1967

An experimental approach to the prediction of retrieval timme using inverted and sequential search techniques for files stored on the disk packs of an IBM OS/360 MOD 50

Morris B. Yagunda
Lehigh University

AN EXPERIMENTAL APPROACH TO THE PREDICTION OF RETRIEVAL TIME USING INVERTED AND SEQUENTIAL SEARCH TECHNIQUES FOR FILES STORED ON THE DISK PACKS OF AN IBM OS/360 MOD 50

by

Morris B. Yaguda

A Thesis

Presented to the Graduate Faculty

of Lehigh University

in Candidacy for the Degree of

Master of Science

Lehigh University

1967

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

May 19, 1967
Date

John M. Carroll
Professor in Charge

Head of the Department

ACKNOWLEDGMENTS

Particular thanks are due to Professor J. M. Carroll, my thesis advisor, for the overall guidance of this study, and to Professors W. A. Smith and G. E. Whitehouse for their suggestions in presenting the concept.

Mr. G. E. Whitney and Dr. R. P. Thayer of the Western Electric Engineering Research Center deserve thanks for their encouragement and advice.

Mrs. Marie R. Schultz of the Western Electric Research Center has earned my grateful acknowledgment for her tireless efforts in preparing the several drafts of this paper.

Sincere appreciation is expressed to all who have made my participation in the graduate program a reality.

Finally, my wife, Karen, deserves my special gratitude for the patience and loving encouragement which she has always shown.

TABLE OF CONTENTS

Table of Contents (continued)

LIST OF TABLES

ABSTRACT

In the course of designing file organization systems, comparing packaged generalized file processing systems, or when evaluating alternative hardware/software configurations, it is often desirable to have a good estimate of system performance before committing large expenditures of money and manpower.

This thesis is intended to illustrate the experimental approach to the determination of performance prediction functionals. It is suggested that this approach could be useful for the class of problems considered to be "bench-marks" for computing systems.

The scope of this paper will be limited to the use of an experimental approach to find a prediction formula for the time required to retrieve data from disk stored files under certain specified conditions. Needless to say, other conditions could have been specified, and a different formula determined. What is significant, however, is the experimental approach itself. The formula developed in this paper can be used with confidence, however, if the intended application meets the conditions specified in this paper.

This paper begins with a discussion of the information retrieval problem being investigated. The inverted file and sequential techniques for "solving" this problem are then described since they will be used in the paper.

The thesis then proceeds by specifying the experimental conditions, selecting the appropriate experimental designs, and presenting the results of the experiment. The experimental results are then discussed and verified.

I INTRODUCTION

A. Objective of Thesis

The object of this thesis is to determine a formula which can be used, under certain conditions, to predict the time required to retrieve records from a file which satisfy a multiple element inquiry using sequential and inverted file retrieval techniques.

B. Conditions of Applicability

The conditions under which this formula will apply are:

1. Each element of the multiple element inquiry is of the form, R_i, where

$$R_i = \{d_i = C_i\}$$

where d_i is the record field number and C_i is an index term that might be contained in field number d_i of any record in the file.

2. The index terms are uniformly distributed among the records and among the inquiries.

3. The file will be stored in one contiguous area on IBM 2311 disk packs.

4. The inquiries will be processed by an IBM System OS/360 Mod 50. Inquiries will be received as card input in an IBM 2540 card reader and responses will be printed out on an IBM 1403 high speed printer.

5. The inverted file will also be stored in a contiguous area on an IBM 2311 disk pack.

6. All record addressing will be accomplished by means of the standard IBM index sequential access method.

7. All records will be stored as fixed length, fixed format records.

8. The independent variables all lie within the following ranges:

(There are four categories of independent variables that will be investigated in this experiment: Basic File Factors, Inverted File Factors, Disk Storage Factors, and Inquiry Factors. The nature of these factors will be discussed below. These factors have been selected on the basis of anticipated (possible) significant effect on retrieval time. These factors will be referred to by alphabetic code in the remainder of this paper.)

a. Basic File Factors

1. Basic record length - the number of characters in each basic record. (Factor A)

2. The number of characters in the basic record call-up number. (Factor B)

3. The number of indexing terms in each basic record. (Factor C)

4. The number of characters in each index term. (Factor D)

5. Total number of distinct index terms in the basic file. (Factor E)

6. Total number of records in the basic record file. (Factor F)

b. **Inverted File Factors**

7. Maximum number of basic record call-up numbers stored in each inverted file record. (Note: If additional call-up numbers must be stored, additional inverted file records will be chained on.) (Factor G)

c. **Disk Storage Factors**

8. Address look-up tables and file records on same (or different) disk pack(s). (Factor H)

9. Inverted and basic files on same (or different) disk pack(s). (Factor J)

10. Number of I/O buffers associated with basic file. (Factor K)

11. Number of I/O buffers associated with inverted file. (Factor L)

12. Number of inverted file records in a block as stored on disk. (Factor M)

13. Number of basic file records in a block as stored on disk. (Factor N)

d. Inquiry Factors

14. Inquiry requests print out of complete basic record or call-up number only. (Factor O)

15. Number of "and" elements in each inquiry. (Factor P)

e. Levels Used for Each Factor

These levels were selected as being representative and, at the same time, computationally feasible. The levels are:

Factor	High Level	Low Level
A	120 characters	75 characters
B	10 "	5 "
C	10 terms/record	5 terms/record
D	6 char/term	3 char/term
E	100 terms/file	50 terms/file
F	2000 records/file	1000 records/file
G	100 call-up nbrs/rec	50 call-up nbrs/rec
H	1 = same pack	0 = different packs
J	1 = same pack	0 = different packs
K	2 buffers/file	1 buffer/file
L	2 buffers/file	1 buffer/file
M	3 rec/block	1 rec/block
N	15 rec/block	2 rec/block
O	1 = full record	0 = call-up nbr only
P	3 elements/inquiry	2 elements/inquiry

II THE GENERAL RETRIEVAL PROBLEM

A. The Inquiry

The general information retrieval problem is to find those records in a file which satisfy a particular logical constraint on one or more fields. Before elaborating on this point, it is convenient to define "field-value". The term "index term" is often used synonomously with "field-value".

A field within a record may contain one or more alphanumeric or special characters. We shall define a particular set of characters as a "field value". That is, considering a field to be an algebraic variable which may only assume values from a finite set, a field-value is <u>one</u> of the values from this finite set.

For example, if the "i"th record consisted of English prose, then the "j"th <u>field</u> in that record might assume the <u>value</u> of any word in the dictionary.

A retrieval request, R, in general, asks for all records containing one or more particular field-values consistent with a given logical restraint. That is, let C_1, C_2 ... C_q be q distinct field values mentioned in a request. Then R has the following form:

$$R = R_1 \cap R_2 \cap \ldots \cap R_q$$

where

R_i = "i"th component of retrieval request R

$\cap$ = Set algebra operator for "intersection" which is equivalent to logical "AND"

Each component, R_i, of the retrieval request has the following general form

$$R_i = \{d_i, \text{relational-operator}_i, C_i\}$$

where

d_i = integer denoting which field in the record is involved in the "i"th component of the retrieval request.

relational operator$_i$ = one of the logical operators from the set

$$\{=, \neg =, >, \neg >, <, \neg <\}$$

C_i = a set of characters (i.e., a field-value) which might be contained in the "d_i"th field of one or more records.

Referring to the personnel example cited earlier, a request asking for all records with department number (field 2) of 7202 and job code (field 3) of 1402 would map into the following general form

$$R = R_1 \cap R_2$$

where

$$R_1 = \{2, =, 7202\}$$
$$R_2 = \{3, =, 1402\}$$

The phrase "multiple index term inquiry" will be used in this paper to denote an inquiry of the general form described above.

Any retrieval request, no matter how complicated, can always be reduced to a series of retrieval requests of the above general form. For example, consider the following request, R, which asks for all records having R_1 and R_2 and either R_3 or R_4.

$$R = R_1 \cap R_2 \cap (R_3 \cup R_4)$$

This may be reduced, using set algebra, to the following:

$$R = (R_1 \cap R_2 \cap R_3) \cup (R_1 \cap R_2 \cap R_4)$$

This may be considered to be two requests $R^{(1)}$ and $R^{(2)}$ where

$$R^{(1)} = R_1 \cap R_2 \cap R_3$$

$$R^{(2)} = R_1 \cap R_2 \cap R_4$$

Therefore, the general form for retrieval requests presented above may be considered to be an upper bound on the complexity of retrieval requests that should be considered when planning an information retrieval system.

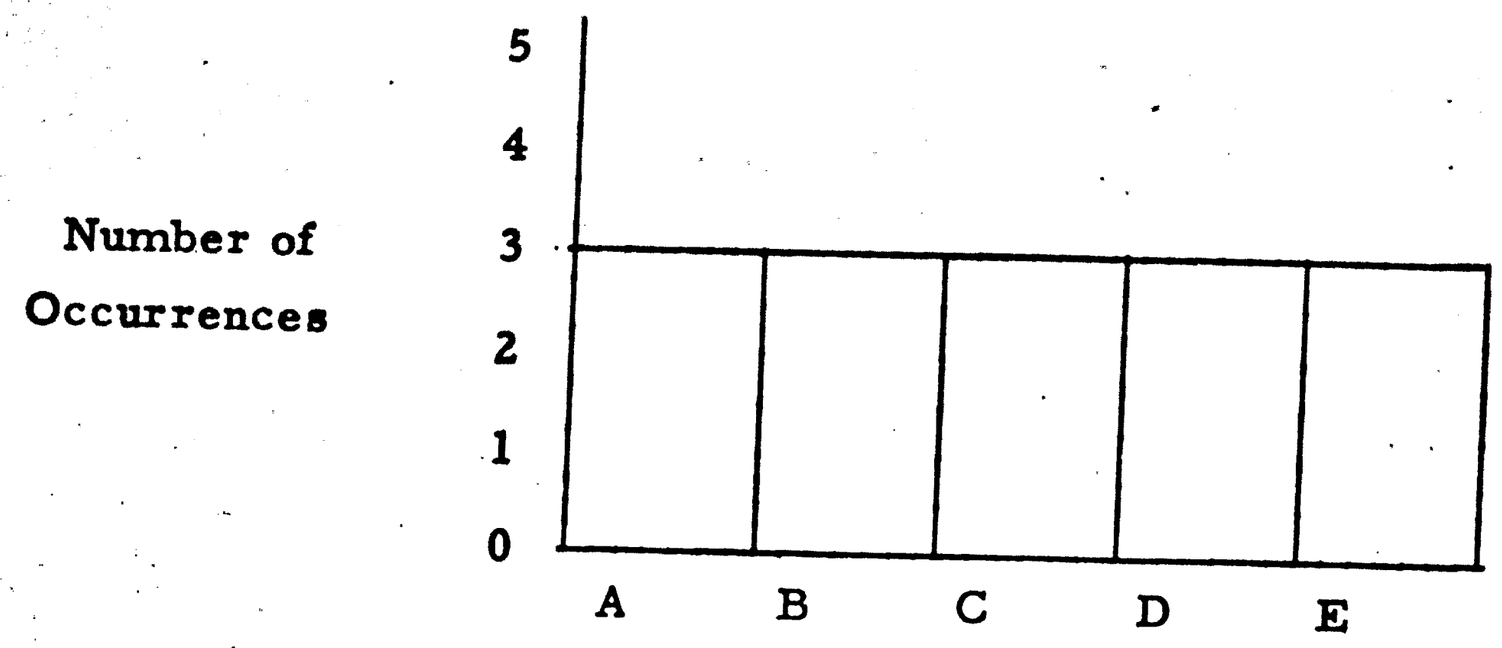
B. Distribution of Index Terms

If we consider a given basic file, it is possible to construct a histogram representing frequency of occurrence of each index

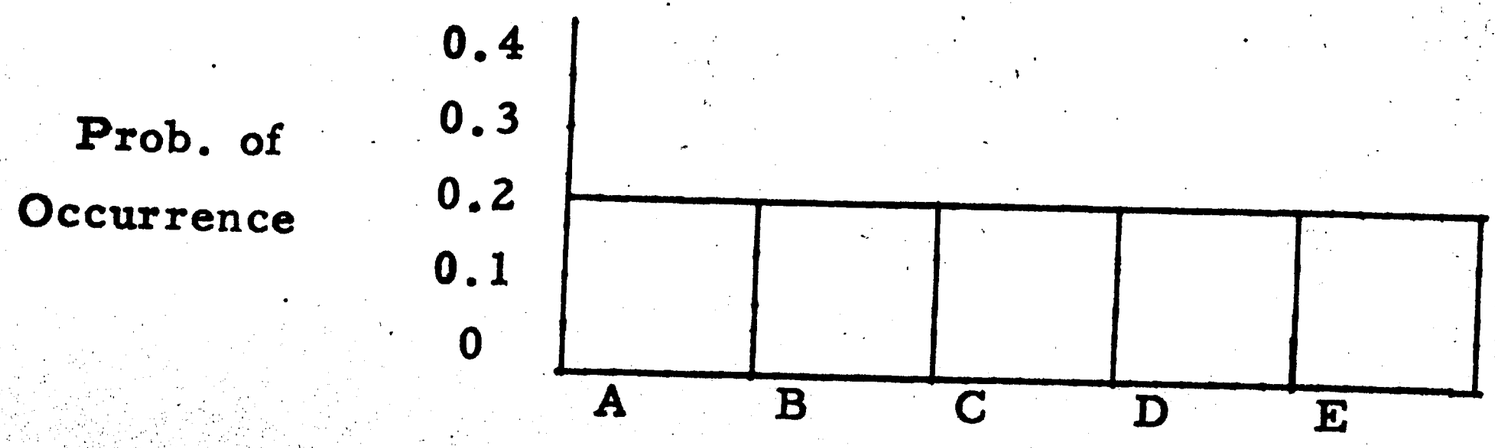
term in the file. For example, consider the following file consisting of five records. Each record contains a document number and up to three descriptive terms.

Document Number	Index Terms
1	B, C, E
2	A, C, D
3	A, B, D
4	A, C, E
5	B, D, E

For this file, we can construct the following histogram.



The number of occurrences of index terms can be normalized and a probability distribution constructed. For this example, we have



For this example, the inquiries are distributed among the records according to a uniform distribution.

A similar probability distribution could be constructed for the probability of occurrences of each index term in the inquiries.

In this paper, a uniform distribution of key values among the records and in the inquiries will be assumed. Similar studies to this one could be performed assuming other distributions.

III. THE INVERTED FILE TECHNIQUE

A. Description

There are several known techniques for retrieving records from random access devices when the retrieval is to be made on the basis of multiple index terms.

A description of the most common methods is in Appendix 2.

The inverted file method is generally considered to be the best for a wide range of applications. In this paper, only the inverted file and sequential search method will be investigated. A similar experimental approach is directly applicable to the other methods, and will be suggested as a theme for further study.

The inverted file technique will be described in some detail below. Appendix 3 contains illustrations of the terminology used for file structure.

1. The Basic File

Let us denote a conventional file as the "basic file". Each record in this file contains a unique identifier (call-up number) and several other units of information (field-values).

2. The Inverted File

Let us now transform the basic file into a new file (called an inverted file) using the following rule for mapping:

Rule: For each unique field-value which might appear in an inquiry against this file, create an inverted file record containing the call-up numbers of those basic records which contain the given field-value.

B. Example of Inverting a File

Consider the following file consisting of five records. Each record contains a document number, and up to three descriptive index terms.

Document Number	Index Terms
1	A, B, C
2	B, C, D
3	C, D, E
4	A, E
5	B, D, E

Using the rule stated above, we can "invert" this file to obtain:

Index Term	Document Numbers
A	1, 4
B	1, 2, 5
C	1, 2, 3
D	2, 3, 5
E	3, 4, 5

It should be noted that each unique index term serves as the identifier for the inverted file record associated with it, and that

the index terms are approximately uniformly distributed among the records.

C. Utility of the Inverted File

The utility of the inverted file technique becomes apparent when one considers an inquiry of the form, "Which records have a given set of index terms?".

Consider the example given above. If we ask, "Which records have B and C?", we have two choices. First, we can look at the basic file and examine each record in sequence to see if it contains B and C. This is, of course, the sequential search technique. Secondly, we can go to the inverted file and look at the inverted file records for B and C. The "intersection" of the inverted records for B and C contains the call-up numbers for the basic basic records containing both B and C. That is, the set intersection of {1, 2, 5} and {1, 2, 3, 4} is {1, 2}. These two basic records contain B and C.

Clearly the above example is trivial. However, the difficulty of the search increases rapidly with an increase in the number of index terms per record and/or the number of basic records.

If we consider a file containing thousands of records with a dozen or so index terms per record and several hundred (or more) different index terms, the problem of conducting a search for those records containing a given set of index terms is quite formidable. This problem, therefore, lends itself to a computer solution.

It should be clear at this point that the inverted file technique is strictly applicable only to a subset of the general form of the inquiry as described above. That is, the inverted file technique is only of utility when processing inquiries whose elements contain only the "=" relational operator.

However, inquiries which contain elements involving other relational operators may be processed in two steps. First, the inverted file technique may be used to retrieve those records satisfying the elements involving the "=" operator. Secondly, these records can then be checked sequentially to see if they satisfy the other inquiry elements.

IV THE EXPERIMENTAL APPROACH

A. General Procedure

The general procedure to be used in this paper for the experimental approach to determining retrieval time is the following:

1. Specify the system which will be the object of the experiment.
2. Select the appropriate experimental designs.
3. Design and perform the experiment.
4. Use multiple regression to find the desired functional.
5. Verify the accuracy of the functional.

B. Specification of System

The system being studied in this paper consists of two phases: a file creation phase, and an inquiry processing phase. This is illustrated in Figure IV-C.

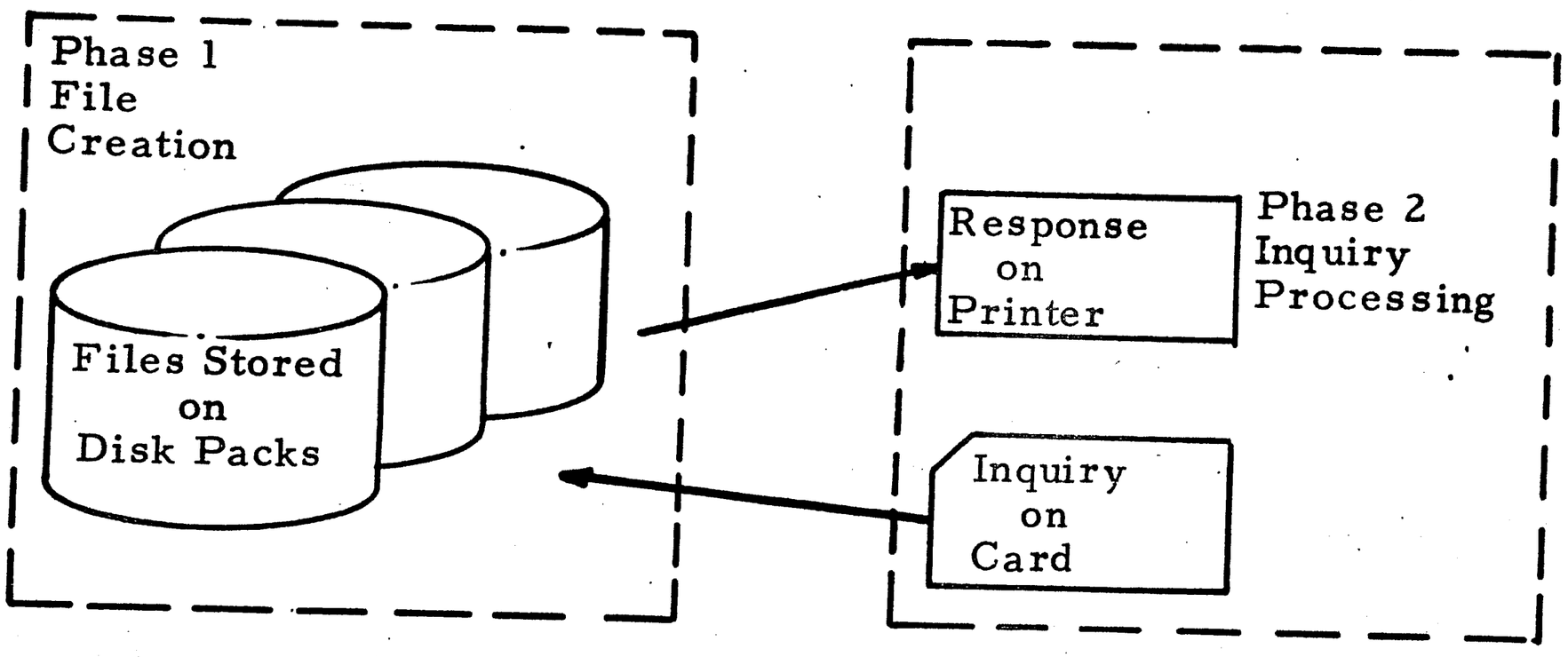


Figure IV-C Phases of the System

1. Assumed Characteristics of the File Creation Phase - Basic File

a. Each basic file record will have a unique identification number and several index terms.

b. The index terms will be uniformly distributed among the basic records by using a pseudo random number generator (power residue method) with a different seed each time the file is created. The seeds will come from a table of random numbers.

c. The basic file records will be stored in the sequence of their identification numbers.

d. The basic file records will be stored in one contiguous area on a disk pack.

e. A table look-up procedure will be used to randomly access a given basic file record.

f. The main directory used for the table look-up procedure will be stored in one contiguous area on a disk pack. The entries in this table will identify the cylinder on which a given record is located. The first track in the cylinder will then contain a low order table which will give the disk address corresponding to a given record. (Note: This is the standard IBM index sequential access method)

g. The main directory area for the basic file can be

located on a different disk pack than the basic file area itself, or they may both reside on the same pack.

h. The basic file records will be fixed length, fixed format records.

i. The basic file records will be blocked when stored on disk.

2. Assumed Characteristics of the File Creation Phase - Inverted File

a. Each index term in the basic file will identify an inverted file record.

b. The inverted file records will be fixed length, fixed format records and will contain an identification number, a chaining field, and a fixed number of positions in which to store the identification numbers of the basic file records which contain the given index term.

c. If this fixed number of positions is not enough to store all the basic file identification numbers, then the chaining field will be used to indicate the next inverted record which also corresponds to the given index term. In this way, as many inverted file records as are required may be "chained" together.

d. The inverted records will be stored in sequence of their identification numbers.

e. The inverted file records will be stored in one contiguous area on a disk pack.

f. A table look-up procedure will be used to randomly access a given inverted file record.

g. The standard IBM index sequential access method will be used.

h. The main directory area for the inverted file can be located on a different pack than the inverted file area itself, or they may both reside on the same pack.

h. The inverted file (and directory) areas may or may not be on the same packs as the basic file (and directory) areas.

i. The inverted file records may or may not be blocked when stored on disk.

3. Assumed Characteristics of the Inquiry Phase

a. Each inquiry will contain several elements of the form {field number, =, index term}. These elements will be joined by the logical operator, "AND".

b. The field number and index terms will be uniformly distributed among the inquiries by using a pseudo random

number generator (power residue method) with a different seed for each set of inquiries. The seeds will come from a table of random numbers.

c. The inquiry may request that the entire record be printed out, or it may request only the identification number of those records satisfying the inquiry.

V. GENERAL PROCEDURE FOR THE EXPERIMENT

The experiment will consist of a set of trials. Each trial consists of two phases. First, a basic file will be created, inverted, and stored on disk in accordance with a particular treatment of the basic file, inverted file, and disk storage factors.

Secondly, after the creation of these files, inquiries will be processed using the inverted file system in accordance with each of the four possible combinations of the two inquiry factors.

Furthermore, as a control block for comparative purposes, an identical set of inquiries will be processed using a sequential search of each record in the basic file. This sequential search will also be performed in accordance with each of the four possible combinations of the two inquiry factors.

The experiment will proceed in four stages:

Stage 1 A sufficient number of trials will be carried out to determine which of the fifteen factors being investigated have significant main effects.

Stage 2 Those factors which are found to have significant main effects will be investigated further. Additional trials will be carried out, by varying only those factors found to have significant main effects, to determine which of the first order interactions between these factors are themselves significant.

Stage 3 Multiple regression analysis will then be used to find the least squares fit for the experimentally obtained data as a function of the main and first order interaction effects identified as significant above.

Stage 4 The functional determined in Stage 3 will then be verified by comparing predicted to actual results for files created at levels other than those used in Stages 1 and 2. If gross errors exist in prediction, then additional experimentation will be required to evaluate higher order effects. If the errors fall within 95% confidence intervals, the linear model will be considered adequate.

VI. DESIGN AND RESULTS FOR STAGE 1 - TEST OF MAIN EFFECTS

A. Design - Stage 1

1. Main Plot - File Generation Factors

An independent measure of the significance of each of the thirteen file generation factors can be obtained by using a standard fractional fractorial design. This design is a 1/256 replication of 13 factors consisting of 32 treatments. This plan is reproduced below in standard form.

Plan 256.13.16[1]

Block 1	Block 2
(1)	eghjkm
abcdefghiklmn	abcdfln
adjmn	adeghkn
bcefghkl	bcfjlm
cdfghn	cdefjkmn
abejklm	abghl
acfghjm	acefk
bdekln	bdghjlmn
fgjkln	efhlmn
abcdehm	abcdgjk
adfgklm	adefhjl
bcehjn	bcgkmn
cdhjkl	cdeglm
abefgmn	abfhjkn
achklmn	acegjln
bdefgj	bdfhkm

---

[1] "Fractional Factorial Experiment Designs for Factors at Two Levels;" U.S. Department of Commerce, National Bureau of Standards, Applied Mathematics Series Nbr 48, p. 75, April, 1957.

Due to the fact that the file generation phase of each trial consumes a sizeable amount of time (approximately 5-10 minutes), and that a large number of treatments (32) must be carried out, this main plot design will have only one replicate. (i.e. Each treatment will be used once.) The total number of trials for the main plot in this first stage will therefore be thirty-two.

The experimental error will be estimated by pooling the sums of squares of the 16 effects other than the main effects. It should be noted that this error contains the first order interactive effects as well as the true error. It is, however, the best estimate that can be made without resorting to a large number of additional trials.

2. Sub Plot - Inquiry Factors

Once the file has been generated, inquiries will be processed against the file. Each inquiry will be processed first by the inverted file technique and then, for comparison purposes, by sequentially searching the entire file. This experimental approach is, of course, based on a split-plot design technique. Each inquiry will be processed four times, each such time corresponding to one of the four treatments formed by the four possible combinations of the two inquiry factors, each of which has two levels.

The sub plot will have two replicates. For each replicate, a different inquiry will be processed. The field number

and index terms in the inquiries are selected on the basis of a uniform distribution, using a different seed for the pseudo random number generator for each trial.

The sub plot analysis can be carried out by considering the file generation phase as a factor, X, which has thirty-two levels. Each level corresponds to one of the treatments used when generating the file. The inquiry processing phase can be considered to be the sub plot factor, Y, which has four levels. Each level corresponds to one of the four possible treatments of the two inquiry factors, each of which has two levels.

Two-way Classification

	X_1	X_2	- - - - - - - - -	X_{32}
Y_1	t_{111}, t_{112},	t_{121}, t_{122}		$t_{1\,32\,1}$, $t_{1\,32\,2}$
⋮				
Y_4	$t_{3\,11}$, $t_{4\,12}$	$t_{4\,21}$, $t_{4\,22}$		$t_{4\,32\,1}$, $t_{4\,32\,2}$

Where X_i the treatment for trial i

Y_1 is inquiry factor treatment P

Y_2 " " " " OP

Y_3 " " " " (1)

Y_4 " " " " O

The standard analysis for a two-way classification can then be carried out to determine whether the Y factor is significant. If it is, then the two inquiry factors may then be independently estimated.

$$SS(P) = \frac{[(S_1 + S_2) - (S_3 + S_4)]^2}{256}$$

$$SS(O) = \frac{[(S_2 + S_4) - (S_1 + S_3)]^2}{256}$$

$$SS(OP) = \frac{[(S_2 + S_3) - (S_1 + S_4)]^2}{256}$$

Where $S_i = \sum_{j=1}^{32} \sum_{k=1}^{2} t_{ijk}$

B. Stage 1 - Results Test of Main Effects

Tables VI-1 and VI-2 show the analysis of variance for the first stage inverted file system search and the sequential search respectively. The purpose of this first stage was to identify which of the factors have significant main effects on retrieval time for the two search procedures. The observed data for this stage is in Appendix 5.

1. Discussion of Table VI-1 - Inverted Search

a. Main Plot

The ANOVA is based on a standard Yates analysis considering the average of the eight observed times for each treatment as the observed time for that treatment.

The ANOVA table for the file generation factors (the main plot) indicates that Factor C is significant at 99%, and that Factors E, F, G are significant at 95% confidence.

Factor H is only significant at 94%, but will be investigated further since the sum of squares for error is enlarged by all effects of order higher than one. Furthermore, since this is a highly fractionated design, the sum of squares computed for each main effect, although independent of all other main effects, contains the sums of squares of 255 other effects of order higher than one. (i.e. These "aliased" effects are "confounded" with the main effects.) For these two reasons, Factor H will be investigated further before being dismissed as insignificant.

The sign of each effect, independent of all other effects, is also listed in the ANOVA table. It will be revealed, in the next stage of this experiment that some

of the first order interactions, thusly confounded, are, indeed, independently significant. It is assumed that all other higher order effects are negligible.

b. Sub Plot

The sub plot ANOVA is based on a standard two-way analysis of variance, in which the error is estimated from the two observations taken for each treatment.

The sub plot ANOVA indicates that the file generation (Factor X), the inquiries (Factor Y), and their interaction are all independently significant at 99% confidence.

c. Inquiry Factors

The inquiries effect (Factor Y) from the sub plot was shown to be significant above. Therefore, the sum of squares for Y is broken into three independent estimates of the inquiry factors, O and P, and their interaction, OP. All three of these are found to be significant when tested against the estimate of error from the sub plot.

Table VI-1 ANOVA For Inverted File System - Stage 1

Main Plot (File Generation Effects)

Effect	DOF	Sum of Squares	Mean Square	F Value	Sign of Effect
A	1	0.620	-	0.017	+
B	1	27.483	-	0.739	-
C	1	602.742	-	16.215**	+
D	1	40.393	-	1.087	-
E	1	215.163	-	5.788*	-
F	1	255.123	-	6.863*	+
G	1	265.382	-	7.139*	+
H	1	157.448	-	4.235 ✓	+
J	1	16.263	-	0.438	+
K	1	0.080	-	0.002	-
L	1	43.020	-	1.157	-
M	1	63.432	-	1.706	+
N	1	50.163	-	1.350	-
Error[1]	18	669.082	37.171	-	
Total	31	2,407.271	-	-	

Sub Plot

X	31	2,407.271	77.654	187.118**	
Y	3	876.565	292.188	704.067**	
XY	93	1,870.687	20.115	48.470**	
Error	128	53.121	0.415		
Total	255	5,207.644			

Inquiry Factors (Obtained From Sub Plot Factor Y)

O	1	614.909	-	1,481.708**	+
P	1	3.084	-	7.431**	+
OP	1	258.572	-	623.065**	-
Error	128	53.121	0.415		
Total	131	929.686			

** Denotes significance at 99% probability.
* Denotes significance at 95% probability.
✓ Denotes significance doubtful, but further investigation will be carried out.

[1] The sum of squares for the error is obtained by pooling all effects other than main effects.

2. Discussion of Table VI-2 - Sequential Search

a. Main Plot

The ANOVA is based on a standard Yates analysis, considering the average of the eight observed times for each treatment as the observed time for that treatment.

The ANOVA table indicates that Factors A, D and F are significant at 99%, and Factor C is significant at 95%. Factor N is only significant at 91% but will be investigated further (for the same reasons that H was investigated further for the inverted search) as discussed above.

b. Sub Plot

The sub plot ANOVA is based on a standard two-way analysis of variance, in which the error is estimated from the replications.

The sub plot ANOVA indicates that the file generation Factors (X) is highly significant but that the inquiry Factors (Y) and their interaction are not significant.

Therefore, the inquiry factors will no longer be investigated for the sequential search technique.

Table VI-2 ANOVA For Sequential Search - Stage 1

Main Plot (File Generation Effects)

Effect	DOF	Sum of Squares	Mean Square	F Value	Sign of Effect
A	1	316,381.711	-	56.823**	+
B	1	4,055.310	-	0.728	-
C	1	29,628.931	-	5.321*	-
D	1	69,358.259	-	12.457**	-
E	1	322.946	-	0.058	-
F	1	271,431.166	-	48.750**	+
G	1	2,919.666	-	0.524	+
H	1	157.959	-	0.028	-
J	1	2,666.451	-	0.479	+
K	1	2,388.564	-	0.429	+
L	1	13,777.287	-	2.474	-
M	1	93,223	-	0.017	-
N	1	19,226.370	-	3.453✓	-
Error[1]	18	100,221.312	5,567.851	-	-
Total	31	832,629.450	-	-	

Sub Plot

X	31	832,629.450	26,859.015	3,651.307**	+
Y	3	26.483	8.828	1.200	+
XY	93	234.838	2.525	0.343	-
Error	128	941.568	7.356		
Total	255	833,832.339			

** Denotes significance at 99%.
* Denotes significance at 95%.
✓ Denotes significance doubtful, but further investigation will be carried out.

[1] The sum of squares for the error is obtained by pooling all effects other than main effects.

3. Summary of Results of Stage 1

a. Inverted File Search - Significant Factors

C - The number of index terms in each record.

E - The number of distinct index terms in the basic file.

F - The number of basic records in the file.

G - The maximum number of call-up numbers in an inverted record.

H - Directories and records stored on same (different) pack(s).

O - Output of call-up numbers or full basic record.

P - The number of elements in the inquiry.

OP - The interaction between O and P.

b. Sequential Search - Significant Factors

A - The number of characters in the basic record.

C - The number of index terms in each basic record.

D - The maximum number of characters for each index term.

F - The number of basic records in the file.

N - The number of basic records per block on disk.

VII. DESIGN AND RESULTS FOR STAGE 2 - TEST OF FIRST ORDER INTERACTIONS

A. Design - Stage 2

1. Main Plot - File Generation Factors

In Stage 1, five of the original 13 file generation factors were selected for further investigation for each of the two search techniques. This further investigation will consist of determining which of the first order interactions of these five selected factors are also significant.

A full, completely randomized 2^5 factorial design will be used for this stage. The experimental error will be estimated by pooling the sums of squares of all effects of order higher than first order interactions. Once again, due to the time required to carry out the necessary 32 treatments, a single replicate will be employed. The designs for the second stage for the two search techniques are presented below.

Plan for Inverted Search (Factors C, E, F, G, H)

(1)	g	h	gh
c	cg	ch	cgh
e	eg	eh	egh
ce	ceg	ceh	cegh
f	fg	fh	fgh
cf	cfg	cfh	cfgh
ef	efg	efh	efgh
cef	cefg	cefh	cefgh

Plan for Sequential Search (Factors A, C, D, F, N)

(1)	f	n	fn
a	af	an	afn
c	cf	cn	cfn
ac	acf	acn	acfn
d	df	dn	dfn
ad	adf	adn	adfn
cd	cdf	cdn	cdfn
acd	acdf	acdn	acdfn

2. Sub Plot - Inquiry Factors

In the first stage, the inquiry factors were shown to be insignificant for the sequential search. However, the inquiry factors were shown to be highly significant for the inverted file search. Therefore, the inquiry factors will be investigated further in relation to the inverted search only. Once again, there will be two inquiries processed by each of the four possible treatments of the inquiry factors. The elements of the inquiries will be selected randomly as before.

B. Stage 2 - Results - Test of Main and First Order Interaction Effects

The factors identified as significant in Stage 1 of this experiment were investigated further in a full factorial design. The analysis of variance for these full factorial designs are presented in Tables VII-1 and VII-2. The observed data for this stage is in Appendix 5.

1. Discussion of Table VII-1 - Inverted Search

a. Main Plot

The Yates analysis of the main plot indicates that C, E, CE, F, CF, EF, G, and EG are significant at 99%, and H and EH are significant at 95%. The sign of each of these effects acting independently is given in the table.

b. Sub Plot

The sub plot analysis indicates that file generation factors, inquiry factors and their interaction are all significant at 99%. The sum of squares for the inquiry factors will be analyzed below.

c. Inquiry Factors

The analysis of the inquiry factors indicates that O, P, and OP are all significant at 99% confidence.

Table VII-1 ANOVA For Inverted Search - Stage 2

Main Plot (File Generation Effects)

Effect	DOF	Sum of Squares	Mean Square	F Value	Sign of Effect
C	1	548.248	-	151.685**	+
E	1	203.417	-	56.280**	-
CE	1	125.047	-	34.596**	-
F	1	295.155	-	81.661**	+
CF	1	38.998	-	10.790**	+
EF	1	55.690	-	15.407**	-
G	1	91.651	-	25.357**	+
CG	1	8.578	-	2.373	-
EG	1	50.106	-	13.863**	+
FG	1	0.274	-	0.075	-
H	1	26.315	-	7.280*	+
CH	1	8.540	-	2.363	+
EH	1	23.204	-	6.420*	-
FH	1	1.056	-	0.292	-
GH	1	1.773	-	0.490	-
Error[3]	16	57.830	3.614	-	
Total	32	1,535.873	-	-	

Sub Plot

X	31	1,535.873	49.544	155.872**
Y	3	840.213	280.071	882.630**
XY	93	1,056.713	11.363	35.750**
Error	128	40.685	0.3178	
Total	255	3,473.484		

Inquiry Factor Effects (Obtained from Sub Plot Factor Y)

O	1	552.798	-	1,739.179**	+
P	1	3.342	-	10.514**	+
OP	1	284.068	-	893.717**	-
Error	128	40.685	0.3178		
Total	131	880.898			

** Denotes significance at 99%.
* Denotes significance at 95%.

[3] The sum of squares for the error was obtained by pooling all effects of order higher than two.

2. Discussion of Table VII-2 - Sequential Search

a. Main Plot

The Yates analysis of the main plot indicates that A, C, D, and F are significant at 99%, and CD and N at 95%.

b. Sub Plot

The sub plot analysis was not made since the inquiry factors were shown to be significant in Stage 1 of this experiment.

Table VII-2 ANOVA For Sequential Search - Stage 2

Main Plot (File Generation Effects)

Effect	DOF	Sum of Squares	Mean Square	F Value	Sign of Effect
A	1	323,351.023	-	163.605**	+
C	1	24,112.383	-	12.200**	-
AC	1	61.856	-	0.031	-
D	1	56,702.385	-	28.689**	-
AD	1	2,256.309	-	1.141	-
CD	1	14,714.569	-	7.445*	-
F	1	313,498.368	-	158.620**	+
AF	1	44,241.864	-	22.384	-
CF	1	637.398	-	0.322	-
DF	1	3,877.428	-	1.961	-
N	1	10,165.076	-	5.143*	-
AN	1	1,011.494	-	0.511	+
CN	1	364.228	-	0.184	+
DN	1	1,583.005	-	0.800	+
FN	1	2,785.715	-	1.409	-
Error[3]	16	31,622.564	1,976.41		
Total	32	830,985.16			

** Denotes significance at 99%.
* Denotes significance at 95%.

[3] The sum of squares for the error was obtained by pooling all effects of order higher than two.

3. Summary of Results of Stage 2

On the basis of the analysis of variance for Stage 2, the following effects were found to be significant and will be used for Stage 3.

1. Inverted Search

C - Number of index terms in each basic record.

E - Total Number of different index terms in the file.

CE - Interaction between C and E.

F - Number of basic records in file.

CF - Interaction between C and F.

EF - Interaction between E and F.

G - Maximum number of call-up numbers in inverted record.

EG - Interaction between E and G.

H - Directory and file on same (or different) pack(s).

EH - Interaction between E and H.

O - Printout consists of full record or call-up number only.

P - Number of elements in the inquiry.

OP - Interaction between O and P.

2. Sequential Search

A - Number of characters in the basic record.

C - Number of index terms per record.

D - Maximum number of characters per index term.

CD - Interaction between C and D.

F - Number of basic records in file.

AF - Interaction between A and F.

N - Number of basic records per block on disk.

VIII. STAGE 3 - MULTIPLE REGRESSION

In this stage of the experiment, multiple regression will be used to find the polynomial which has the least squared deviation from the experimentally obtained data. The variables in this polynomial will be those main and first order interaction effects found to be significant in Stage 2 for the inverted and sequential search techniques. The interaction between the file generation and inquiry factors for the inverted search, which was found to be significant in the sub plot analyses of Stages 1 and 2, will also be considered.

A. Multiple Regression for Inverted Search

As a result of Stages 1 and 2, we have determined the independent variables that are significant. Stage 1 estimated which main effects were significant.

The analysis of variance for Stage 2 gave an estimate of which main and first order effects were independently significant. In addition, the sub plot analysis of Stage 2 indicated that the XY interactions between the file generation factors are independently significant.

We do not have an independent estimate of the significance of these main plot-sub plot (XY) interactions. This additional information can be obtained in two ways. Either the split plot design must be abandoned and replaced by a completely randomized

factorial design encompassing all factors, or the multiple regression analysis can be used to discriminate between which of these XY interactions are significant for the model and which are not.

The first alternative would be very costly and time consuming. The second alternative causes some reduction in the degrees of freedom for estimating error in the multiple regression analysis of variance, which implies some decrease in sensitivity when applying the F ratio test. However, there are 238 degrees of freedom for estimating error. A glance at the F tables reveals that the difference in F values between 238 and infinity is not very significant. Therefore, the second alternative will be followed. That is, we will use multiple regression analysis considering, as independent variables, those main and first order effects found to be significant (i.e. C, E, F, G, H, O, P, CE, CF, EF, EG, EH, OP) and, in addition, all of the XY interactions (i.e. OC, OE, OF, OG, OH, OCE, OCF, OEF, OEG, OEH, PC, PE, PF, PG, PH, PCE, PCF, PEF, PEG, PEH). Stepwise multiple regression will be used. Variables will be introduced in the model in the order of their significance if they are <u>at least</u> 95% significant.

The analysis of variance for the multiple regression, Table VIII-1, indicates the significance of each independent variable in the resultant mathematical model for retrieval time. The relative magnitudes of the F ratios indicate the relative significance of the independent variables in the model.

The mathematical model produced by the multiple regression is:

$$t = b_0 + \sum_{1}^{17} b_i X_i$$

Where t is retrieval time in seconds

b_i is the coefficient of X_i

X_i is the value of the "i"th independent variable.

The values of the coefficients are presented below:

i	b_i	X_i
0	5.31158	constant
1	0.03124×10^{-2}	OCF
2	-0.05941×10^{-3}	OEF
3	-4.04291	OP
4	0.02052×10^{-2}	PEG
5	0.18272	OE
6	1.66627	OC
7	0.01949×10^{-2}	EG
8	-1.96169	OCE
9	0.05163×10^{-2}	CF
10	4.05500	OH
11	-0.13813	E
12	0.03442	PE
13	-0.03961	OEH
14	-0.01396×10^{-2}	PCF
15	0.39083×10^{-2}	OF
16	-0.02915	G
17	-5.35582	O

This model fits the 256 observed data points with a multiple correlation coefficient of 0.92522.

Table VIII-1 ANOVA For Multiple Regression-Inverted Search

Source	DOF	Sum of Squares	Mean Squares	F-Value
OCF	1	1,372.35	-	653.16**
OEF	1	614.64	-	292.53**
OP	1	164.92	-	78.49**
PEG	1	119.98	-	57.10**
OE	1	110.94	-	52.80**
OC	1	102.98	-	49.01**
EG	1	102.15	-	48.62**
OCE	1	97.00	-	46.16**
CF	1	57.70	-	27.46**
OH	1	50.42	-	24.00**
E	1	43.41	-	20.66**
PE	1	35.98	-	17.13**
OEH	1	34.81	-	16.57**
PCF	1	26.61	-	12.67**
OF	1	15.18	-	7.23**
G	1	13.60	-	6.47*
O	1	10.74	-	5.11*
Error	238	500.06	2.10	
Total	255	3,473.48		

** Denotes 99% confidence.
* Denotes 95% confidence.

B. Multiple Regression for Sequential Search

As a result of Stages 1 and 2, we have determined the independent variables that are significant. The sub plot analyses revealed that the XY interaction was negligible.

Therefore, the multiple regression analysis will be applied in a straightforward manner to these significant independent variables.

The analysis of variance for the multiple regression, Table VIII-2, indicates the significance of each independent variable in the resultant mathematical model for retrieval time. The relative magnitudes of the F ratios indicate the relative significance of the independent variables in the model.

The mathematical model produced by the multiple regression is:

$$t = b_0 + \sum_{i}^{6} b_i X_i$$

Where t is retrieval time in seconds

b_i is coefficient of X_i

X_i is the value of the "i"th independent variable.

The values of the coefficients are presented below:

i	b_i	X_i
0	20.97293	constant
1	10.64587×10^{-4}	AF
2	-2.02172	CD
3	-0.03381	F
4	-0.96944	N
5	5.21572	C
6	5.24116	D

This model fits the observed data with a multiple correlation coefficient of 0.972669.

Table VIII-2 ANOVA For Multiple Regression - Sequential Search

Source	DOF	Sum of Squares	Mean Squares	F-Value
AF	1	655,282.34	-	3,640.89**
CD	1	89,154.98	-	495.36**
F	1	25,420.01	-	141.24**
N	1	10,165.05	-	56.48**
C	1	4,792.07	-	26.63**
D	1	1,582.26	-	8.79**
Error	249	44,814.64	179.98	
Total	255	831,211.36		

** Denotes significance at 99%.

IX. STAGE 4 - VERIFICATION OF FUNCTIONAL

A. General

In order to verify the prediction formulas determined above, the following procedure was adopted.

1. Five additional trials of the experiment were carried out.

2. For each of the above trials, confidence limits were computed for predicted values of time. The observed retrieval times were then compared with this confidence interval.

B. Verification Trial Description

The verification trials may be described by levels used for the dependent variables in the formulas obtained by multiple regression. Once again, a randomly chosen seed was used for the random number generator used for each trial.

Inverted Search

Trial Number	FACTOR LEVELS						
	C	E	F	G	H	O	P
1	9	90	1800	90	1	1	2
2	8	80	1200	60	1	0	3
3	9	54	1500	55	1	1	3
4	6	54	1500	80	0	1	2
5	7	63	1100	70	0	0	2

Sequential Search

Trial Number	FACTOR LEVELS				
	A	C	D	F	N
1	75	9	5	1800	2
2	100	8	4	1200	8
3	75	9	5	1500	2
4	90	6	6	1500	10
5	115	7	3	1100	10

C. Confidence Limits for Verification Trials

The predicted value obtained from multiple regression has a variance, $V(\underline{X}_i)$,:[1]

$$V(\underline{X}_i) = [\underline{X}_i^T C \underline{X}_i]\sigma^2$$

Where

C is the inverse matrix used when solving for the multiple regression coefficients (i.e. $C = (X^T X)^{-1}$)

σ^2 is the variance of the prediction, and is estimated by, S^2, the error mean square obtained in the multiple regression ANOVA.

$\underline{X}_i$ is the array of values for the n independent variables which correspond to the "i"th dependent value of the m sets of values used in the multiple regression analysis.

That is

$$\underline{X}_i = [X_{0i}, X_{1i}, X_{2i}, \ldots, X_{ni}] \quad \text{for } i = 1, 2, \ldots, m$$

---

[1] Draper, N. R. and H. Smith, "Applied Regression Analysis," John Wiley and Sons, p. 56, N. Y. 1966.

Therefore,

$$\mathbf{X} = \begin{bmatrix} \underline{X}_1 \\ \underline{X}_2 \\ \vdots \\ \underline{X}_m \end{bmatrix}$$

When using the formula obtained from multiple regression to predict a dependent variable for a new set of independent variables, the variance on this prediction, σ^2_{new}, is

$$\sigma^2_{new} = V(\underline{X}_{new}) + \sigma^2$$

The confidence limits on the prediction when using a new set of values for $\underline{X}$ is therefore

$$t = t_{(predicted,\ new)} \pm t_{(a,\ N-2)} \sqrt{V(\underline{X}_{new}) + S^2}$$

Where $t_{(a,\ N-2)}$ is the Student-t value of 2.326 at $a = 95\%$ and N-2 = 254 degrees of freedom.

D. Results of Verification Trials - Inverted Search

Run Number	Predicted Time	Std. Deviation of Prediction	95% Confidence Interval	Observed Time
1	10.267	4.017	0.333→20.201	10.240
2	5.599	1.435	0.856→10.342	5.104
3	10.577	3.993	0.697→20.457	8.600
4	7.915	3.212	0.000→16.110	7.410
5	3.469	1.388	0.000→ 8.136	3.170

Since the observed time is in the 95% confidence interval in each case, there is no statistical reason to say the prediction is unlikely. However, it should be noted that the variance of the prediction increases outside of the experimental range.

E. Results of Verification Trials - Sequential Search

Run Number	Predicted Time	Std. Deviation of Prediction	95% Confidence Interval	Observed Time
1	83.763	24.026	19.792→147.734	76.800
2	98.276	9.836	59.642→136.910	97.384
3	70.653	21.118	12.368→128.938	65.030
4	94.380	18.356	41.540→147.220	92.020
5	117.510	22.229	57.156→177.864	108.770

Since the observed time is in the 95% confidence interval in each case, there is no statistical reason to say the prediction is unlikely. However, it should be noted that the variance of the prediction increases outside of the experimental range.

X. DISCUSSION OF INTERACTIONS

It is difficult to understand the significance of an interaction, such as OEF, without some intuitive feeling for its meaning. Consequently, the interactions listed in the ANOVA for the multiple regression will be discussed below.

A. Inverted Search

The most significant interaction is OCF. In the multiple regression, the level of this interaction is evaluated by taking the product of the levels of O, C, and F (i.e. OCF = O x C x F).

The product of C (the number of index terms per basic record) and F (the number of records in the basic file) is the total number of entries in the inverted file.

The O factor is 1 when the entire basic record is to be printed out in response to the inquiry and 0 otherwise. The product of O and CF represents the fact that a large inverted file combined with a large amount of printout significantly affects retrieval time.

The next most significant interaction is OEF. EF is the product of E (the number of different index terms in the basic file) and F (the number of basic records in the basic file). This product may be interpreted as follows: Given a particular number of basic records and a uniform distribution of index terms among

them, the number of records having any one particular index term will decrease as E increases. That is, as EF increases, the number of call-up numbers in any given inverted file record will decrease. Therefore, shorter chains of inverted file records must be moved. Also, merging these shorter inverted file records to satisfy the elements of a given inquiry results in fewer "hits". EF in itself is not significant. However, OEF signifies that the time reduction obtained as EF increases is highly accentuated when full printout is desired.

The next most significant interaction is OP, which is the interaction between O (printout consists of full record or call-up number) and P (the number of elements in the inquiry).

As the number of elements in the inquiry increases, there are fewer records in the file which will satisfy all of the elements. Therefore, OP is a measure of the fact that less time is required to print out fewer records.

The interaction PEG is the next most significant interaction. As G (the maximum number of index terms is an inverted record) increases for a fixed level of E (the number of distinct index terms in the basic file) the amount of unused space in the last inverted record (read in for each inquiry element) increases. EG therefore indicates that the use of fixed format records for the inverted file will cause unnecessary data transfer from disk to core. P is

the number of elements in the inquiry, and since the unnecessary data movement represented by EG occurs for each inquiry element, PEG is the total such wasteful motion. Interactions OE, OC, OH, and OF are all positive and reflect the fact that printing out the entire record will increase retrieval time as any of the other factors (E, C, H, F) also increases.

Interactions OCE and OEH are measures of the decreased time required to process and print out responses to inquiries which involved merging relatively short lists.

Interaction PCF, which is negative, indicates that, for a given inverted file containing CF entries, the number of records satisfying all the elements in the inquiry decreases as the number of inquiry elements increases. These fewer responses require less time to process and output.

B. Sequential Search

The most significant factor affecting retrieval time for a sequential search is the AF interaction. The product of A (the number of characters per record) and F (the number of records in the file) is AF (the total number of characters in the file). Since the sequential search involves moving all of every record from disk to core, it is quite reasonable to expect that the total number of characters in the file is the most significant factor.

The next most significant interaction is CD which is the product of C (the number of index terms per record) and D (the number of characters per index term). This interaction is a measure of the number of characters that must be tested when checking each record sequentially for several key field values.

XI. CONCLUSIONS

A. The Formulas

The following formulas for retrieving records from files stored on the disk packs of an IBM System OS/360 using inverted and sequential search techniques were determined and verified experimentally.

1. Inverted Search Formula

$$t = b_0 + \sum_{i=1}^{17} b_i X_i$$

Where t is retrieval time in seconds

And

i	b_i	X_i
0	5.31158	constant term
1	0.03124×10^{-2}	O x C x F
2	-0.05941×10^{-3}	O x E x F
3	-4.04291	O x P
4	0.02052×10^{-2}	P x E x G
5	0.18272	O x E
6	1.66627	O x C
7	0.01949×10^{-2}	E x G
8	-1.96169	O x C x E
9	0.05163×10^{-2}	C x F
10	4.05500	O x H

i	b_i	X_i
11	-0.13813	E
12	0.03442	P x E
13	-0.03961	O x E x H
14	-0.01396×10^{-2}	P x C x F
15	0.39083×10^{-2}	O x F
16	-0.02915	G
17	-5.35582	O

Symbol	Factor	Range
C	Number of index terms in basic record	(5 to 10)
E	Number of different index terms in file	(50 to 100)
F	Number of records in basic file	(1000 to 2000)
G	Max number of index terms in inverted record	(50 to 100)
H	Directories and prime areas on packs	(Diff. = 0, Same = 1)
O	Printout	(Call-up nbr = 0, Full= 1)
P	Number of elements in inquiry	(2 to 3)

2. Sequential Search Formula

$$t = b_0 + \sum_{i=1}^{6} b_i X_i$$

Where

t = retrieval time in seconds.

And

i	b_i	X_i
0	20.97293	constant term
1	10.64587×10^{-4}	A x F
2	-2.02172	C x D
3	-0.03381	F
4	-0.96944	N
5	5.21572	C
6	5.24116	D

Symbol	Factor	Range
A	Number of characters in basic record	(75 to 120)
C	Number of index terms in each record	(5 to 10)
D	Max number of characters per index term	(3 to 6)
F	Number of basic records in file	(1000 to 2000)
N	Number of basic records per block on disk	(2 to 15)

3. Example of Use of Formula

Consider an inverted file system with the following characteristics:

C = Number of index terms in basic record = 8.

E = Number of different index terms in file = 80.

F = Number of records in basic file = 1200.

G = Max number of index terms in inverted record = 60.

H = Directories and prime areas on the same packs (H = 1).

P = Number of elements in the inquiry = 3.

For this data, for example:

$$X_{14} = P \times C \times F = 3 \times 8 \times 1200 = 288 \times 10^2$$

and, therefore

$$b_{14}X_{14} = (-0.01396 \times 10^{-2})\ (288 \times 10^2) = -4.02047$$

To find retrieval time we compute the following:

i	b_i	X_i code	X_i value	b_iX_i
1	0.03123×10^{-2}	OCF	0	0.0
2	-0.05941×10^{-3}	OEF	0	0.0
3	-4.04291	OP	0	0.0
4	0.02052×10^{-2}	PEG	162×10^2	2.95487
5	0.18272	OE	0	0.0
6	1.66627	OC	0	0.0
7	0.01949×10^{-2}	EG	48×10^2	0.93552
8	-1.96169	OCE	0	0.0
9	0.05163×10^{-2}	CF	96×10^2	4.95647
10	4.05500	OH	0	0.0
11	-0.13813	E	80	-11.05030
12	0.03442	PE	240	8.26079
13	-0.03961	OEH	0	0.0
14	-0.01396×10^{-2}	PCF	288×10^2	-4.02047
15	0.39083×10^{-2}	OF	0	0.0
16	-0.02915	G	60	-1.74899
17	-5.35582	O	0	0.0
Total				0.28789

$$t = b_0 + \sum b_i X_i$$

$$t = 5.31158 + 0.28789$$

$$t = 5.59947 \text{ seconds}$$

This predicted time of 5.59947 seconds compares well with the actual observed time of 5.104 seconds.

B. Inverted Versus Sequential Search

Some pertinent statistics for the observed data points are:

Statistic	Inverted Search	Sequential Search
Number of significant independent variables	17	6
Number of dependent variables (retrieval time)	1	1
Number of observations	256	256
Mean value of observations (sec.)	6.551	112.186
Standard deviation of observations (sec.)	3.691	57.093
Maximum observed time	26.024	231.763
Minimum observed time	2.513	29.338

The inverted search was superior to the sequential search throughout the entire range of all variables investigated in this work.

Furthermore, close examination of the analysis of variance for the inverted search multiple regression function reveals that the size of the basic file has no significant effect upon retrieval time for the inverted search. The most significant factor affecting

retrieval time is the total number of entries in the inverted file. From these facts, one can conclude that the inverted file technique is particularly useful for files with a large number of records with only a few number of index terms per record.

C. Utility of the Experimental Approach

In the course of carrying out this four stage experiment, sixty-eight trials (32 for Stage 1, 32 for Stage 2, 4 for Stage 4) were required. Each trial required, on the average, 15 minutes of time on the computer used (an IBM System 360/Mod 50). That is, 17 hours of computer time was required for trials. The analysis of this data required approximately 2 more hours of computer time. Debugging the computer programs required approximately another 5 hours. The total computer time required was therefore 24 hours. At commercial rates of approximately 200 to 400 dollars an hour, this is 4800 to 9600 dollars!

The results obtained in this experiment apply only when the specified conditions are also met. In order to determine a more general model, a larger, more costly experiment must be conducted.

It appears then that only large user corporations, large consulting firms, or large computer manufacturers could bear the cost of such large scale experiments applied to one or more

bench mark problems. However, these same concerns might have a great deal of interest in carrying out just such projects.

The general methodology used in this paper (including the PL/I computer program listed in the appendix) could be used to conduct a more limited experiment which would cost far less than the present experiment, or the larger, more general experiment discussed above. A more limited experiment, yielding limited results, might prove to be of great utility to an engineer designing a particular information retrieval system, which might have a high rate of usage, or a long life over which the experiment costs could be amortized.

XII. RECOMMENDATIONS FOR FURTHER STUDY

This paper has considered only retrieval time for inverted and sequential searches under specified inquiry and index term distribution conditions.

There are, therefore, several avenues for further study related to this thesis, that is:

1) Other search techniques can be studied;

2) A broader system scope could be taken (i.e. consider file maintenance time, file building time, storage costs, etc., as well as retrieval time;

3) Different assumptions can be made for the distribution of index terms among the basic file records and inquiries;

4) Mixed element types for the inquiries (i.e. elements containing other than the "=" relational operator) can be studied, and;

5) Any combination of recommendations 1 thru 4 could be a subject for further investigation.

BIBLIOGRAPHY

Section I Computer System Selection and Bench Mark Problems

1. Herman, D. J., "The Use of a Computer to Evaluate Computers," AFIPS Conference Proceedings, 1964 Spring Joint Computer Conference, Vol. 25, pp. 383-395.

2. Joslin, E. O., "Cost Value Technique for Evaluation of Computer System Proposals," AFIPS Conference Proccedings, 1964 Spring Joint Computer Conference, Vol. 25, pp. 367-382.

3. Gregory, R. H., and R. L. Van Horn, "Automatic Data Processing Systems," Editions 1 and 2, Wadsworth.

4. Gregory, R. H., and R. L. Van Horn, "Business Data Processing and Programming," Wadsworth.

5. Conway, Gibbons, and Watts, "Business Experience with Electronic Computers," Price Waterhouse & Co.

6. Williams, et. al., "A Methodology for Computer Selection Studies," Computers and Automation, May, 1963.

7. Gosden, J. A., "The Computer Chooser's Quandry; Which Machine and Why?," Datamation, December, 1962.

8. Castillo-Fernandez, Jose A., "Enrique Rivera-Santana, "Technique for Evaluating Electronic Computers," Data-Processing, September, 1962.

9. Sisson, R. L., "How to Be A Comparison Shopper for Computers," Business Management Magazine, October, 1962.

10. "Better Computer Comparisons for You," EDP Analyzer, June, 1963.

11. "New Ways for EDP System Studies," EDP Analyzer, September, 1963.

12. Canning, R. G., "Selection Procedure, EDP Systems Analysis Technique, 1962, from Volume I, Standard EDP Reports by Auerbach IBNA.

13. Friedland, E. I., "How to Select the Best Computer: A Conceptual Outline," July, 1963, MITRE Working Paper, W6231.

14. Bagley, P. R., "Data Collection for Evaluation of EDP Proposals," July, 1963, MITRE Working Paper, W6283.

15. "Management of Data Processing Equipment," Air Force Manual 171-9, March, 1962.

16. Rosenthal, S., "Analytical Technique for Automatic Data Processing Equipment Acquisition," AFIPS Conference Proceedings, 1964 Spring Joint Computer Conference, Vol. 25, pp. 359-366.

17. Hillegass, "The Use of Bench Mark Problems," Computers and Automation, January, 1966.

Section II File Organization

18. Baker, F. T., "Some Storage Techniques for Use with Disk Files," Scientific Report No. ISR-4 (Final Report) Information Storage and Retrieval, The Computation Laboratory, Harvard University, Cambridge, Massachusetts, pp. III-1 to III-43.

19. Dzubak, B. J., and C. R. Warburton, "The Organization of Structured Files," Communications of the ACM, Vol. 8, Issue 7, July, 1964, pp. 446-452.

20. Colilla, R. A., and B. H. Sams, "Information Structures for Processing and Retrieving," Communications of the ACM, Vol. 5, Issue I, Jan., 1962, pp. 11-16.

21. Peterson, W. W., "Addressing for Random-Access Storage," IBM Journal of Research and Development, Vol. 1, Issue 2, April, 1957, pp. 130-146.

22. Johnson, L. R., "An Indirect Chaining Method for Addressing on Secondary Keys," Communications of the ACM, Vol. 4, Issue 5, May, 1961, pp. 218-222.

23. Prywes, N. S., et. al., "The Multi-List System Technical Report No. 1," Part 1, Vol. 2, Moore School of Electrical Engineering, Report 62-10, November, 1961.

24. Martin, J., "Programming Real-Time Computer Systems," Prentice Hall, Englewood Cliffs, New Jersey, 1965.

25. Buchholz, W., "File Organization and Addressing," IBM Systems Journal, June, 1963.

26. Prywes, N. S., "The Organization of Files for Command and Control," Moore School of Electrical Engineering, March, 1964.

Section III Evaluation of IR Systems

27. Conger, C. R., "The Simulation and Evaluation of Information Retrieval Systems," HRB-Singer, Inc., State College, Pa., April, 1965.

28. Gagliardi, V. O., "Data File Size and Its Relation to the Bayesian Effectiveness of an Information Retrieval System," Air Force Systems Command - USAF, L. G. Hanscom Field, Bedford, Massachusetts, April, 1965.

29. Heckman, R. P., "A Method for Investigating the Behavior of Attributes Which Belong to Information Storage and Retrieval Systems," A Master's Thesis, Georgia Institute of Technology, August, 1965.

Section IV Related Information Retrieval System Studies

30. Blunt, C. R., "An Information Retrieval System Model," HRB-Singer, Inc., State College, Pa., October, 1965.

31. Chien, R. T., and W. D. Frazer, "An Application of Coding Theory to Document Retrieval," IBM Research Division, RC 1499, November, 1965.

32. Lombardi, L. A., "Theory of Files," Proceedings Eastern Joint Computer Conference, pp. 137-140, December, 1960.

33. Salton, G., "Progress in Automatic Information Retrieval," IEEE Spectrum, 1965.

34. _____ "Information Storage and Retrieval, Scientific Report No. ISR-2, Harvard University, Cambridge, Massachusetts, September, 1962.

Section V Experiment Design

35. Chew, V., ed., "Experimental Designs in Industry," John Wiley & Sons, Inc., New York, 1958.

36. Cochran, W. G., and G. M. Cox, "Experimental Designs," John Wiley & Sons, Inc., New York, 1957.

37. Cox, D. R., "Planning of Experiments," John Wiley & Sons, Inc., New York, 1958.

38. Davies, O. L., ed., "The Design and Analysis of Industrial Experiments," Hafner Publishing Company, New York, 1963.

APPENDIX 1

Details of the Computing System

A. The Computer System Hardware Configuration

The computing system used for this experiment was the IBM System 360/MOD 50 at the Computing Center of the Western Electric Company Engineering Research Center at Princeton, New Jersey. This system may be described as a disk pack oriented system with card, magnetic tape, and paper tape input, and paper tape, magnetic tape, card, and high speed printer output. For this experiment, the 2540 card reader was used as input for the inquiries, and the response for the inquiries was outputted on the 1403 printer. The disk units were used to store the computer programs and the files required for the experiment.

The system had eight 2311 disk drives sharing one 2841 storage control unit which provided a channel to the central processor.

B. Computing System Software

The computer program used to generate and search the files for this experiment was written in PL/I Version 1 Release 1. As this release of PL/I did not yet have random access capability, disk accessing was accomplished by means of basic assembler language subroutines.

The PL/I main program and the two assembly language subroutines were compiled and stored on disk in a private library in the form of an executable module.

The executive routines were provided by the standard IBM System 360/Operating System, Release 7.

APPENDIX 2

Alternative File Organization Techniques for Information Retrieval Systems

Alternative File Organization Techniques for Information Retrieval Systems

Since the time involved in accessing a disk storage device is much greater than core processing time, the effectiveness of any disk-file organization technique will be largely dependent on the following two critical conditions.

(1) If many records must be read from the file, it is far faster if they are stored consecutively and can be read as a track or tracks, and

(2) If records cannot be stored consecutively, then as few records as possible should be read (Ref. 18, pp. III-5).

A. Primary-Key Technique

One way to organize a file to meet these two objectives is to sort the file sequentially on the most frequently requested key-field, which we shall now define as the "primary-key" or "primary index term". For example, suppose a key-field used for department number were the key-field (or index term) most frequently mentioned in an inquiry against a particular file. Then all records in the file would be sorted and then stored in sequence based on department number.

In turn, each group of records having the same department number could then be sorted on the basis of some other index term.

It should be noted that some disk-storage addressing techniques require that each record be stored in sequence on a unique identifier. This poses no problem for the primary key technique, since a two segment key can be used. The first segment would be department number for this example, and the second segment would be an index representing relative position of the record with respect to the first record having the same department number. See Figure 2-1 for an example of a series of records for department numbers 250 and 251 using a two segment key.

2 segment key =	dept-nbr	relative-position
25000	250	00
25001	250	01
.	.	
.	.	
.	.	
25010	250	10
25100	251	00
.	.	
.	.	
.	.	
25120	251	20

Figure 2-1 Illustration of Records with Two-Segment Identifiers

A directory containing primary key field-values (department numbers in this example) and corresponding disk addresses (actual or symbolic) of the first record having that field value must be maintained either in core storage or on some other random access device.

If, upon receipt of an inquiry, a primary key field-value is found to be in the inquiry, the directory is searched for the disk address corresponding to the first record having the given field-value (or index term).

The first record is then read into a core buffer along with the other records in the same block on the disk track. Each record having the same first segment can then be examined in turn to see if it satisfies the remaining key field-values in the inquiry. Additional blocks may be brought into the core buffer area and processed until the last record having the given primary key field-value has been examined. Records satisfying all the inquiry field-values are written out.

It should be noted that this method has met the two objectives stated above when the primary key is in the inquiry. That is, the disk is accessed only a relatively few times, and records, when read in, are read in consecutively.

If an inquiry does not contain a primary key field-value, then the entire file must be searched. This is, of course, a disadvantage of the primary key technique.

B. Chaining on Secondary Keys (or Index Terms)

1. Description

It is often the case that an inquiry may refer to keys other than the "primary key" on which the file is organized.

These other keys may be denoted as "secondary keys". Any efficient scheme to retrieve a record on the basis of secondary key field-values must consider the two restrictions imposed by the relatively large disk access and reading times. That is, for any scheme,

(1) If many records must be read from the file, it is far faster if they are stored consecutively and can be read as a track, and

(2) If records cannot be stored consecutively, then as few records as possible should be read

Since the file is not sequentially organized on secondary keys, it is not possible to achieve efficiency in regard to the first restriction when retrieving on a secondary key. To achieve efficiency by reading as few records as possible requires some sort of scheme for knowing which records in the file have a given secondary key field-value.

One way of doing this is known as chaining on secondary keys. This method may be described as follows. In each record there is a field associated with each secondary key field called a chaining field. This field contains the address (actual or symbolic) of the previous record in the file having the same field-value for that secondary key field. See Figure 2-2 for an illustration of this technique.

Core Table for "Age"

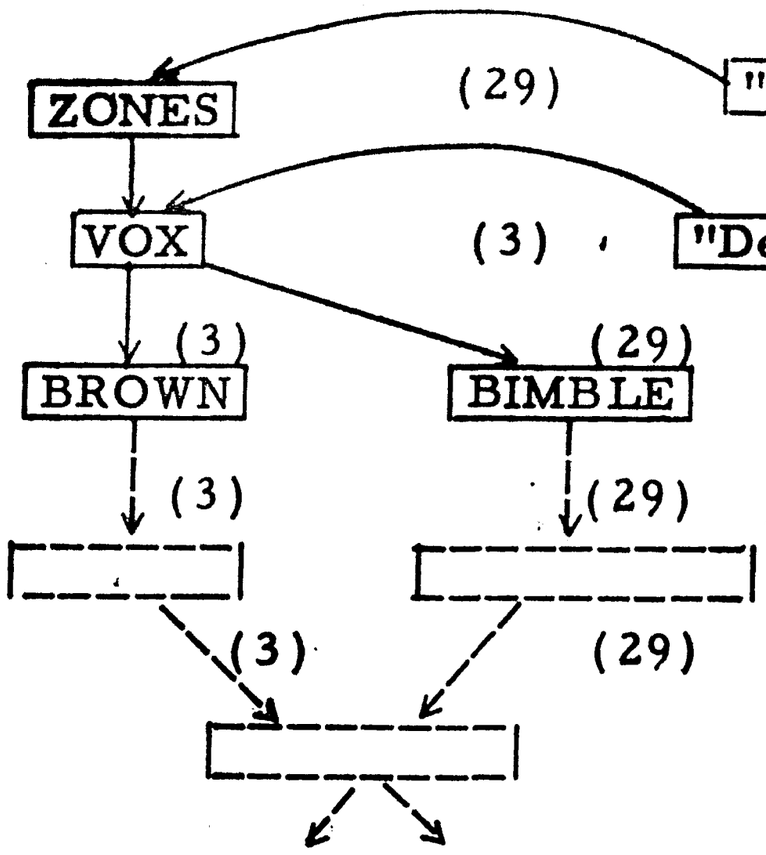
Age	Last Record	No. of Records
20	BROWN	1
21	BALL	1
-	-	-
29	ZONES	3

Core Table for "Number of Dependents"

No. of Dependents	Last Record	No. of Records
0	-	-
1	-	-
2	ZONES	3
3	VOX	2

Primary Key	Age	Chain for Age	Dependents	Chain for Dependents
BALL	21	-	2	-
BIMBLE	29	-	2	BALL
BROWN	20	-	3	-
VOX	29	BIMBLE	3	BROWN
ZONES	29	VOX	2	BIMBLE

Figure 2-2 Secondary Key Chaining



Note: The intersection of the two chains is the response to the inquiry.

Other records, when in file.

Figure 2-3 Secondary Key Chain Intersection

For each secondary key field value, there is a table in core storage (or stored on a random access device) which contains the address (actual or symbolic) of the last record in the file and the total number of records having that field-value. This is known as backward chaining. Forward chaining may also be used.

In Figure 2-2 there is an example of a file in which the records are organized by name (the primary key) which have two secondary keys: age; and number of dependents. There is a table in core storage for each of the secondary keys indicating the primary key of the last record and the total number of records having that secondary key field value. In each record, associated with each secondary key field, is a field indicating the primary key of the previous record having the same secondary key field value.

Suppose now that we have a retrieval request asking for all persons who are 29 years old and have three dependents. We go to the two secondary key tables and note there are 3 records having age 29 and 2 records having 3 dependents. Since we want to retrieve as few records from the disk as possible, we will follow the chain for 3 dependents which has only 2 records. From the core table we see that VOX is the last record having 3 dependents. We then read the VOX record into core and check to see if he is 29 years old. If he were, he would satisfy the inquiry and his name would be printed out.

Since VOX is 29 years old, his name is printed out.

Returning to the chain for 3 dependents, we see that BROWN, is the next record having 3 dependents. We then read the BROWN record into core and check to see if he is 29 years old. He is not and so his name is not printed out. The next address field in the BROWN record associated with 3 dependents is found to be blank. This indicates the end of the chain and the end of the retrieval search for the inquiry.

If both the primary key and several secondary keys are mentioned in an inquiry, it is usually faster to process the search by using the primary key since the records having that primary key are ordered sequentially and can be read in as a track or tracks.

C. Multi-List (Superfield Chaining)

This technique is directly analogous to the secondary key chaining method. The difference between these two methods is that, in the multi-list method, two or more key-field values are concatenated to form a "superfield". These superfield-values are then chained. The purpose of this concatenation is to shorten the length of the chains which must be followed on disk at the expense of building and searching a larger table of superfield-values in core.

APPENDIX 3

Notation for File Structure

A. File Structure

Let $\underline{X}$ denote a complete file stored in a disk storage device. Since disk storage addresses are one-dimensional, consider $\underline{X}$ to be a one-dimensional array (a vector) as illustrated in Figure 1. The elements in this vector are records. There are N records in the file.

$$\underline{X} = X_1, X_2, \ldots\ldots, X_i, \ldots\ldots, X_N$$

Figure 1 A File Considered as a Vector With Records as Elements

Let X_i represent the "i"th record in the file. Each of the records may also be considered a one-dimensional vector as shown in Figure 2. The elements of the record vector are fields. There are n_i fields in the "i"th record.

$$X_i = x_{i1}, x_{i2}, \ldots\ldots, x_{ij}, \ldots\ldots, x_{in_i}$$

Figure 2 A Record Considered as a Vector With Fields as Elements

Each field, in turn may be considered a one-dimensional array whose elements are taken from the set of the letters of the alphabet, the ten digits from 0 to 9, and some specified set of special characters. There are m_{ij} characters in the "ij"th field. See Figure 3.

$$x_{ij} = c_{ij1}, c_{ij2}, \ldots\ldots, c_{ijk}, \ldots\ldots, c_{ijm_{ij}}$$

Figure 3 A Field Considered as a Vector With Characters as Elements

The entire file may also be considered as a one-dimensional array whose elements are fields or characters, although it is not convenient nor instructive to do so, since we are interested, presumably, in selecting certain records from the file on the basis of their contents.

B. Fixed Format, Fixed Length Records (FFFL)

The number of characters in a field may or may not be the same as for other fields in the same record or for corresponding fields in other records. A very common file situation, however, is when corresponding fields in all records have the same number of characters, and all records have the same number of fields. This type of file is known as a file with fixed length, fixed format records. These records may be described as follows: (See Figure 3B).

1. Each record, x_i, has n fields, that is

   $n = n_i \quad i = 1, 2, 3, \ldots, N$

2. The "j"th field in the "i"th record, x_{ij}, has the same number of characters, m_{ij}, as the "j"th field in all other records, that is,

$$m_j = m_{ij} \qquad i = 1, 2, 3, \ldots, N$$

$$j = 1, 2, 3, \ldots, n$$

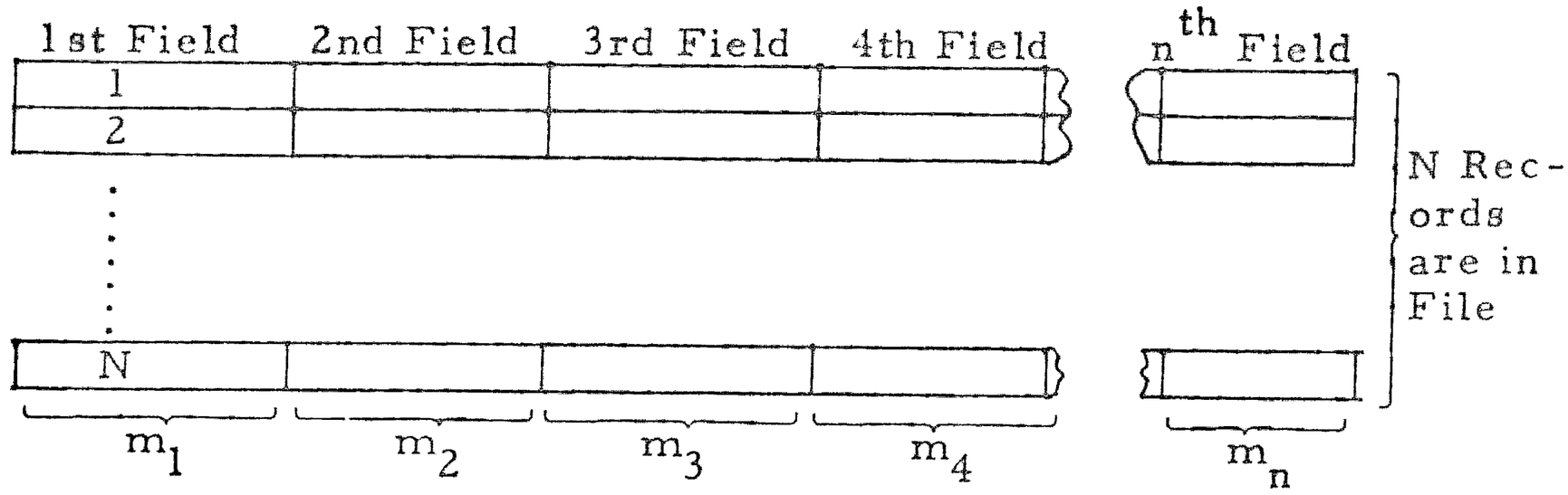


Figure 3B Block Schematic of FFFL Records

APPENDIX 4

Characteristics of Disk Storage Devices

Characteristics of Disk Storage Devices

A. General Description

The physical aspects of disk-storage devices are well known. In a typical disk storage unit there are many disks mounted on a common shaft. Each disk is coated with a magnetic material. The shaft turns on the order of 200-400 revolutions per sec. There may be one reading head for each disk surface used for storage, or there may only be one reading head for the entire unit. In the latter case, the reading head must move vertically from disk to disk as well as in and out across the surface of a given disk. There may be one or more of these units attached to a CPU using one or more individual or shared channels.

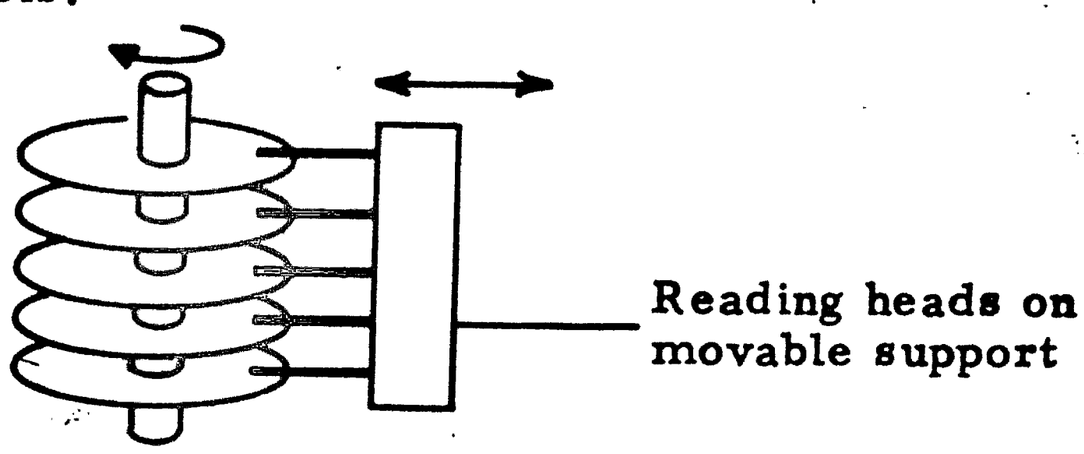


Figure 4-1 Typical Disk Storage Configuration

B. Factors Affecting Information Retrieval

There are two operating characteristics of disk storage devices which are important to any record retrieval scheme.

First of all, the time to position the reading head over a particular disk track is much greater than core processing times.

Therefore, it is highly desirable to minimize the number of times the reading head must be moved.

Secondly, once the head has been positioned, an entire track can be transferred to a core buffer at a relatively high transfer rate. Therefore, if more than one record must be obtained from disk storage, it would be desirable to have the records stored sequentially on the same disk track.

C. Finding a Record on a Disk Device (Addressing)

The problem of how a given record may be identified and consequently retrieved is known as the "addressing problem". There are several techniques for accomplishing this. They are discussed below. The following discussion is based largely on James Martin's work (ref. 24).

1. Sequential Scanning

This method of finding a record is to scan the file, inspecting each record in turn, until some identifier is recognized. However, this is a very time consuming procedure.

2. Direct Addressing

The easiest, and often the most economical way to solve the addressing problem is to know the machine address of the record in question. For example, in some banking applications the account numbers are the disk storage address of that account.

However, direct addressing is often infeasible. For example, employee numbers or inventory part numbers are usually of significance on their own and are not easily changed.

3. Algorithmic Addressing

It may be possible to organize a logical file so that the addresses within that file may be calculated from the reference information.

This method, however, may not be efficient in its use of file space. An airline, for example, may have 150 flight numbers. The algorithm might use these and the date to calculate the file address. However, not every flight flies on every day; hence some of the addresses generated will not contain a record.

4. Table Look-up

A sorted table is often used as an index to a random-access file. It lists the reference number of items (record keys) along with the addresses where they are stored. When this is done, the computer has to search through the table rather than search through the file. A considerable amount of time may be saved, but space is needed to store the table.

There would not normally be one table for all the records in the files but rather primary and secondary tables, or a

whole hierarchy of tables. The primary table would give the location of the secondary table, and this would give the location of the item in the files.

5. Randomizing

This is one of the most common methods of addressing. It locates the majority of items but never all items with one file reference. Hence it is quicker than file scanning or table look-up. However, it does not give a high file density. Seventy to eighty per cent packing is a reasonable figure to aim at.

The first step in this type of addressing is to convert the item's reference number into a random number that lies within the range of the file addresses where the record is to be located.

With this method of addressing there is a probability that the correct record will not be found the first time. A new reference will then have to be made to an overflow location.

Also, the random address could refer to a "pocket" of records which is then scanned to find a particular record.

APPENDIX 5

Experimental Data

A. Program Listings

The following programs were used to obtain the experimental data. The program named P1 is the main program which generates a file and then processes inquiries. This main program is written in PL/I.

There are two sub-programs used for disk accessing written in System 360 Basic Assembler Language. The sub-program called LOADMBY was used to store records on disk, and SRCHMBY was used to retrieve records randomly from the disk.

B. Experimental Data

Before and after processing each inquiry, the PL/I main program accessed the CPU interval timer and displayed this information on the printer. These time indications formed the basis for the data presented below.

```
/* TEST OF INVERTED FILE SYSTEM */

P1:PROCEDURE OPTIONS(MAIN);
  DECLARE 1 INITA, 2 I1 BIT (16), 2 I2 BIT (16),
  1 INITR, 2 I3 BIT (16), 2 I4 BIT (16),
  (N1,N2,N3,N4) FIXED BINARY (16,0);

/* READ IN PARAMETERS FOR FILE STRUCTURE */

  GET LIST (NBR_CHAR_IN_R_KEY,NBR_DIF_KEYVALUES,NBR_ADRS_IN_A_REC,
  NBR_CHAR_IN_FILLER, NAP,NRP,NAI,NRI,NBLKR,NBLKA,
  NBR_CHAR_IN_A_KEY,NBR_R_REC,NVAR,NITR,N1,N2,N3,N4,NSEED);
  NCIT=NBR_CHAR_IN_A_KEY;
  NCR=1B+NBR_CHAR_IN_FILLER+NBR_CHAR_IN_R_KEY+NITR*NCIT;
  I1=N1; I2=N2; I3=N3; I4=N4;

/* WRITE OUT DESCRIPTION OF THIS RUN  */

PUT PAGE LINE (5) EDIT ('TEST PROGRAM FOR CREATING AND SEARCHING AN',
'INVERTED FILE STRUCTURE') (A(43),A(24));
PUT SKIP(2) LIST ('A MASTER''S THESIS PROJECT BY MORRIS YAGUDA');
PUT SKIP LIST ('JUNE,1967');
PUT SKIP(3) LIST('THIS RUN WILL HAVE THE FOLLOWING CHARACTERISTICS');
PUT EDIT ('BASIC RECORD FILE',
'NUMBER OF CHARACTERS IN CALL-UP NUMBER=',NBR_CHAR_IN_R_KEY,
'NUMBER OF CHARACTERS IN RECORD=', NCR,
'TOTAL NUMBER OF RECORDS IN FILE=',NBR_R_REC,
'NUMBER OF INDEX TERMS IN EACH RECORD=',NITR,
'MAXIMUM NUMBER OF CHARACTERS IN EACH INDEX TERM',NCIT,
'NUMBER OF DISTINCT INDEX TERMS IN FILE=',NBR_DIF_KEYVALUES) (
SKIP(2),A,SKIP,6 (X(5),A(60),F(6,0),SKIP));
PUT EDIT ('INVERTED (ASPECT) FILE',
'MAXIMUM NUMBER OF BASIC RECORD CALL-UP NUMBERS IN RECORD=',
NBR_ADRS_IN_A_REC) (SKIP(2),A,SKIP, 2 (X(5),A(60),F(6,0),SKIP));
PUT EDIT ('DISK STORAGE CONSIDERATIONS',
```

```
'INDEX FOR BASIC FILE STORED ON UNIT',NRI,
'INDEX FOR INVERTED FILE STORED ON UNIT',NAI,
'PRIME AREA FOR BASIC FILE STORED ON UNIT',NRP,
'PRIME AREA FOR INVERTED FILE STORED ON UNIT ',NAP,
'NUMBER OF I/O BUFFERS FOR BASIC FILE', N3,
'NUMBER OF I/O BUFFERS FOR INVERTED FILE', N1,
'NUMBER OF BASIC FILE RECORDS PER BLOCK ON DISK=', NBLKR,
'NUMBER OF INVERTED FILE RECORDS PER BLOCK ON DISK=',NBLKA)
(SKIP(2),A,SKIP, 8 (X(5),A(60),F(6,0),SKIP));

/* ENTER P2 BLOCK WHICH WILL DYNAMICALLY ALLOCATE STORAGE
   ACCORDING TO THE PARAMETERS READ IN ABOVE  */

P2:BEGIN;
DECLARE SEQFILE FILE;
DECLARE (BLANK_CHECK,SEARCH_KEY) CHAR (NBR_CHAR_IN_R_KEY),
(IX,IY) FIXED BINARY (31,0),
SRCHKEY CHAR (NBR_CHAR_IN_A_KEY);
DECLARE RCOUNT PICTURE '99999', ITFIL CHAR (NCIT-11B),
RKEY(NITR) PICTURE '999',ACOUNT PICTURE '999',
KEYFLD (NITR),
IKEY(NITR) PICTURE '999',
(OPNAR,CLSAR,PUTA,PUTR,GETA,GETR) CHAR(1),
AD (NBR_DIF_KEYVALUES,NBR_ADRS_IN_A_REC) CHAR (NBR_CHAR_IN_R_KEY),
LAST (NBR_DIF_KEYVALUES) FIXED BINARY (9,0),
1 ADRS_REC, 2 DELCD CHAR(1), 2 ADRS_IDENT CHAR(NBR_CHAR_IN_A_KEY),
2 PREV_ADRS_REC CHAR (NBR_CHAR_IN_A_KEY),
   2 ADRS (NBR_ADRS_IN_A_REC) CHAR (NBR_CHAR_IN_R_KEY),
T1 (NBR_DIF_KEYVALUES) CHAR (NBR_CHAR_IN_R_KEY),
1 DISK_REC, 2 DELCD CHAR (1), 2 DISK_IDENT CHAR (NBR_CHAR_IN_R_KEY),
2 RIT(NITR) CHAR(NCIT), 2 RFIL (NBR_CHAR_IN_FILLER) CHAR (1);

/* INITIALIZE WORK AREAS */

ITFIL='0000000000';
RFIL='#';
```

```
OPNAR='3'; CLSAR='4'; PUTA='1'; PUTR='2'; GETA='1'; GETR='2';
BLANK_CHECK='                ';  T1=BLANK_CHECK;
DISK_REC.DELCD=' ';
IX=NSEED;
AD=BLANK_CHECK; LAST=0B;
ACOUNT=1;
RCOUNT=1;
CALL LOADMBY (OPNAR);
OPEN FILE (SEQFILE) OUTPUT;

/* GENERATE RECORDS, ASSIGNING KEYS RANDOMLY AMONG RECORDS */

GEN_RCDS: DO I=0B TO NBR_R_REC-1;
KEY_GEN: DO J=1B TO NITR;
CALL RANGEN (IX,IY,YFL);
IX=IY;
RKEY(J)=YFL*NVAR+1B+(J-1B)*NVAR;
RIT(J)=ITFIL||RKEY(J);
END KEY_GEN;
RCOUNT=I;
DISK_IDENT=RCOUNT;

/* ENTER DISK_IDENT IN APPROPRIATE INVERTED FILE RECORDS */

LOOP1: DO J=1B TO NITR; CALL ADDADRS; END LOOP1;
ADDADRS:PROCEDURE;
K2=RKEY(J);
LAST(K2)=LAST(K2)+1B; K1=LAST(K2);
AD(K2,K1)=DISK_IDENT; IF LAST (K2)=NBR_ADRS_IN_A_REC THEN
B1:BEGIN;LAST(K2)=0B; ADRS_IDENT=ACOUNT;
   ADRS=AD(K2,*);PREV_ADRS_REC=T1(K2); ACOUNT=ACOUNT+1B;
   AD(K2,*)=BLANK_CHECK; T1(K2)=ADRS_IDENT;
   CALL LOADMBY (PUTA,ADRS_REC);
   END B1;
END ADDADRS;
```

```
/* WRITE BASIC RECORD ON DISK */

 CALL LOADMBY (PUTR,DISK_REC);
 PUT FILE (SEQFILE) EDIT (DISK_REC) (A(1),A(NBR_CHAR_IN_R_KEY),(NITR)
 A(NCIT),(NBR_CHAR_IN_FILLER) A(1));
 END GEN_RCDS;

/* WRITE INVERTED FILE RECORDS FROM WORKING STORAGE */

 WRITE_RCDS: DO I=1B TO NBR_DIF_KEYVALUES;
 IF AD(I,1)=BLANK_CHECK THEN GO TO EX1;
 ADRS_IDENT=ACOUNT;ADRS=AD(I,*);PREV_ADRS_REC=T1(I);ACOUNT=ACOUNT+1B;
 CALL LOADMBY (PUTA,ADRS_REC);
 T1(I)=ADRS_IDENT; EX1:END WRITE_RCDS;
 CLOSE FILE (SEQFILE);
 CALL LOADMBY (CLSAR);

/* BEGIN INVERTED FILE SEARCH */

 CALL SRCHMBY (OPNAR,INITA,INITR);
 LIMIT=NBR_ADRS_IN_A_REC +1B;

 /* REPEAT THIS SECTION OF THE SEARCH PROGRAM FOUR TIMES, ONCE FOR
    EACH COMBINATION OF THE TWO OUTPUT FORMATS AND THE TWO CHOICES
    FOR THE NUMBER OF INDEX TERMS IN EACH INQUIRY  */

 REPEAT_SECTION: DO KK=1B TO 100B;
 PUT PAGE LIST ('BEGIN TIMING INVERTED SEARCH: TIMER=', TIME);

/* ACCEPT INQUIRY AND PRINT IT OUT */

 START: GET LIST (INBR,KEY_CHECK);
 GET LIST ((KEYFLD(J),IKEY(J) DO J=1B TO KEY_CHECK));
 IF IKEY(1)=999 THEN GO TO EXIT_RTN;
 PUT EDIT ('INQUIRY NBR',INBR,' ASKS FOR RECORDS WITH ',
 ('KEY FIELD NBR ',KEYFLD(J),'=', ITFIL||IKEY(J) DO J=1B TO KEY_CHECK
```

```
)) (SKIP(2),A(11),F(5,0),A(23),(KEY_CHECK) (A(14),F(3,0),A(1),
A(NCIT+1B)));
PUT EDIT ('TIMER=    ',TIME) (SKIP,2 A(9));
PUT EDIT ('THESE RECORDS ARE ') (SKIP,A(18));

/* CLEAR WORKING STORAGE AREAS */

CLEAR_WS: DO J=1B TO NBR_DIF_KEYVALUES; AD(J,*)=BLANK_CHECK;END
CLEAR_WS;
K=1;

/* GET AND LOAD APPROPRIATE INVERTED FILE RECORDS IN SEARCH AREA */

MAX_NBR=KEY_CHECK;
SRCH1: DO I=1B TO MAX_NBR; KDUM=IKEY(I); SRCHKEY=T1(KDUM);
IF SRCHKEY=BLANK_CHECK THEN GO TO START;
GET_ADRS_REC: CALL SRCHMBY (GETA,SRCHKEY,ADRS_REC);
AD(K,*)=ADRS;
IF PREV_ADRS_REC=BLANK_CHECK  THEN B2: BEGIN;
IF K>10110B THEN GO TO MERGE_LISTS;
   K11=K; K=10111B; GO TO EX2; END B2;
SRCHKEY=PREV_ADRS_REC;
K=K+1B; GO TO GET_ADRS_REC;

/* MERGE LISTS FOR THE TWO KEY VALUES BEING MATCHED */

MERGE_LISTS:IEMP2=101101B; K44=1B; K22=K; IEMP1=1B; K33=1B;
M1: IF AD(K22,K44)=BLANK_CHECK THEN GO TO START1;
   IF AD(K11,K33)=BLANK_CHECK THEN GO TO START1;
IF AD(K11,K33) = AD(K22,K44) THEN GO TO M2;
   IF AD(K11,K33) <AD(K22,K44) THEN B4: BEGIN; K33=K33+1B;
      GO TO M3; END B4; K44=K44+1B; GO TO M4;

/* AT END OF SEARCH, RETRIEVE RESPONSES TO INQUIRY */

M2: IF KEY_CHECK>2 THEN GO TO M5; IF (KK=1B)|(KK=11B) THEN PUT EDIT
```

```
(AD(K11,K33))(A(15)); ELSE B5: BEGIN; SEARCH_KEY=AD(K11,K33);
   CALL SRCHMBY (GETR,SEARCH_KEY,DISK_REC);
   PUT EDIT(DISK_REC)(SKIP,A(1),A(NBR_CHAR_IN_R_KEY),(NITR) A(NCIT),
   (NBR_CHAR_IN_FILLER) A(1)); K44=K44+1B;K33=K33+1B;
   GO TO M4; END B5;

/* LOAD INTERSECTION OF LISTS INTO TEMPORARY STORAGE */

M5: AD(IEMP2,IEMP1)=AD(K11,K33); IEMP1=IEMP1+1B; K44=K44+1B;
K33=K33+1B;
IF IEMP1 ¬= LIMIT THEN GO TO M4; IEMP2=IEMP2+1B; IEMP1=1;
M4: IF K44¬=LIMIT THEN GO TO M3; K22=K22-1B; K44=1B;
IF K22=10110B THEN GO TO START1;
M3: IF K33 ¬=LIMIT THEN GO TO M1; K11=K11-1B;
K33=1B; IF K11 =0B THEN GO TO START1; GO TO M1;
START1: K11=1B; IF KEY_CHECK<3 THEN GO TO START; IEMP1=1B;

/* CLEAR SEARCH WORKING STORAGE AREA */

CLEAR_IT: DO J=1B TO 101100B;
   AD(J,*)=BLANK_CHECK; END CLEAR_IT;
K11=1B;

/* LOAD TEMPORARY STORAGE INTO SEARCH AREA */

LOAD_TEMP: IF AD(IEMP2,01)=BLANK_CHECK THEN GO TO T2;
   AD(K11,*)=AD(IEMP2,*);
T2: IEMP2=IEMP2-1B; IF IEMP2=101100B THEN GO TO CLEAR_TEMP;
   K11=K11+1B; GO TO LOAD_TEMP;

/* CLEAR TEMPORARY STORAGE AREA */

CLEAR_TEMP: DO J=101101B TO 110010B; AD(J,*)=BLANK_CHECK;
   END CLEAR_TEMP;
   T3: KEY_CHECK=KEY_CHECK-1B; K=10111B;
EX2: END SRCH1;
```

```
   GO TO START;

   /* AFTER LAST INQUIRY, WRITE OUT THE TIME */

   EXIT_RTN:PUT SKIP LIST ('END INVERTED SEARCH:  TIMER=', TIME);

/* BEGIN SEQUENTIAL SEARCH */

   PUT PAGE LIST ('BEGIN SEQUENTIAL SEARCH:  TIMER=', TIME);

/* ACCEPT INQUIRY AND PRINT IT OUT */

   STARTSEQ: GET LIST(INBR,KEY_CHECK);
   GET LIST ((KEYFLD(J),IKEY(J) DO J=1B TO KEY_CHECK));
   IF IKEY(1)=999 THEN GO TO EXIT_RTN2;
   PUT EDIT('INQUIRY NBR',INBR,' ASKS FOR RECORDS WITH ',
   ('KEY FIELD NBR ',KEYFLD(J),'=', ITFIL||IKEY(J) DO J=1B TO KEY_CHECK
   ))(SKIP(2),A(11),F(5,0),A(23),(KEY_CHECK) (A(14),F(3,0),A(1),
   A(NCIT+1B)));
   PUT EDIT ('TIMER=    ',TIME)(SKIP,2 A(9));
   PUT EDIT('THESE RECORDS ARE ')(SKIP,A(18));

/* RETRIEVE AND CHECK EACH RECORD IN FILE IN SEQUENCE */

   SRCH2: DO I=1B TO NBR_R_REC; GET FILE (SEQFILE) EDIT (DISK_REC)
   (A(1),A(NBR_CHAR_IN_R_KEY),(NITR) A(NCIT), (NBR_CHAR_IN_FILLER)
   A(1));
   CHECK_LOOP: DO J=1B TO KEY_CHECK; IF IKEY(J)¬=RIT(KEYFLD(J))
   THEN GO TO SEQ_EX; END CHECK_LOOP;
   IF (KK=1B)|(KK=11B) THEN PUT EDIT(DISK_IDENT) (A(15));
   ELSE
      PUT EDIT(DISK_REC) (SKIP,A(1),A(NBR_CHAR_IN_R_KEY),(NITR) A(NCIT),
   (NBR_CHAR_IN_FILLER) A(1)); SEQ_EX: END SRCH2;
   CLOSE FILE (SEQFILE);
   GO TO STARTSEQ;
   EXIT_RTN2: PUT SKIP LIST ('END SEQUENTIAL SEARCH:  TIMER=',TIME);
```

```
 END REPEAT_SECTION;
 CALL SRCHMBY (CLSAR);

/* RANDOM NUMBER GENERATOR PROCEDURE */

(NOFIXEDOVERFLOW):RANGEN:PROCEDURE (IX,IY,YFL);
 IY=IX*10000000000000011B;
 IF IY<0B THEN IY=IY+11111111111111111111111111111111B+1B;
 YFL=IY/11111111111111111111111111111111B; RETURN; END RANGEN;
 END P2;
 END P1;
```

```
LOADMBY START 0
 ENTRY LOADIS
LOADIS SAVE (14,12)
 BALR 8,0
 USING *,8,9,10
LOAD L 9,=A(LOAD+4096)
 L 10,=A(LOAD+8192)
 ST 13,SAVEBLK+4
 LA 13,SAVEBLK
 L 2,0(1)
 L 3,0(2)
 CLI 0(3),X'F1'
 BC 8,PUTA
 CLI 0(3),X'F2'
 BC 8,PUTR
 CLI 0(3),X'F3'
 BC 8,OPNAR
 CLI 0(3),X'F4'
 BC 8,CLSAR
 WTO 'NO BRANCH'
 BC 15,RETURN
RETURN L 13,SAVEBLK+4
 RETURN (14,12)
CLSAR CLOSE (RFILE)
 CLOSE (AFILE)
   FREEPOOL AFILE
 FREEPOOL RFILE
 BC 15,RETURN
OPNAR OPEN (RFILE,(OUTPUT))
 OPEN (AFILE,(OUTPUT))
 BC 15,RETURN
PUTA L 4,4(1)
 L 0,0(4)
 PUT AFILE,(0)
 BC 15,RETURN
```

```
PUTR L 4,4(1)
 L 0,0(4)
 PUT RFILE,(0)
BCR BC 15,RETURN
 USING IHADCB,5
ERRORA LA 5,AFILE
 B ERROR
ERRORR LA 5,RFILE
 B ERROR
ERROR TM DCBEXCD2,X'E0'
 BC 8,SOME1
 TM DCBEXCD2,X'C0'
 BC 8,E2
 TM DCBEXCD2,X'80'
 BC 8,E1
 WTO 'SEQ CHK'
 B SOME1
SOME1 TM DCBEXCD1,X'24'
 BC 8,RETURN
 TM DCBEXCD1,X'20'
 BC 8,E5
 WTO 'NO SPACE'
 B RETURN
E2 WTO 'CLOSE ERROR'
 B SOME1
E1 WTO 'DUP RCD'
 B SOME1
E5 WTO 'UNCOR ERROR'
 B CLSAR
AFILE DCB DSORG=IS,MACRF=(PM),DDNAME=AFILE,OPTCD=MY,RECFM=FB,RKP=1,     1
                NTM=40,CYLOFL=1,BFALN=F,SYNAD=ERRORA
RFILE DCB DSORG=IS,MACRF=(PM),DDNAME=RFILE,OPTCD=MY,RECFM=FB,RKP=1,     1
                NTM=40,CYLOFL=1,BFALN=F,SYNAD=ERRORR
 DS 0D
SAVEBLK DS 18F
 DCBD DSORG=IS,DEVD=DA
```

END LOADIS

```
SRCHMBY START 0
 ENTRY SRCHDS
SRCHDS SAVE (14,12)
 BALR 8,0
 USING *,8,9,10
LOAD L 9,=A(LOAD+4096)
 L 10,=A(LOAD+8192)
 ST 13,SAVEBLK+4
 LA 13,SAVEBLK
 L 2,0(1)
 L 3,0(2)
 CLI 0(3),X'F1'
 BC 8,GETA
 CLI 0(3),X'F2'
 BC 8,GETR
 CLI 0(3),X'F3'
 BC 8,OPNAR
 CLI 0(3),X'F4'
 BC 8,CLSAR
 WTO 'NO BRANCH'
 BC 15,RETURN
GETA LM 5,6,4(1)
 L 0,0(5)
 SETL AFILE,K,(0)
 L 0,0(6)
 GET AFILE,(0)
 ESETL AFILE
 BC 15,RETURN
GETR LM 5,6,4(1)
 L 0,0(5)
 SETL RFILE,K,(0)
 L 0,0(6)
 GET RFILE,(0)
 ESETL RFILE
 BC 15,RETURN
```

```
OPNAR LM 5,6,4(1)
 L 2,0(5)
 L 0,0(2)
   GETPOOL AFILE,(0)
 L 2,0(6)
 L 0,0(2)
   GETPOOL RFILE,(0)
 OPEN (AFILE,(INPUT))
 OPEN (RFILE,(INPUT))
 BC 15,RETURN
CLSAR CLOSE (AFILE)
 CLOSE (RFILE)
 BC 15,RETURN
RETURN L 13,SAVEBLK+4
 RETURN (14,12)
 USING IHADCB,5
ERRORR LA 5,RFILE
 B ERROR
ERRORA LA 5,AFILE
 B ERROR
ERROR TM DCBEXCD1,X'9B'
 BC 8,SOME1
 B SOME2
SOME1 TM DCBEXCD1,X'9A'
 BC 8,E7
 TM DCBEXCD1,X'98'
 BC 8,E6
 TM DCBEXCD1,X'90'
 BC 8,E4
 TM DCBEXCD1,X'80'
 BC 8,E3
 WTO 'LL NOT FD'
 B SOME2
E3 WTO 'INVAL RQST'
 B SOME2
E6 WTO 'INP NOT RCHD'
```

```
 B SOME2
E7 WTO 'OUT NT RCHD'
 B SOME2
E4 WTO 'UNCOR ERROR'
 B SOME2
SOME2 TM DCBEXCD2,X'30'
 BC 8,RETURN
 TM DCBEXCD2,X'10'
 BC 1,RETURN
 WTO 'CLOSE ERROR'
 B CLSAR
AFILE DCB DSORG=IS,MACRF=(GM,SK),DDNAME=AFILE,BFALN=F,
                EODAD=RETURN,SYNAD=ERRORA                          1
RFILE DCB DSORG=IS,MACRF=(GM,SK),DDNAME=RFILE,BFALN=F,
                EODAD=RETURN,SYNAD=ERRORR                          1
 DS 0D
SAVEBLK DS 18F
 DCBD DSORG=IS,DEVD=DA
 END SRCHDS
;
```

DATA FOR STAGE 1 INVERTED SEARCH

TREATMENT OF INQUIRY FACTORS O AND P (HORIZONTAL)
TREATMENT OF FILE GENERATION FACTORS A TO N (VERTICAL)

	P	OP	(1)	O
(1)	3.74	4.05	2.18	3.56
	3.39	3.42	1.86	3.86
ABCDEFGHJKLMN	6.16	6.29	4.10	9.14
	6.55	6.59	4.49	7.55
ADJMN	7.05	8.49	3.93	10.65
	6.78	8.10	3.64	10.13
BCEFGHKL	7.39	7.73	4.81	5.70
	7.25	7.73	4.89	6.52
CDFGHN	6.92	6.89	4.55	5.82
	6.85	6.83	4.56	6.15
ABEJKLM	6.96	8.00	4.22	11.53
	7.03	7.25	4.35	10.80
ACFGHJM	4.83	5.46	3.06	5.01
	4.94	5.25	3.23	7.14
BDEKLN	6.78	6.80	4.48	5.11
	0.43	6.94	4.49	4.91
FGJKLN	10.88	17.68	6.65	39.13
	10.22	17.43	6.39	39.98
ABCDEHM	5.57	5.57	3.79	6.70
	5.86	6.23	3.81	6.01
ADFGKLM	5.75	6.07	3.35	6.33
	5.66	6.47	3.38	8.07
BCEHJN	6.62	7.03	4.36	7.57
	6.14	6.69	3.98	6.09
CDHJKL	3.59	3.58	2.19	4.32
	3.56	3.56	2.00	4.47
ABEFGMN	7.14	7.18	4.68	4.91
	7.14	7.32	4.59	5.74

ACHKLMN	6.64	7.43	3.81	10.48
	6.82	8.34	3.82	11.73
BDEFGJ	3.87	3.82	2.56	2.84
	3.91	3.89	2.59	3.19
EGHJKM	6.57	8.90	4.11	17.22
	6.46	10.26	4.14	19.73
ABCDFLN	9.36	10.86	6.39	10.16
	8.97	8.97	5.92	10.32
ADEGHKN	5.86	6.52	3.39	6.76
	5.54	5.51	3.06	5.11
BCFJLM	7.35	7.26	4.74	6.42
	7.02	7.02	4.59	5.66
CDEFJKMN	6.82	7.90	3.78	8.58
	6.42	7.62	3.46	8.15
ABGHL	4.03	4.04	2.51	2.89
	4.11	4.14	2.81	3.38
ACEFK	4.04	4.10	2.52	2.98
	4.12	4.20	2.82	3.46
BDGHJLMN	6.87	7.43	3.99	4.86
	6.52	7.19	3.63	5.70
EFHLMN	7.41	7.44	4.55	4.84
	7.34	7.27	4.60	4.94
ABCDGJK	8.14	11.64	5.37	20.80
	8.07	11.68	5.08	19.15
ADEFHJL	2.24	5.77	2.23	5.73
	2.43	4.82	2.46	4.65
BCGKMN	5.78	7.47	3.52	11.31
	6.04	8.12	3.66	13.24
CDEGLM	4.87	4.89	3.17	4.33
	4.63	4.86	3.20	5.17
ABFHJKN	5.44	5.72	3.01	5.92
	5.69	5.95	2.97	5.83
ACEGJLN	8.60	8.93	5.73	8.61
	8.75	9.42	5.93	11.63
BDFHKM	8.55	11.48	5.03	26.02
	8.40	13.11	5.01	24.50

DATA FOR STAGE 1 SEQUENTIAL SEARCH

TREATMENT OF INQUIRY FACTORS O AND P (HORIZONTAL)
TREATMENT OF FILE GENERATION FACTORS A TO N (VERTICAL)

	P	OP	(1)	O
(1)	77.54	77.65	77.48	77.97
ABCDEFGHJKLMN	77.54	77.48	77.53	78.34
	228.64	228.82	228.58	230.59
ADJMN	228.72	228.63	228.64	230.13
	54.14	54.41	53.97	56.16
BCEFGHKL	53.95	54.41	54.03	56.18
	214.95	215.16	215.01	215.50
CDFGHN	215.05	215.23	215.00	215.52
	99.91	99.97	99.98	100.32
ABEJKLM	100.00	99.99	99.91	100.32
	104.00	104.18	103.53	104.86
ACFGHJM	104.10	104.13	103.52	104.86
	77.88	78.09	77.92	78.32
BDEKLN	77.98	77.98	77.91	79.07
	77.48	77.45	77.49	77.67
FGJKLN	77.50	77.61	77.51	77.86
	228.69	230.78	228.76	236.55
ABCDEHM	228.75	230.89	228.83	235.66
	53.86	53.93	53.93	54.33
ADFGKLM	53.97	54.04	53.98	54.54
	215.10	215.24	215.11	216.21
BCEHJN	215.27	215.57	215.26	196.22
	57.12	57.19	57.05	57.55
CDHJKL	57.14	57.21	57.14	57.61
	99.85	100.10	99.92	100.84
ABEFGMN	99.82	99.93	99.84	100.85
	103.36	103.41	103.35	103.40
	103.64	103.55	103.42	104.00

ACHKLMN	77.91	78.25	77.90	80.35
	77.91	78.58	77.96	81.02
BDEFGJ	66.23	66.19	66.19	66.32
	66.21	66.31	66.25	66.65
EGHJKM	101.43	102.22	101.45	105.01
	101.57	102.62	101.56	98.02
ABCDFLN	104.92	105.13	104.99	105.81
	104.96	104.97	104.87	106.13
ADEGHKN	107.04	107.31	107.06	108.03
	107.00	107.03	106.98	107.81
BCFJLM	29.50	29.45	29.42	29.42
	29.40	29.33	29.33	29.30
CDEFJKMN	193.85	194.24	193.86	195.68
	193.79	194.25	193.74	195.82
ABGHL	47.29	47.27	47.31	47.32
	47.32	47.32	47.32	47.62
ACEFK	94.59	94.59	94.64	94.64
	94.64	94.65	94.66	95.22
BDGHJLMN	131.07	131.18	130.97	131.00
	131.02	131.08	130.86	131.06
EFHLMN	100.95	100.95	100.95	101.03
	100.97	100.97	100.98	101.08
ABCDGJK	105.59	106.54	105.55	109.65
	105.60	106.48	105.46	109.31
ADEFHJL	107.30	108.43	107.12	108.48
	107.22	108.00	107.15	108.25
BCGKMN	29.97	30.18	29.90	30.80
	30.07	30.28	29.97	31.35
CDEGLM	193.43	193.45	193.34	193.74
	193.43	193.57	193.47	194.31
ABFHJKN	49.08	49.17	49.07	49.57
	49.07	49.14	49.09	49.60
ACEGJLN	132.45	132.58	132.45	133.05
	132.49	132.56	132.42	133.24
BDFHKM	58.73	58.99	58.68	60.14
	58.58	58.85	58.46	60.35

DATA FOR STAGE 2 INVERTED SEARCH

TREATMENT OF INQUIRY FACTORS O AND P (HORIZONTAL)
TREATMENT OF FILE GENERATION FACTORS A TO N (VERTICAL)

	P		OP		(1)		O	
(1)		3.59		3.58		2.19		4.32
C		3.56		3.56		2.00		4.47
E		4.99		6.39		3.01		9.16
CE		5.21		6.12		3.21		9.70
F		4.03		4.04		2.51		2.89
CF		4.11		4.14		2.81		3.38
EF		4.53		5.17		3.11		5.56
CEF		4.30		4.26		2.85		4.36
G		4.96		5.67		2.98		7.30
CG		4.74		5.18		2.96		6.46
EG		8.14		11.64		5.37		20.80
CEG		8.07		11.68		5.08		19.15
FG		4.87		5.13		3.21		3.42
CFG		4.75		6.88		3.08		5.28
		6.16		6.29		4.10		9.14
		6.55		6.59		4.49		7.55
		5.17		5.04		2.76		3.84
		5.12		5.61		2.78		4.14
		7.05		8.49		3.93		10.65
		6.78		8.10		3.64		10.13
		6.68		6.63		4.34		5.02
		6.58		6.54		4.37		5.01
		7.35		7.26		4.74		6.42
		7.02		7.02		4.59		5.66
		6.82		7.90		3.78		8.58
		6.42		7.62		3.46		8.15
		8.61		10.33		5.15		19.19
		8.47		10.51		5.04		17.68

EFG	7.39	7.73	4.81	5.70
	7.25	7.73	4.89	6.52
CEFG	8.43	8.63	5.66	9.28
	8.38	8.77	5.56	9.12
H	3.42	3.68	1.95	4.07
	4.01	4.49	2.39	5.89
CH	6.57	8.90	4.11	17.22
	6.46	10.26	4.14	19.73
EH	3.81	3.81	2.45	2.87
	3.82	3.86	2.62	3.25
CEH	4.83	5.46	3.06	5.01
	4.94	5.25	3.23	7.14
FH	6.96	8.00	4.22	11.53
	7.03	7.25	4.35	10.80
CFH	8.72	11.51	5.52	22.09
	8.33	11.80	5.31	20.04
EFH	3.87	3.82	2.56	2.84
	3.91	3.89	2.59	3.19
CEFH	6.53	6.97	4.25	7.78
	6.41	7.05	4.33	10.32
GH	5.86	6.52	3.39	6.76
	5.54	5.51	3.06	5.11
CGH	6.38	7.63	3.76	13.67
	6.13	7.77	3.46	11.27
EGH	6.92	6.89	4.55	5.82
	6.85	6.83	4.56	6.15
CEGH	7.03	7.01	4.61	6.44
	7.39	7.61	4.81	6.84
FGH	6.80	7.26	3.87	9.55
	6.09	6.09	3.40	8.02
CFGH	8.55	11.48	5.03	26.02
	8.40	13.11	5.01	24.50
EFGH	7.21	7.20	4.59	5.30
	7.40	7.37	4.84	5.58
CEFGH	8.60	8.93	5.73	8.61
	8.75	9.42	5.93	11.63

DATA FOR STAGE 2 SEQUENTIAL SEARCH

TREATMENT OF INQUIRY FACTORS O AND P (HORIZONTAL)
TREATMENT OF FILE GENERATION FACTORS A TO N (VERTICAL)

	P	OP	(1)	O
(1)	77.48	77.45	77.53	77.67
	77.55	77.61	77.56	77.86
A	107.09	107.31	107.18	108.03
	107.11	107.03	107.13	107.81
C	65.43	65.64	65.24	67.64
	65.31	65.71	65.25	67.65
AC	114.36	115.24	114.30	118.49
	114.32	115.45	114.33	118.68
D	64.59	64.60	64.50	65.04
	64.51	64.44	64.46	65.36
AD	102.40	102.58	102.15	103.24
	102.40	102.45	102.13	103.35
CD	29.60	29.45	29.50	29.50
	29.57	29.33	29.49	29.55
ACD	78.00	78.09	78.01	78.32
	77.95	77.98	77.99	79.07
F	151.92	152.10	151.91	152.67
	151.96	152.08	151.85	153.57
AF	231.76	231.86	231.59	233.17
	231.11	231.33	231.03	233.66
CF	106.13	106.54	106.14	109.65
	105.66	106.48	105.62	109.31
ACF	228.80	228.82	228.57	230.59
	228.71	228.63	228.65	230.13
DF	104.14	104.18	103.69	104.86
	104.07	104.13	103.60	104.64
ADF	193.94	194.24	194.00	195.68
	193.92	194.25	193.93	195.82

CDF	77.66	78.16	77.68	79.81
	77.72	78.07	77.65	79.74
ACDF	155.53	156.51	155.48	160.59
	155.44	156.28	155.49	160.64
N	68.39	68.34	68.32	68.82
	68.41	68.52	68.35	69.04
AN	114.08	114.18	114.07	115.01
	114.05	114.08	114.03	115.08
CN	53.94	53.93	54.00	54.33
	53.93	54.04	54.00	54.54
ACN	101.54	102.22	101.53	102.22
	101.50	102.62	101.52	102.20
DN	47.30	47.27	47.37	47.32
	47.36	47.32	47.38	47.62
ADN	99.97	99.97	100.09	100.32
	99.98	99.99	100.05	100.32
CDN	30.37	30.61	30.26	31.51
	30.21	30.37	30.24	31.17
ACDN	75.92	76.26	75.93	78.10
	75.93	76.29	75.86	78.17
FN	66.29	66.19	66.25	66.32
	66.30	66.31	66.26	66.65
AFN	215.02	215.16	215.07	215.50
	215.02	215.23	215.08	215.52
CFN	115.64	116.27	115.49	118.67
	115.80	116.67	115.62	119.64
ACFN	206.98	207.97	207.00	214.29
	206.96	208.33	206.84	214.36
DFN	105.79	105.91	105.76	106.36
	105.63	105.75	105.70	106.43
ADFN	196.93	197.08	196.83	198.53
	196.98	197.19	196.99	198.55
CDFN	57.17	57.19	57.10	57.55
	57.19	57.21	57.12	57.61
ACDFN	132.50	132.58	132.50	133.05
	132.52	132.56	132.52	133.24

VITA

PERSONAL HISTORY

Name:	Morris Boris Yaguda
Date of Birth:	November 27, 1936
Place of Birth:	Philadelphia, Pennsylvania
Parents:	Gussie S. and Frank Z.
Wife:	Karen Roxena
Children:	Mata Gussie

EDUCATIONAL BACKGROUND

Camden High School 1951 - 1954

Cornell University
Bachelor of Mechanical
Engineering 1954 - 1959

Lehigh University
Candidate for Master
of Science in Industrial
Engineering 1965 - 1967

PROFESSIONAL EXPERIENCE

Western Electric Company, Inc.
New York, New York, Systems Equipment Engineer
October, 1962 to July, 1964.

Denver, Colorado, Systems Equipment Engineer
July, 1964 to July, 1965

Princeton, New Jersey, Research Engineer,
July, 1965 to June, 1967