexibility of MOLDS, the many types of processing which can be done, the relatively easy way to use command format. This particular sequence was performed in the on-line version with chance for user-system interaction after each command.

MOLDS Search Query—Example 2 (On-Line mode)

MOLDS Commands:	*Explanation:*
FIND BIBL MARC/BIB/E/X/	Find all records in the file named MARC for which the block named BIB contains a value equal to (E) X (X in the block indicates presence of bibliographies). The list of selected records is to be stored in a location called BIBL.
FIND ENGL MARC/LANG/E/ENG/	Find all documents in the file named MARC for which the block named LANG contains a value equal to (E) ENG, i.e. English language books. The list of selected records is to be stored in a location called ENGL.
DEFINE BOTH BIBL/AND/ENGL/	Define a new list called BOTH which consists of the documents common to both BIBL and ENGL, i.e., all English language books with bibliographies.
STORE SUBS 3/ALPHA/13/	Inform the system that the user wishes to store, via the console, a list of values which will be called SUBS. The list will contain 3 elements which will be alphanumeric (ALPHA) as opposed to strictly numeric. The longest element will not exceed 13 characters.
ELEMENT 1 =	(System responds with these words.)
PRINTING/	User inserts first value by typing it on the console.
ELEMENT 2 =	(System responds with these words.)
TYPE-SETTING/	User inserts second value.
ELEMENT 3 =	(System responds with these words.)

TYPE-FOUNDING/	User inserts third value. User has now created an independent list of three distinct values — PRINTING, TYPE-SETTING, TYPE-FOUNDING and stored them in a location called SUBS.
SELECT ALL BOTH/SUBJ/SUBS/	Select all records from the list called BOTH for which the values in the block named SUBJ are equal to any of the values in the list called SUBS, i.e. those records for which the subject heading is PRINTING, TYPE-SETTING, or TYPE-FOUNDING. The selected records are stored in a location called ALL.
COUNT NO. ALL/	Count the number of records in the list called ALL. The count is stored in a location called NO.
SHOW NO.//	Display the contents of NO. on the CRT.
PRINT ALL/MAIN/TITL/LCNO //ALL/PUBL/PLCE//	Produce a 5-column printed listing consisting of the values in the blocks named MAIN (main entry), TITL (title), LCNO (library of Congress classification number), PUBL (publisher), PLCE (place of publication) from each record of the list called ALL.
MAXIMUM BIG ALL/HITE/	From the list called ALL, select the record containing the maximum value in the block named HITE (height). The record is stored in a location called BIG.
AVERAGE AVE ALL/HITE/	Calculate the average of the values in the block named HITE (height) of the list called ALL. The value is stored in a location called AVE.

The following example records another interaction and the results in the off-line or batch mode. Notice the error message which did not interrupt the search. This result also includes a report on the length of Central Processing Unit (CPU) time each operation takes in hours, minutes, seconds and tenths of seconds. Any line preceded by *C* indicates that the line was printed by the computer; any line minus the *C* indicates that the information was typed in by the user.

MOLDS Retrieval—Example 3 (Batch Mode)

```
C    PLEASE ENTER YOUR PROGRAM
C    LINE  1
     *********PAULINE ATHERTON**********
C    INVALID COMMAND NAME
C    SET IN AT 185 DAY OF 1969      16-01-17.1
C    LINE  1
     PROGRAM TALLY A/
C    LINE  1
     COUNT B A/
C    LINE  2
     PRINT B//
C    LINE  3
     END
C    SET IN 185 DAY OF 1969         16-01-17.5
C    LINE  2
     FIND D2 SSOZ/DEW2/NE/O?
C    SET IN AT 185 DAY OF 1969      16-02-38.7
C    LINE  3
     FIND D1 SSOZ/DEW1/NE/                /
C    SET IN AT 185 DAY OF 1969      16-03-56.7
C    LINE  4
     TALLY D2/
C        950.00
C    SET IN AT 185 DAY OF 1969      16-03-57.3
C    LINE  5
     TALLY D1/
C        905.00
C    LINE  6
     STOP
```

COMMENTS ON MARC/MOLDS

Thus far this report has been confined to a more or less factual description of the components of the MARC/MOLDS system. No doubt the reader has asked himself many questions about the system, and made his own critical comparisons between this system and others. What follows are preliminary and necessarily subjective comments based on a

few demonstrations given to students in the School of Library Science and on the authors' own observations and reflections.

System Design

Response Time

Response time (i.e. the time between transmission of a command in the on-line version and its execution) has been on the order of 90 seconds for a search of 620 records, to 20 seconds for an arithmetic operation involving the same number of records. When one thinks of these times in comparison with the time required to perform the same operations manually, they seem rapid. However, 90 seconds appears to be an unreasonably long period of time in a computer-based interactive retrieval environment. Viewers of demonstrations often asked why it took the computer "so long" to perform a search. A user's tolerance for delay appears to vary a great deal with the type of retrieval system he is using. This has been observed on other occasions, but no determination has yet been made of tolerable limits in different environments, a determination that would be important in designing computer-based systems.

Man-System Interaction

A design goal of most other existing interactive retrieval systems seems to be to give the computer certain anthropomorphic qualities and make it into a teacher or a responsive friend. Such systems offer computer-aided query formulation and/or a friendly conversation with the computer. The MOLDS on-line system does not include either of these features. The user must first master a MARC/MOLDS manual which is an explanation of the system and the data base. He then goes on line and gives his command. MOLDS responds by performing that command or by putting out a brief error message if the command format was improper.

Apparently the objective of conversation with the computer as found in most systems is to make it easier for the user to achieve desired results or to make him feel more at ease with the system. The person who plays with an interactive system once or twice probably finds conversations with a computer amusing, novel, and helpful in his first attempts. However, for a serious and steady user, carrying on the same conversation with the computer during each and every session can be tedious, repetitive, time consuming and sometimes circular. The optimum mix of computer-aided and independent user-formulated query is yet to be studied and found. Perhaps MOLDS, because it is a poor conversationalist, could aid in this search. At any rate, the automatic assumption of conversational features as a design goal for computer-based retrieval systems may not be based on sound knowledge of what suits the serious user.

MOLDS Repertory of Commands

The processing commands in the MOLDS query language are a wel-

come and valuable addition to the usual repertory of search and display commands common to most interactive systems. Although the MARC data base does not lend itself to a great deal of processing, we have found some commands useful, particularly COUNT, ORDER, MAXIMUM, MINIMUM, and COMPRESS.

Processing Times

When individual commands of a single search take seconds of CPU time, it is certain that a retrieval system will be expensive if it is employed by a great many users as a general purpose system. Some of the MOLDS commands operating on the MARC data base took whole minutes of CPU time! The authors have learned a great deal about interactive retrieval systems by using MOLDS experimentally, but because of the excessive cost of certain runs, may not be able to continue research with it. Modifications will have to be made to make it more efficient (i.e. cheaper to run) before it could be recommended for general use in the Syracuse University Library School or anywhere else.

If the MOLDS system can be designed to yield good results for certain types of searches with a realistic file size, it will be a boon to the library or educational institution seeking to automate some part of its searching procedures.

Data Base

Noah Prywes (14) has commented, "The effectiveness in retrieving documents is highly dependent on the amount of labor and processing invested in the storage of documents." The minimum amount of processing done on the MARC tapes has, in fact, limited the effectiveness of retrieval. The extreme simplicity of the general MOLDS data base structure is worthy of study. The efficiency and cost of retrieval using this structure needs to be compared very carefully with more sophisticated threaded lists. One extremely important factor to consider will undoubtedly be the effect of increasing the size of the file.

As pointed out before, the MOLDS system requires an exact match of punctuation and spelling between retrieval criteria and stored data items, a match difficult to achieve. To be sure, this is partially a limitation in the MOLDS system that may be relaxed by incorporating a capability to search for root words and key letter combinations. However, the many inconsistencies in abbreviations, punctuation, and spelling that appear in bibliographic records when information on title pages is transcribed, as on the MARC tapes, can enormously complicate effective retrieval. MARC or non-MARC bibliographic records will always contain some "author" variations that such a system as MOLDS may have to accommodate. This is a very knotty problem.

These comments are not to be construed as a criticism of the fine work the Library of Congress has done in its MARC Pilot Project. The MARC

Pilot Project record format, with sometimes indistinct data elements (special punctuation marks and symbols), was not specifically designed for computer-based interactive search systems. Hopefully, the use herein described to which the MARC data base has been put, and the experience derived from that use, will be of value as future modifications of the MARC format are made. After all, reference retrieval, using bibliographic information, automated or manual, is natural to libraries and is, indeed, one of the purposes for which that information is recorded in the first place. Since one of the true values of a computer-based file lies in making multiple use of the records, it becomes imperative to test the various uses to which these records can be put.

THE FUTURE USE OF MARC/MOLDS AT SYRACUSE UNIVERSITY

The MARC/MOLDS system has undergone continual modification in data base structure and query language during the first year of work on it. A computer-based system must be capable of such flexibility, for changes should be accomplished easily and smoothly. No system is perfect, especially in its early days, least of all MOLDS.

It is intended to continue investigation into information-seeking behavior, and to use MARC/MOLDS occasionally along with other retrieval systems. Another paper describes use of the MARC file with the IBM/Document Processing System (15).

SUMMARY

This report has tried to describe, not sell, MARC/MOLDS as fairly as possible in the belief that some of its features should be considered by persons designing interactive systems, and by those responsible for refinement of the MARC format. The searching capability is valuable as it increases the access points to the data. The arithmetic and logical operations provide an opportunity to perform certain studies of the MARC data base. The MARC files will eventually have many applications beyond technical processing functions in libraries. These applications would be more practically implemented if the MARC format were modified to accommodate them and if librarians would use systems such as MOLDS during their exploration of alternatives.

MARC/MOLDS as a computer-based system has many weaknesses. Outnumbering and to some extent overshadowing the concrete statements about its faults is its great potential. Many questions have been raised which remain unanswered. Questions dealing with the basic design of the system and data base are indicative of the development and experimentation which must be done before computer-based interactive retrieval in libraries is a practical reality.

ACKNOWLEDGMENTS

The work on this project has been supported by Rome Air Develop-

ment Center (Contract S. U. No. AF30 (602)-4283). Related work, supported by a grant from the U. S. Office of Education, provided an education in understanding of the MARC tapes.

The authors gratefully acknowledge the comments made by Phyllis A. Richmond and Frank Martel on the original manuscript. Mrs. Sharon Stratakos, programmer most responsible for MOLDS, contributed a great deal to the authors' understanding of this retrieval program and its potential use with a bibliographic reference file such as MARC.

PROGRAM

Microfiches and photocopies of the following may be obtained from National Auxiliary Publications Service of ASIS: "Rome Project Program Description: MOLDS Support Package" (NAPS 00884).

REFERENCES

1. Avram, Henriette: *The MARC Pilot Project, Final Report* (Washington, D. C.: Library of Congress, 1968).
2. *A User-Oriented On-Line Data System* (Syracuse, N. Y.: Syracuse University Research Corp., 1966). 2 v.
3. Freeman, Robert R.; Atherton, Pauline: *AUDACIOUS—An Experiment with an On-Line Interactive Reference Retrieval System Using the Universal Decimal Classification as the Index Language in the Field of Nuclear Science* (New York: American Institute of Physics, April 25, 1968) (AIP/UDC—7).
4. Burnaugh, H. P.; et al: *The BOLD User's Manual (Revised)* (Santa Monica, Cal.: Jan. 16, 1967) (TM—2306/004/01).
5. Cegala, L.; Waller, E.: *COLEX User's Manual* (Falls Church, Va.: System Development. Feb., 1969) (TM—WD—(L)—405/000/00).
6. Smith, J. L.; MICRO: *A Strategy for Retrieving Ranking and Qualifying Document References* (Santa Monica, Cal.: Jan. 15, 1966) (SP 2289).
7. Green, James Sproat: *GRINS: An On-Line Structure for the Negotiation of Inquiries* (Bethlehem, Pa.: Lehigh University, Center for the Information Sciences, September 1967).
8. Computer Command and Control Company: *Description of the Multilist System* (Philadelphia, Pa.: July 31, 1967.
9. National Aeronautics and Space Administration, Scientific and Technical Information Division: *NASA/RECON User's Manual* (Washington, D. C.: October 1966).
10. Kessler, M. M.: *TIP User's Manual* (Cambridge, Mass.: Massachusetts Institute of Technology, Dec. 1, 1965).
11. Biomedical Communication Network: *User's Training Manual* (Syracuse, New York: December 1968).
12. Welch, Noreen O.: *A Survey of Five On-Line Retrieval Systems* (Washington, D. C.: Mitre Corp., August 1968) (MTP-322).

13. Avram, Henriette D.; Knapp, John F.; Rather, Lucia J.: *The MARC II Format* (Washington, D. C.: Library of Congress, 1968).
14. Prywes, Noah S.: *On-Line Information Storage and Retrieval* (Philadelphia, Pa.: University of Pennsylvania, Moore School of Electrical Engineering, June 1968).
15. Atherton, P.; Wyman, J.: "Searching MARC Project Tapes Using IBM/Document Processing System," *Proceedings of American Society for Information Science,* 6 (1969), 83-88.