IS-3389
Distribution Category
UC-32

DATA BASE DESIGN CONSIDERATIONS FOR A KEY WORD BASED RETROSPECTIVE INFORMATION RETRIEVAL SYSTEM

J.R. Jordan and C.G. Maple

AMES LABORATORY, USAEC
IOWA STATE UNIVERSITY
AMES, IOWA

Date Transmitted: June 1974

PREPARED FOR THE U. S. ATOMIC ENERGY COMMISSION DIVISION OF RESEARCH
UNDER CONTRACT W-7405-eng-82

IS-3389

Distribution
Category
UC-32

DATA BASE DESIGN CONSIDERATIONS FOR A KEY WORD BASED RETROSPECTIVE INFORMATION RETRIEVAL SYSTEM

J. R. Jordan and C. G. Maple

Ames Laboratory, USAEC
Iowa State University
Ames, Iowa 50010

Date Transmitted: June 1974

PREPARED FOR THE U. S. ATOMIC ENERGY COMMISSION
DIVISION OF RESEARCH UNDER CONTRACT NO. W-7405-eng-82

Available from: National Technical Information Service
Department A
Springfield, VA 22151

Price: Microfiche $1.45
Paper Copy $4.00

TABLE OF CONTENTS

		PAGE
	ABSTRACT	iv
I.	INTRODUCTION	1
II.	THE PROBLEM	1
	A. THE RETRIEVAL PROCESS	1
	B. UPDATING THE DATA BASE	3
III.	CONCLUSIONS	7

ABSTRACT

This document examines the nature of a key-word based information retrieval system and discusses the various problems involved. It presents the reason for choosing the Indexed Sequential Access Method (ISAM) for file structure.

Data Base Design Considerations for a Key Word Based Retrospective Information Retrieval System

I. Introduction

The purpose of this paper is to examine the nature of a key-word based information retrieval system to decide what sort of underlying file structures are required to make the project feasible. If possible we would like to be able to use to good advantage one of the existing file structures as supplied from a vendor, specifically, IBM. It is a foregone conclusion that the language to be employed is PL/1 because of its character manipulative features and the rich set of file handling capabilities that it has.

II. The Problem

A. The Retrieval Process

The object is: given a file which contains information describing what terms are in what documents and a request for documents, find the set of documents which satisfy the request. The document may be described by an array, D.

$$D = \begin{bmatrix} d_1 \\ d_2 \\ \cdot \\ \cdot \\ \cdot \\ d_n \end{bmatrix} = \begin{bmatrix} t_{11}, t_{12}, \ldots, t_{1m_1} \\ t_{21}, t_{22}, \ldots, t_{2m_2} \\ \cdot \\ \cdot \\ \cdot \\ t_{n1}, d_{n2}, \ldots, t_{nm_n} \end{bmatrix}$$

Each row describes a document. d_i is the document number and t_{ij} is the j^{th} term in document i. A request is a boolean expression involving a set of terms ($\{t_1, t_2, \ldots, t_\ell\}$) and the boolean operators (* = AND, + = OR and $\neg$ = NOT). For each document d_i, i = 1, 2, ..., n, construct a boolean representation of it as follows:

$$\text{Let } \tau_{ij} = f(d_i, t_j) = \begin{cases} \underline{\text{true}} \text{ if } t_j \in d_i \\ \underline{\text{false}} \text{ if } t_j \notin d_i \end{cases}$$

$$\text{Then let } \Delta = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_n \end{bmatrix} = \begin{bmatrix} \tau_{11} & \tau_{12} & \cdots & \tau_{1m_1} \\ \tau_{21} & \tau_{22} & \cdots & \tau_{2m_2} \\ & & \vdots & \\ \tau_{n1} & \tau_{n2} & \cdots & \tau_{nm_n} \end{bmatrix}$$

A document d_i is said to satisfy the request R if, after substituting τ_{ij} for t_j in R, R is true. A straight-forward way of determining the set of documents which satisfies the request is to match the request against every row of D. If D contains few rows, or if D has many rows but it is expected that R will be true in a great many cases, this is an acceptable procedure. But if D is large and R will be true in only a few cases, this approach is very expensive.

We may construct another matrix from D, call it T by inverting the roles of d and t. That is, let the t's be the row label and the d's be the data entries in the rows. This is then called an inverted file(1) and we have:

$$T = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_m \end{bmatrix} = \begin{bmatrix} d_{11}, d_{12}, \ldots, d_{1n_1} \\ d_{21}, d_{22}, \ldots, d_{2n_2} \\ \vdots \\ d_{m1}, d_{m2}, \ldots, d_{mn_m} \end{bmatrix}$$

Each row of T describes a term (t_j) in that it has the document numbers of all the documents in which t_j is found.

We are now able to enter T and extract the document numbers associated with each term.

We may then re-invert this subset and construct a new but much smaller D matrix which may be economically scanned in the straight-forward manner described above.

When we had only the D matrix, we were obligated to search it serially, matching the request against each row. The T matrix on the other hand, may be entered randomly. If the number of terms in the request is very large, a serial search may still be in order.

The random access to T must be made on the basis of a character string key, the term. Additionally it will be necessary to access all of the terms which begin with a given substring of characters -- a so-called "root" or "prefix" matching capability.

B. Updating of the Data Base

It is anticipated that this general system will be used in basically two forms. The primary intention is to have it support an on-going retrospective retrieval system with the source input coming from SDI (2).

In this case the data volumes are tremendous both from an initial standpoint and a growth rate. The other use will be for smaller systems and smaller growth rates. It would be very advantageous if the basic ideas could be adaptable to both kinds of systems. The small case is of relatively little concern since it will not be very expensive to update regardless of the choice of file structures.

The large system which is fed by SDI is another matter. With 5000^{+} new documents each week, we can expect a higher percentage of the terms to have added document numbers each week. Also, initially, the number of terms can be expected to grow quite rapidly. After the system has been in operation for a while the growth rate of new terms will slow (see Figure 1). So we will have both new records and expanded records to deal with. Because of the high incidence of updating, sequential access is almost mandatory.

Another problem comes from the fact that the length of the rows in T have very large variance. (See Figure 2.) This almost immediately rules out a file structure which supports only fixed length records.

In summary, the constraints are:

1. Random access in the retireval mode (low reference rate).
2. Sequential access in the update or amendment mode (high reference and volatility).
3. Great variance in record lengths.
4. Character string and substring keys for random entry to the file.

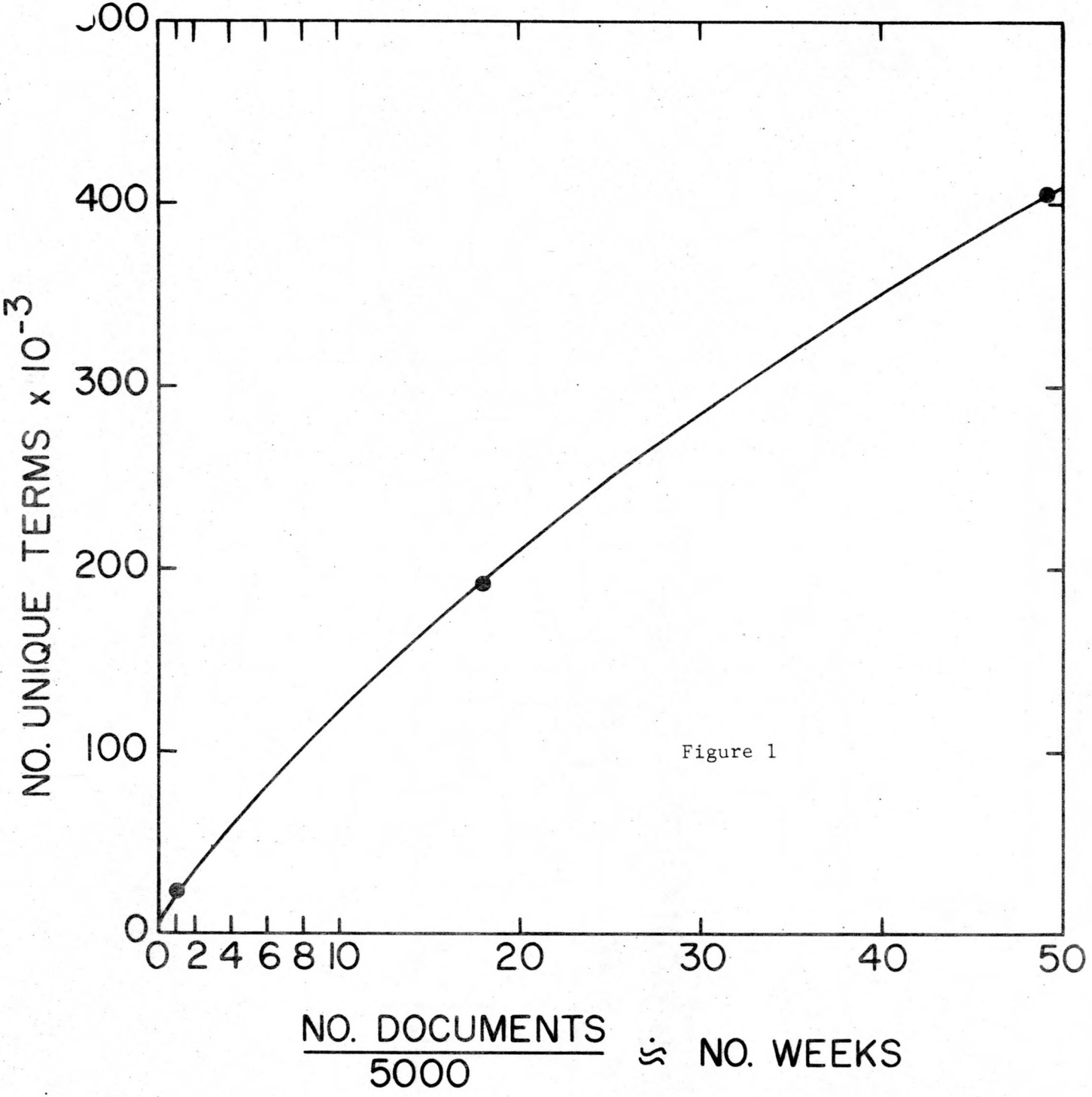
500
400
300
200
100
0
NO. UNIQUE TERMS x 10^{-3}
0 2 4 6 8 10 20 30 40 50
NO. DOCUMENTS / 5000 ≈ NO. WEEKS

Figure 1

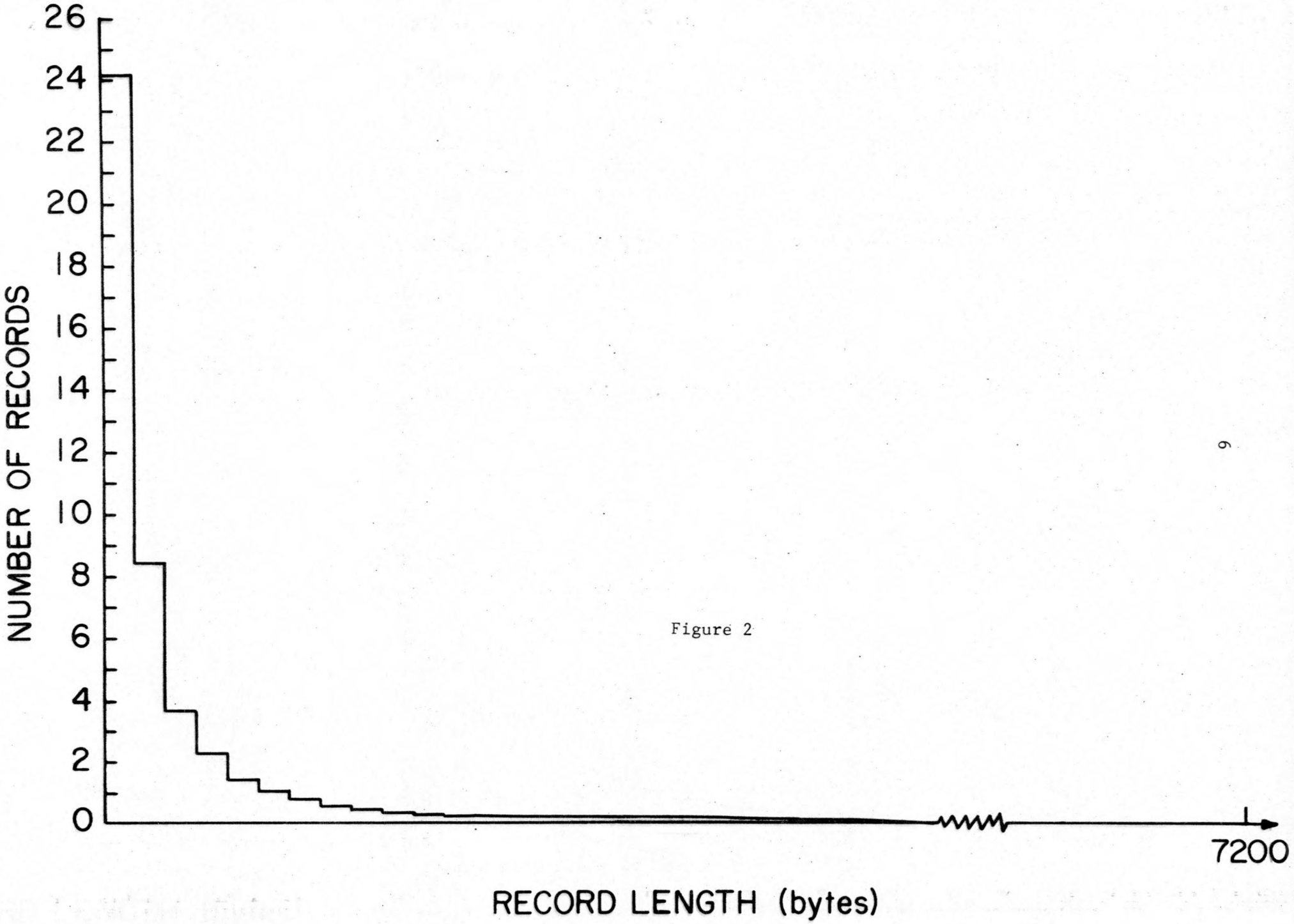
26
24
22
20
18
16
14
12
10
8
6
4
2
0
NUMBER OF RECORDS
7200
RECORD LENGTH (bytes)

Figure 2

III. Conclusions

Without enumerating the problems associated with all the available file structures, we will simply state that only one which is available is able to satisfy all of the above constraints. That is the so-called Indexed Sequential Access Method (ISAM).

Although it actually must, in some circumstances, make several probes to a disk in order to actually retrieve a given record, it appears random to a programmer and is quite fast. It allows sequential access for updating.

It allows arbitrary character strings for keys. Also, it has a feature called GENKEY, which stands for "generic key". It allows specification of a leading substring of a key. The file is positioned at the first record which has that substring as the left part of its key. Then one may access the file sequentially from that point and be able to find all the records which have the substring as the left-most part of the key.

All-in-all ISAM is ready-made for this application.

References

1. Salton, G., Automatic Information Organization and Retrieval, McGraw-Hill, New York (1968).

2. Jordan, J. R., "Let the Computer Select Your Reading List" Datamation (1970).