A MODEL FOR THE EVALUATION OF LARGE SCALE DATA RETRIEVAL SYSTEMS

Kenneth F. Siler

University of California
Graduate School of Management

INTRODUCTION

One effect of our complex society is its need for more information which is the foundation of effective decision-making. However, information is generated by selecting and extracting data which is subsequently condensed and displayed. During the past decade this process has been termed data retrieval and its computer counterpart is called a <u>data retrieval system</u>.

While data retrieval systems come in many different forms (e.g., financial reporting systems, library systems, etc.), they all share one characteristic: the selecting, extracting, and displaying of information from a large computer-stored data bank (the data base). Figure 1 indicates the major components of such a system. The system user controls the content of the query, but the data retrieval system controls the structure of the query, the index, and the data base.

Behind any data retrieval system (and assumed here) is a system which maintains the data base. As new data is collected and as old data is corrected or deleted, the data base must be updated to reflect the changes. This paper neglects any maintenance of the data base and focuses upon the retrieval process. Such a focus is useful and reasonable for two reasons: first, the isolation of the retrieval process can provide much insight into the global problem, and secondly, there are many data retrieval systems where update is insignificant.

Data retrieval research usually focuses upon the file organization concepts of the system. A file organization structures a data base with list processing control elements and with indexes or directories that make the search of the data base efficient. The model described in this paper provides a methodology for evaluating large scale data retrieval systems by measuring the retrieval work load associated with a set of user queries, a specified file organization, and a given data base.

Retrieval workload can be measured in many ways depending upon the computer involved. However, data retrieval systems are typically input-output bound, i.e., the critical path in their processing is the input and output of data from on-line devices to the computer. Hence, it is most appropriate to measure retrieval workload in terms of access frame movements, disk rotations, and response time. <u>Access frame movements</u> refer to the inward and outward movement of the read/write heads on a disk storage device. Because of the high cost of core storage and drum storage, most data retrieval systems store their data bases and indexes on disk storage devices such as the IBM 2314. These devices must physically move their read/write heads to positions where the selected data is stored. <u>Disk rotations</u> refer to the number of rotations required to retrieve data from tracks of a secondary storage device once the read/write head is properly positioned. These rotations account for the latency to find the data and to transfer it. <u>Response time</u> refers to time (seconds) that a computer requires to retrieve, analyze, and display the information corresponding to a query.

In this paper response time is computed from access frame movements, A, and disk rotations, R, using the formula:

$$T = .075\ A + .025\ R \quad \text{(in seconds)}$$

This response time, T, is the average response time neglecting any effects of sharing the computer with other users.

Previous research into data retrieval systems has typically used small data bases with "real" data base records. The term, "real" denotes the use of the original data base records or some truncated form of the original records. Research with "real" records naturally provides very accurate measures of system performance. However, the cost of such research tends to be prohibitive. The model described by this paper does not use "real" data base records. Instead, an abstracted form of the data base is used. The data base abstraction is based on important data base statistics which give a summary of the data base. Due to this statistical representation, the model is called a stochastic model. Rather than examining each data base record, the stochastic model computes measures of system performance by analyzing the statistics of the data base. This stochastic methodology is applied to several file organizations. By analyzing the results, the performance of various file organizations can be measured against a query stream and a data base.

With this stochastic methodology, one can conduct research into file organization techniques and, therefore, into data retrieval systems with a significant reduction of research costs. Hence the objectives for such a model are:

1. To provide a generalized model to represent and study most data retrieval systems.
2. To provide flexibility in modifying the parameters of the data retrieval systems.
3. To provide statistical results and model behavior which can lead directly to choice of design parameters.
4. To minimize the cost of the research.

COMPUTER SIMULATION MODEL

The simulation model is written in PL/I (on an IBM 360/91 computer) for the following reasons, among others: (1) wide acceptance as a general programming language, (2) dynamic allocation of storage, (3) list processing features, (4) half-word binary storage of variables and (5) flexibility and ease of programming. Actual experience indicates that the model operates very efficiently. In simulating 200 queries and 150 cylinders of disk storage with 300 records per cylinder, the model requires 23 cpu seconds for queries with no conjunctions, 31 cpu seconds for queries with one conjunction, and 38 cpu seconds for queries with two conjunctions. The number of conjunctions in a query is a measure of a query's complexity which is defined more precisely later.

Figure 2 depicts the model as a driver program which has as inputs (1) the query stream, (2) the index specification, and (3) the data base. The outputs are a sequence of reports which specify the performance of the model. These elements provide the entire environment of the data retrieval system. The model is stochastic because of the statistical representation of the data base which is created by a separate PL/I program.

The stochastic model is a Monte Carlo simulation. That is, the driver program measures performance by a sampling procedure, it does not control and specify the state of the retrieval system over time. Conceptually, the stochastic model is measuring the workload on a data retrieval system due to a query or set of queries. By passing a sufficiently large representative subset of queries through the stochastic model, one can measure the effect of design strategies.

QUERY REPRESENTATION

The first input component of the model is the query. Mendelson [6] explains how all queries can be represented by either of two canonical formulations. The first is the conjunctive normal form which consists of one or more conjuncts (and's), each of which is a disjunction (or's) of one or more keys, e.g., (A v B) A (C v D), A, A v B, (A v B v C) A (C v D). Here a <u>key</u> is a data item consisting of an attribute and a value by which data base records can be retrieved. In the above examples the letters represent the keys and symbols "A" and "v" represent "and" and "or" respectively.

A verbal translation of the first example is: "give me all employees that are system analysts (A) or programmers (B) and have 10 years (C) or 11 years (D) of experience." In the conjunctive formulation the number of terms that are connected by and's determines the complexity of the query. The examples given above have complexities of 2, 1, 1, and 2, respectively. The second canonical form given by Mendelson is the disjunctive normal form which consists of one or more disjuncts, each of which is a conjunction of one or more keys, e.g., (A B) v (C B), A, A B, (A B C) v (C D).

Most real query streams ask for unique keys, sequences of keys (key-ranges), negations of keys, and conjunctions. Due to this, the stochastic model defines a query as a modification of the conjunctive normal form. Each query is constructed by conjuncting a sequence of key-units where the logical relationship of each key-unit is either affirmation or negation. Here a <u>key-unit</u> is a generic term referring to a single key or to a sequence of keys. In this way complete generality and flexibility in representing queries is possible. A typical query of the simulation might be:

```
QRY = 1       CPX = 3;
IDX = 2       FROM = 40      THRU = 40;
IDX = 1       FROM = 230     THRU = 310;
IDX = -5      FROM = 2       THRU = 2;
```

This query could be translated as: "find all employees with a job title = programmer and a weekly salary ≥ = $240 and ≤ = $310 and a marital status ≥ = married. Here index numbers are changed to index names (IDX = 2 means "job title"), key values to code names (FROM = 40 THRU = 40 means "programmer") or salary dollars, and logical relations to either affirmation or negation (a negative index, e.g., -5 means "not"). Note that the query has a complexity of 3.

The model allows queries to be read as input or created internally (randomly). By inputting specific queries, the researcher can duplicate an actual query stream from historical data. Two uses can be made of such input. First, the correctness of the simulation model can be validated by duplicating a real data retrieval system. Secondly, one can evaluate a hypothetical data retrieval system with a known query stream. Both uses represent important research activities.

When queries are generated internally, the queries and their key-units are randomly generated according to one of several distributions. While almost any distribution can be generated, two very useful distributions, the uniform distribution and a form of the Zipf distribution [12], are implemented. The Zipf distribution (Figure 3) is a popular form of representing natural language text and there is some empirical justification of its use in data retrieval systems [9]. Using random generated queries, a researcher can study many hypothetical situations which may affect the appropriate choice of a file organization.

INDEX REPRESENTATION

The second input component of the model is the index. The purpose of the index is to associate the queries with the data base itself. In the stochastic model, the indexes do not contain any of the actual data base pointers as would be found in an actual data retrieval system. However, sufficient information is represented in the index to be able to calculate the characteristics of an actual data retrieval system.

Query indexing comprises of several steps. First, the key(s) of each key-unit in the query must be located in an index or directory. Secondly, once it is located, the set of accession pointers to the data base must be retrieved. These accession pointers indicate where records with a given key are located in the data base. Accession pointers are retrieved from data in the index itself or in another file called an Accession File. If data from an index is not accession pointer, then it is a pointer to the Accession File which contains the pointers. Finally, all of the accession pointers are disjuncted and conjuncted according to query specifications. The resulting accession list shows how to access the data base and retrieve the records that satisfy the query. These steps are diagrammed in Figure 4.

The model assumes that all indexes to the data base are organized under the index sequential access method (ISAM) [2,3,4] and augmented by an Accession File under certain file organizations.

ISAM is a secondary storage access method that is currently implemented on most computer systems. It provides a method of locating a query key by using an index of keys. The stochastic model assumes that each index of the data retrieval system is organized under ISAM. That is, keys associated with job title, keys associated with weekly salary, and keys associated with marital status have separate ISAM files that form the basis of their indexing strategies. Also, the model assumes that the Accession File is a direct access file that contains accession pointers to the data base under certain file organizations. Two such file organizations that require an Accession File are the inverted list system and the cellular list system. These are described in more detail later.

Figure 5 shows how the indexes are represented in the model. Each index is described by the number of keys in its index and each key is described by a file organization technique (defined later), the number of records containing the key, and the number of buckets (defined later) containing the key. Under the assumption given above, the stochastic model calculates the number of access frame movements, disk rotations, and amount of data transferred from the secondary storage devices. These calculations are dependent upon factors such as:

1. length of keys in each index
2. number of keys in each index
3. blocking factors of the files
4. percent of keys in ISAM overflow
5. location of ISAM overflow
6. number of master indexes in each ISAM file
7. frequency of each key in the data base.

With this information one can estimate the retrieval work load required for query indexing. Most formulas used in the calculations are based on the IBM implementation of ISAM. Other formulas depend upon the file organization techniques that are described later.

DATA BASE REPRESENTATION

The third input component of the model is the data base. A data base is a collection of records. Typically these records will contain information on one area of interest, e.g., personnel of a corporation. The accession list from query indexing points to records in the data base that satisfy the query. The stochastic model assumes that the data base resides on a secondary storage device such as a disk.

If one defines a bucket to be those records that can be accessed with the access frame fixed in one position, then the data base is broken into many storage areas called buckets. An IBM 2314 disk has 200 buckets on its disk pack, namely, its 200 cylinders. The importance of the bucket representing a data base is that access frame movements are very time consuming steps during data retrieval. After access frame movements, the next most time consuming steps are disk rotations. Information on the records within each bucket of a data base provides the basis for calculating rotations.

Figure 6 shows the bucket representation. The data base itself is only a collection of these buckets. One representation of a bucket can actually represent several buckets. As indicated in Figure 5 the probability distribution remains the same. However, the number of records in each bucket is recorded. This can save unnecessary calculations.

All keys of the indexes have their frequencies or probabilities represented for each bucket in the data base. In order to calculate access frame movements and disk rotations, the method of accessing the records in the bucket and the number of records in the bucket that satisfy the query must be known. Query indexing answers the first problem; to answer the second problem, one must make an assumption about the data base records, namely, statistical independence of the keys. Since the buckets representation only gives the frequency or probability of each key in a bucket, the stochastic model assumes independence of the keys and calculates how many records from the bucket satisfy the query as:

$$M = N^* P(K_1)^* P(K_2)^* \ldots {}^*P(K_c)$$

Where M = records from the bucket satisfying the query
N = number of records in the bucket
P(K) = Probability of the query key(s) for each key-unit of the query

As an example suppose one is analyzing the query: "return all records with SEX = MALE and with AGE ≥ = 25 and ≤ = 30." Further suppose that a bucket has the MALE key with a frequency of 300 and the AGE 25 through 30 keys with a total frequency of 100. If there are 500 records in the bucket, then the stochastic model estimates the number of records returned by the query for this bucket as:

$$M = 500^* (300/500)^* (100/500)$$
$$M = 60 \text{ records}$$

One final comment on the data base representation. While the inverted list system retrieves only those records that satisfy a query, other file organizations such as the threaded list system may retrieve more records. Results from the query indexing phase of the simulation show when this occurs and also modifies the record calculations.

FILE ORGANIZATION TECHNIQUES

The model is easily modified to simulate most file organization techniques. Currently, it is programmed to simulate combinations of the threaded list system (TLS), the inverted list system (TLS), and the cellular list system (CIS) which are described in Figure 7. All of these systems assume that the basic indexes are organized as ISAM files. Once a key is found in a directory, the particular mechanics of the file organization technique govern the data retrieval.

The threaded list system is a file organization technique that chains together all of the data base records that contain the same key. Data in the key table, i.e., the index, provides the key value, a pointer to the first data base record with the key, and the length of the key chain. This file organization is known also as knotted lists [11] and Multilist [7, 8].

The inverted list system is a file organization technique where for each key a list of all data base records with the key are created and stored in a separate Accession file. ILS has no chaining. Its principal advantage over TLS is that it retrieves only records satisfying a query even when the query is complex. CDMS [1] and DS-2 [10] are examples of inverted list systems.

The last file organization technique mentioned is the cullular list system. This file organization technique is a combination of TLS and ILS. It has an Accession file which contains pointers to each bucket of the data base where at least one record contains a particular key. Each pointer in the Accession file locates the first data base record in the bucket containing the particular key. Other records in the bucket which contain the key are chained to the first record as in TLS.

Rather than detailing these file organization techniques, the reader can refer to Lefkovitz [5] for further details. The stochastic simulation model simulates information retrieval systems of any size where the file organization technique used is TLS, ILS, or CIS, or any hybrid combination of these.

MODEL VALIDATION

Validation constitutes a major hurdle if no numerical data exists for an actual system. Since the stochastic model measures the retrieval workload associated with a query, it is most difficult to establish the quantitative congruence of the model with reality. Thus far, no real data retrieval system has been located with statistics that measure retrieval workload of a query. What has been found is many systems that measure usage of particular data base records. However, these measures are not associated with any queries.

To establish some basis of validation an alternative strategy was devised. First, another computer model was developed which computed the same retrieval workload but for a "real" representation of the data base. "Real" is written with quote marks to designate that non-pertinent information of each record has been truncated and the remaining keys have been transformed to the integen keys used by the model. Using this model, each data base record is examined to see which queries it satisfied. After accumulating these statistics one can accurately measure the retrieval workload and compare the results with the stochastic model. All three file organization techniques were validated in this way. The "Chi-square Goodness of Fit Test" and "Kolmogorov-Smirnov Test" were applied and their measure indicated a close similarity of the "real" and stochastic models.

Table 1 summarizes these results where the data base and query stream correspond to Zipf's distribution. Each of the file organizations tested included 30 queries with a complexity of two and 30 buckets each with 300 records. The Chi-square statistics for the 30 buckets have 29 degrees of freedom. However, due to cell size, the query statistics had to be aggregated and, hence, reduce the degrees of freedom. The final degrees of freedom are noted within the parentheses of Table 1.

In the second part of the strategy, one retrieval measure was compared with an actual data retrieval system. The statistics came from the Commercial Data Management System (CDMS) used by Lear Siegler, Incorporated. Two months of data were collected on the use of one of their data bases. Although CDMS statistics preclude any hope of measuring accesses and rotations, the system does provide the number of data base records that are returned for each query. This statistic provides at least one point of external validation. Again Chi-square and Kolmogorov-Smirnov tests indicated a close similarity of the CDMS system and the stochastic model. Table 2 summarizes this validation.

The alternative strategy provides some validation of the model. While it does not provide complete validation, the absence of pathological cases during validation further supports the acceptance of the model. Based on recent information, several new data retrieval systems anticipate the collection of pertinent validation data. When such data becomes available, the stochastic model can be validated further.

OPERATING PROCEDURES

The stochastic model is easily adapted to existing data bases and to hypothetical data bases. Three steps are required to operate the model.

First, the queries must be formulated. If the queries already exist, then a simple translation of real queries into coded queries is all that is necessary. This translation is described earlier in this paper under "Query Representation." One only needs to know the code value for each index and the code values for the keys in the index. If one desires to generate the queries internally by random processes, then he specifies the appropriate distributions and the queries are automatically generated.

Secondly, the file organization technique is specified. This is done by selecting one of the three systems (TLS, ILS, or CLS) for each key or group of keys.

Finally, the data base must be formulated as a set of buckets. Figure 8 shows how this is done using an existing data base. Typically, the translation process of Figure 8 is achieved by writing a small program which extracts the keys from data base records. These keys are translated and their frequencies accumulated for each bucket, thus forming the stochastic data base representation.

These three steps define the basic operating procedures for the stochastic model. Output of the model indicates its performance. By varying indexing strategies, query streams, or data bases, one can choose a data retrieval system which most appropriately satisfies the user.

APPLICATION AND RESULTS

The stochastic model is designed to measure the retrieval workload of most data retrieval systems. Hence, if a user can neglect the effect of file updating, the stochastic model can be used effectively to compare alternative data retrieval systems and select the most appropriate one. Thus far the model has been applied to several areas of research including the effect of query complexity on the design of large scale data retrieval systems.

Figure 9 shows a typical application of the model. Three graphs are presented, each graph portraying a different performance measure. Vertical axes give the values of the performance measures and horizontal axes give the level of data base inversion. Data base inversion refers to the number of indexes of a data base that are organized under ILS. In this way, level A is a fully inverted data base and level D has no indexes under ILS, but instead, they are organized under either TLS or CLS. Furthermore, each graph shows queries with complexity of 1, 2, or 3. While it is not the purpose of this paper to analyze the information portrayed by Figure 9, one such analysis shows the usefulness of the model. Under all three performance measures a pure ILS organization is superior to any alternative TLS or CLS organization when a query's complexity is greater than one. This suggests that users with complex queries should use data retrieval systems based on the inverted list system. Results of this type are very significant and demonstrate the usefulness of the model.

It is anticipated that the stochastic model and derivatives of the model will facilitate extended research in data retrieval systems and eventually into generalized data base management systems. Results of this research will help to establish the nature of future generalized data base management systems.

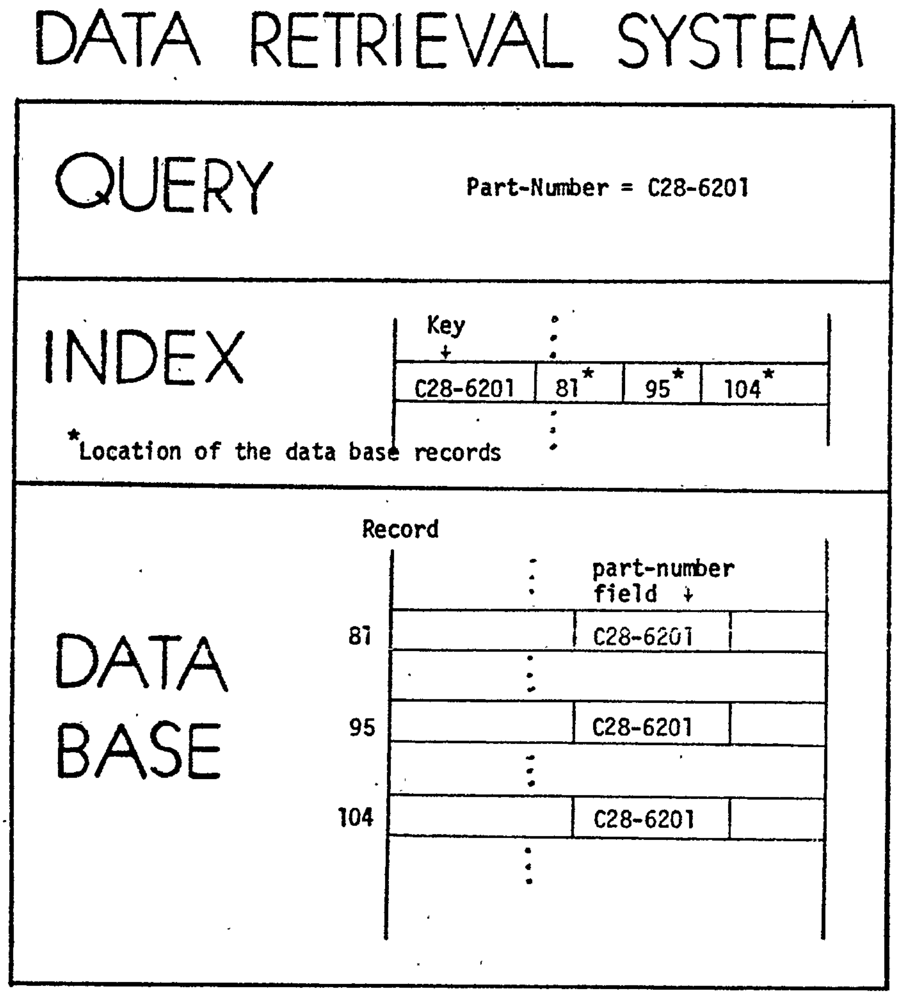


Figure 1. Vertical Description of a Data Retrieval System

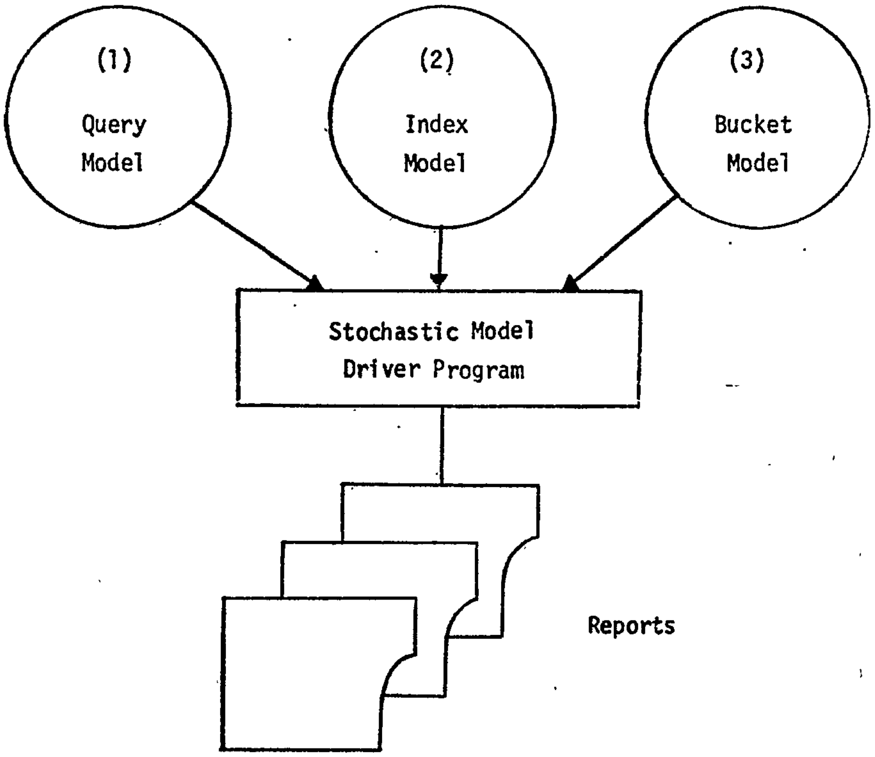


Figure 2. Configuration of the Stochastic Simulation Model

$P(k) = C/k \qquad k = 1, 2, 3, \ldots, N$

where:

N = number of keys in the index,

C = constant, such that

$$\sum_{k=1}^{N} P(k) = 1 \quad \text{and}$$

$C = .3361238$ for $N = 10$

$C = .1925921$ for $N = 100$

$C = .1335835$ for $N = 1000$

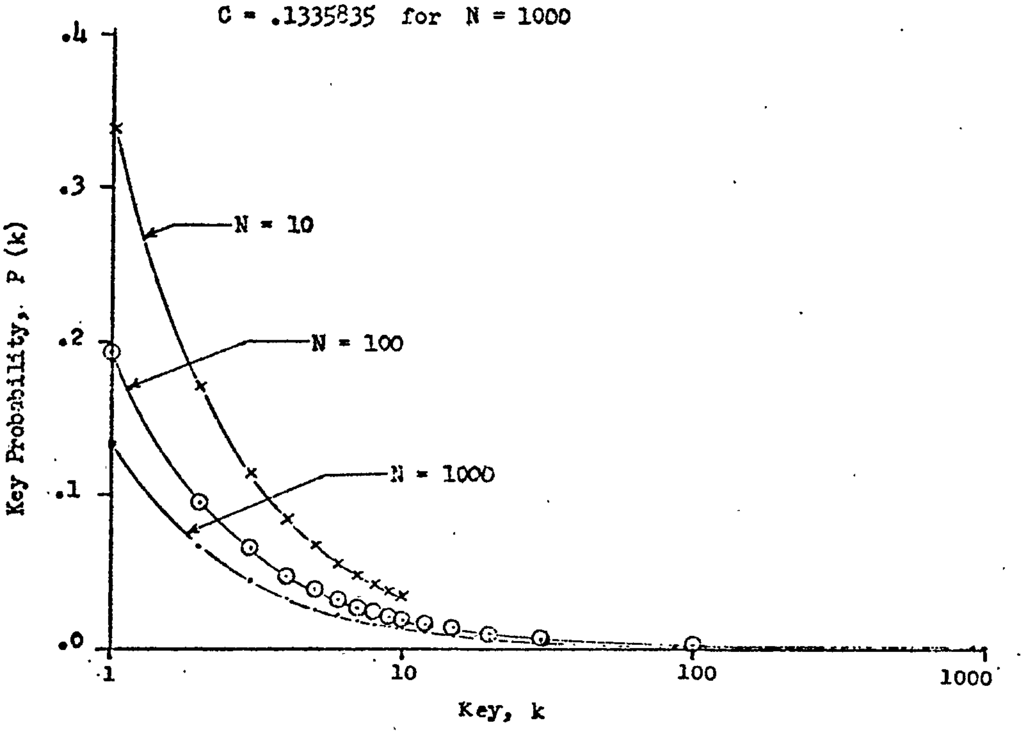


Figure 3

ZIPF's DISTRIBUTION

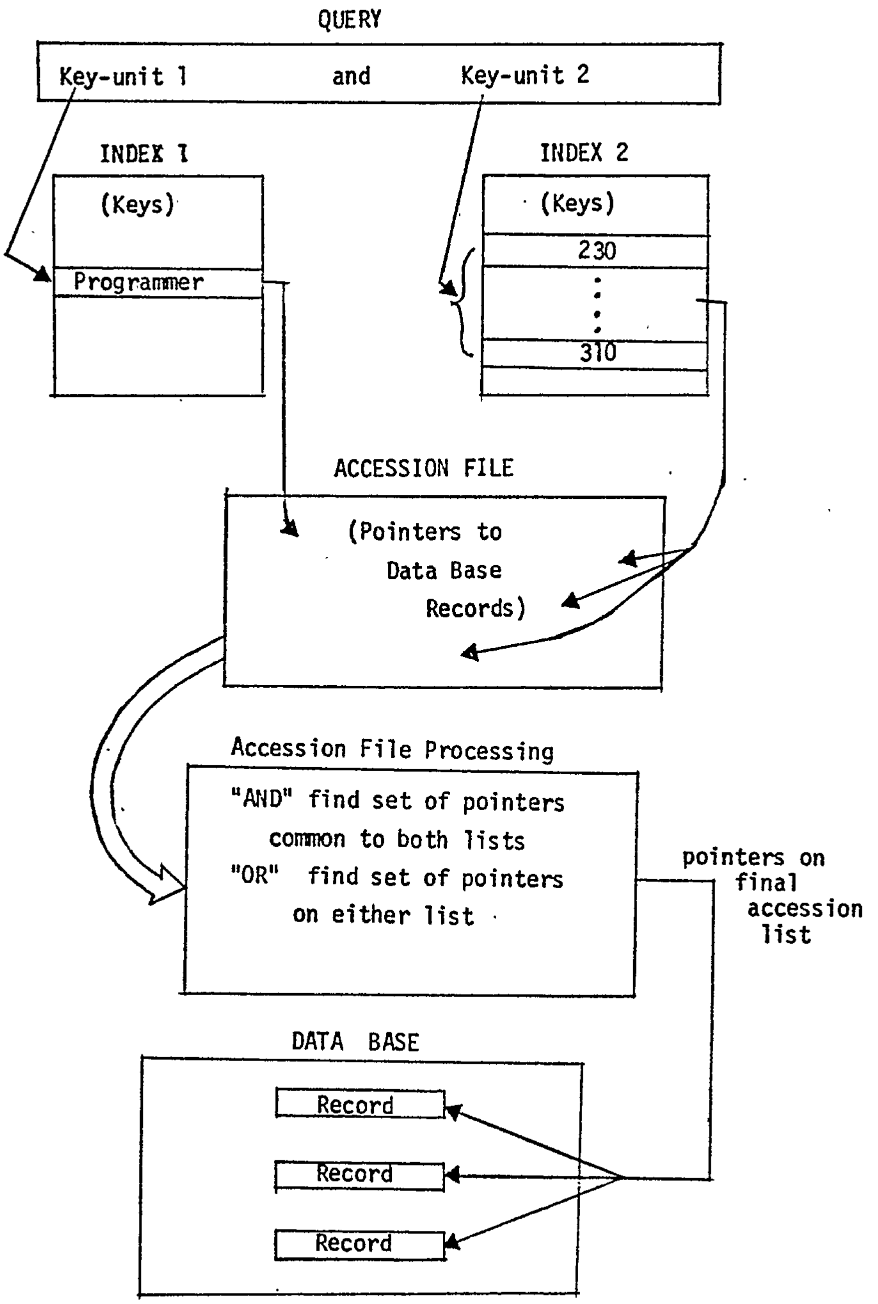
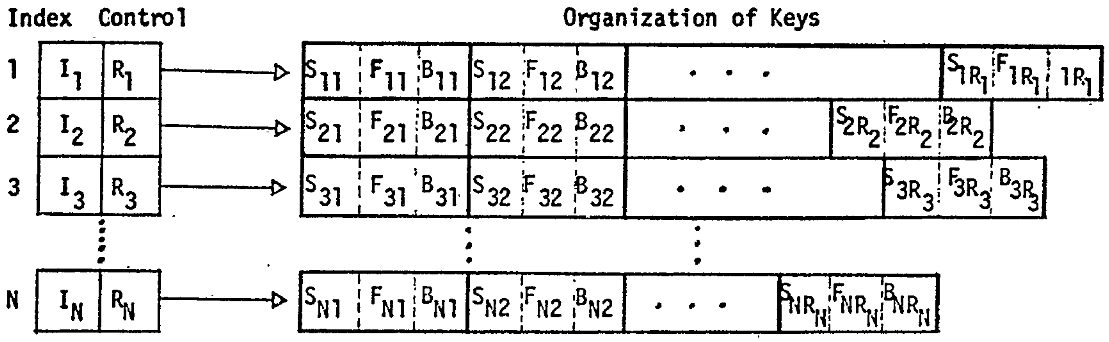


Figure 4. Query Indexing using an Accession File



where:

I_i is the pointer to the key organization of the i^{th} index,

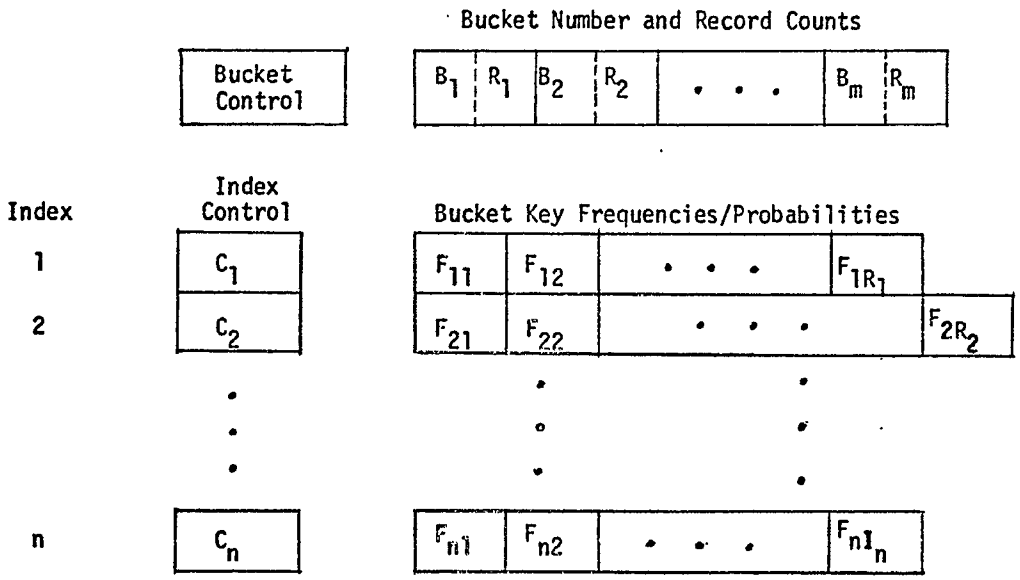
R_i is the number of keys in the i^{th} index,

S_{ik} is the key organization of the k^{th} key of the i^{th} index,

F_{ik} is the data base frequency of the k^{th} key of the i^{th} index,

B_{ik} is the bucket frequency of the k^{th} key of the i^{th} index.

Figure 5. Index Configuration



Where:

B_i	is a bucket number
R_i	is a record count of the associated bucket
C_i	is a control element for the i^{th} index of bucket frequencies
F_{i_j}	is a frequency or probability of the j^{th} key of the i^{th} index
	Bucket Control and Index Controls are necessary elements that specify how to read the bucket information

Figure 6. Bucket Configuration

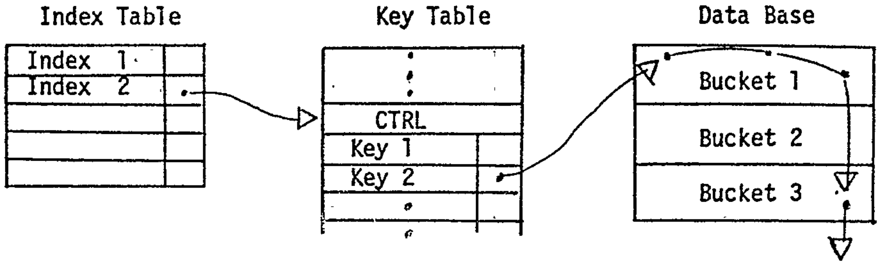


Figure 7a.

Indexing Scheme of Threaded List System

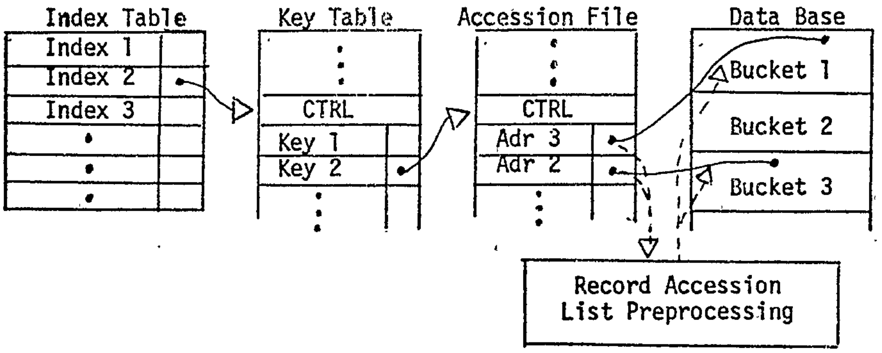


Figure 7b.

Indexing Scheme of Inverted List System

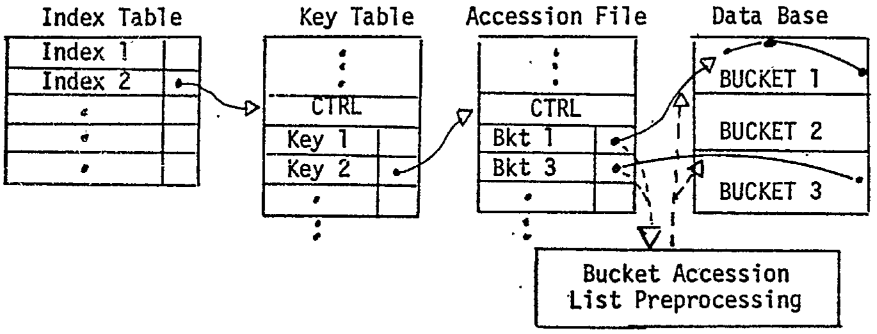


Figure 7c.

Indexing Scheme of Cullular List System

TABLE 1
VERIFICATION OF RECORDS RETURNED

System	Chi-square Tests*		Kolmogorov - Smirnov Test	
	Bucket	Query	Bucket	Query
ILS	14.931 (29)	4.136 (11)	.021	.011
CLS	12.754 (29)	2.371 (5)	.018	.025
TLS	12.324 (29)	4.161 (10)	.016	.008

Chi-square statistics at .05 level of significance:

5 degrees of freedom = 11.070
10 degrees of freedom = 18.307
11 degrees of freedom = 19.675
29 degrees of freedom = 42.577

Kolmogorov-Smirnov critical difference statistics with a = .05 and n = 892 is Da = .045

TABLE 2
SUMMARY OF VALIDATION RESULTS

	Validation Test 1	Validation Test 2
Actual Queries	10	31
Queries after aggregation (due to Chi-square test)	7	11
Simulation Buckets	1	1
Actual Records Returned	501	2787
Simulated Records Returned	499	2911
Chi-square statistic Value at .05 level of significance	2.913 12.592 (6)*	11.909 18.307 (10)*
Kolmogorov - Smirnov statistic Critical Difference at .05 level of significance	.006 .061	.009 .026

*Numbers in parentheses refer to the degrees of freedom

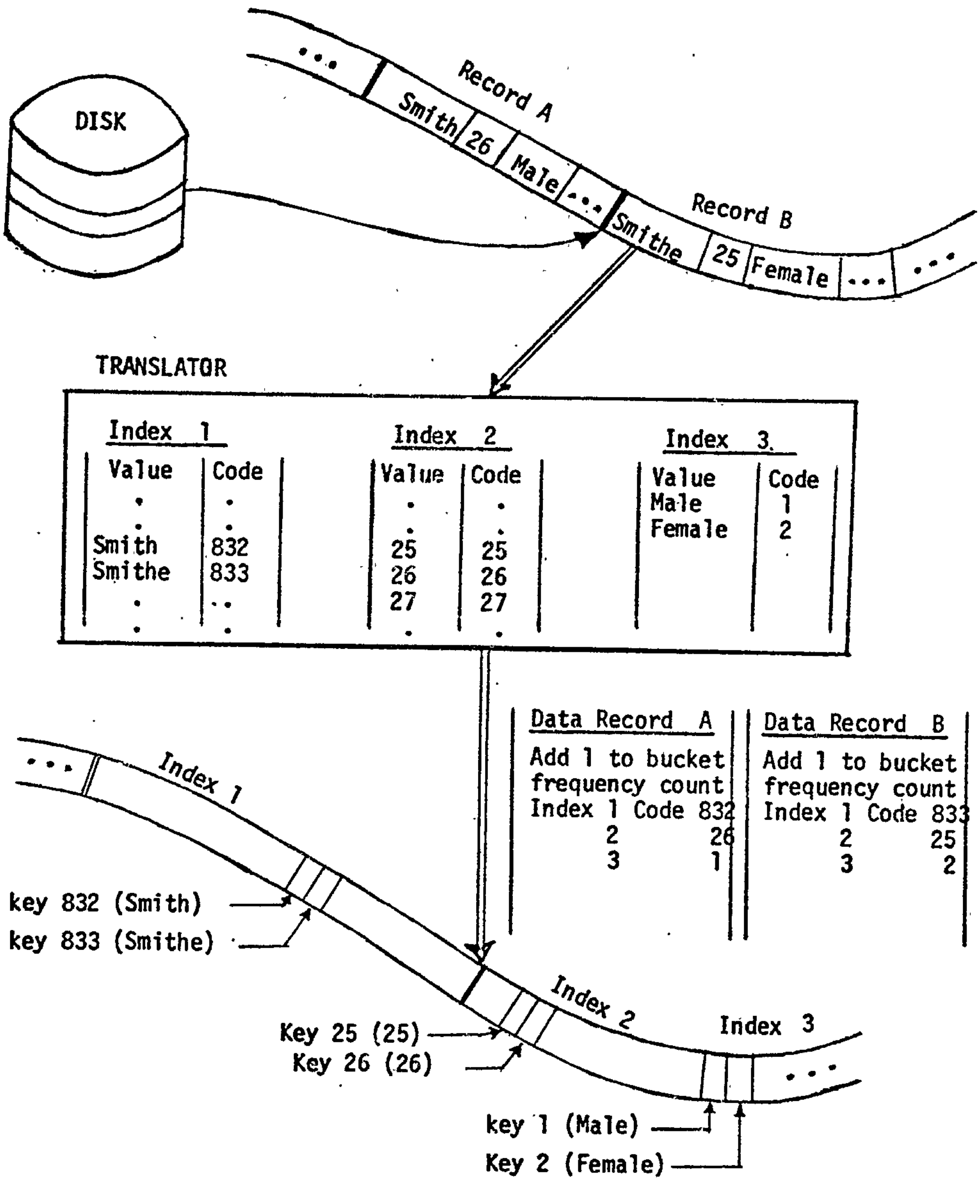
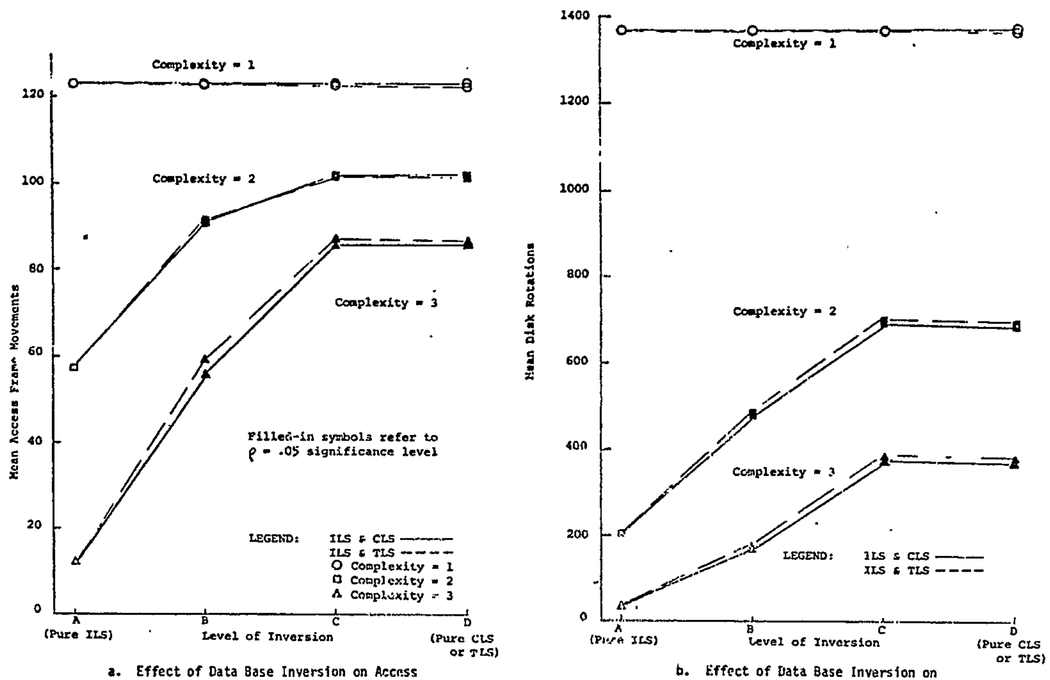


Figure 8. Translation of Real Records into Stochastic Bucket Records



a. Effect of Data Base Inversion on Access Frame Movements.

b. Effect of Data Base Inversion on Disk Rotations

NOTES

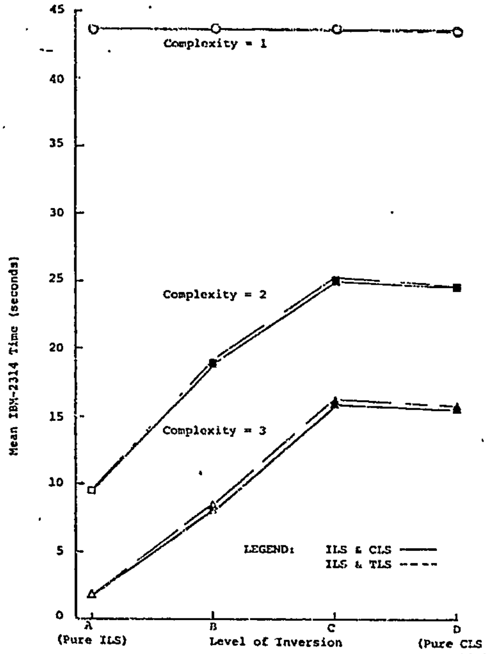
Inversion level A has indexes 1, 2, 3, 4, 5 under ILS

Inversion level B has indexes 3, 4, 5 under ILS and 1, 2 under TLS (----) or CLS (——)

Inversion level C has indexes 5 under ILS and 1, 2, 3, 4 under TLS (----) or CLS (——)

Inversion level D has indexes 1, 2, 3, 4, 5 under TLS (----) or CLS (——)

All measures reflect performance statistics after keys are located in their ISAM files.



c. Effect of Data Base Inversion on IBM-2314 Time

Figure 9. Effect of Data Base Inversion with Zipf Queries and Zipf Data Base.

REFERENCES

(1) Bleier, R.E. and A.H. Vorhaus, "File Organization in the SDC Time-Shared Data Management System (TDMS)," TDMS Project, Systems Development Corporation, SP-2750, August 1967.

(2) International Business Machines Corporation, IBM System/360 Component Descriptions - 2314 Direct Access Storage Facility and 2844 Auxiliary Storage Control, IBM Corp., San Jose, A26-3599-2, 1967.

(3) International Business Machines Corporation, Information Management System/360, Version 2, IBM Corp., White Plains, GH20-0765-1, 1970.

(4) International Business Machines Corporation, Introduction to IBM System/360 Direct Access Storage Devices and Organization Methods, IBM Corporation White Plains, C20-1649-2, October, 1967.

(5) Lefkovitz, David, File Structures for On-Line Systems, Spartan Books, New York, 1969.

(6) Mendelson, Eliott, Introduction to Mathematical Logic, D. Van Nostrand Co., New York, 1965.

(7) Prywes, N.S., and H.J. Gray, "The Multi-List for Real-Time Storage and Retrieval," Proceedings of IFIP Congress 1962, Munich, September, 1962, pp. 273-278.

(8) Prywes, N.S., and H.J. Gray, et al., "The Multi-List Type Associative Memory," Proceedings of Symposium on Gigacycle Computing Systems, AIEE Publications, No. S-136, January 1962, pp. 87-107.

(9) Raven, N., Proceedings of the Third Annual National Colloquium on Information Retrieval, Philadelphia, Pennsylvania, May 1966.

(10) Systems Development Corporation, DS-2 User Manual, Systems Development Corp., Santa Monica, 1970.

(11) Weizenbaum, J. "Knotted List Structures," Communications of the ACM, Vol. V, No. 3, March 1962, pp. 161-165.

(12) Zipf, George Kingsley, Human Behavior and the Principle of Least Effort, Addison Wesley Press, Cambridge, Mass., 1949.