Computer Science and Systems Analysis

Computer Science and Systems Analysis

Technical Reports

Miami University *Year* 1997

Effect of Tunable Indexing on Term Distribution and Cluster-based Information Retrieval Performance

Timothy Schorr
Miami University, commons-admin@lib.muohio.edu

MIAMI UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE & SYSTEMS ANALYSIS

TECHNICAL REPORT: MU-SEAS-CSA-1997-000

Effect of Tunable Indexing on Term Distribution and Cluster-based Information Retrieval Performance

Timothy Lee Schorr

School of Engineering & Applied Science | Oxford, Ohio 45056 | 513-529-5928

EFFECT OF TUNABLE INDEXING ON TERM DISTRIBUTION AND CLUSTER-BASED INFORMATION RETRIEVAL PERFORMANCE

A Thesis

Submitted to the Faculty of Miami University

in partial fulfillment of

the requirements for the degree of

Master of Systems Analysis

by

Timothy Lee Schorr

Miami University

Oxford, Ohio

1997

Advisor Faith Cox

Reader Alton F. Sanders

Reader Valerie V. Cross

Abstract

EFFECT OF TUNABLE INDEXING ON TERM DISTRIBUTION AND CLUSTER-BASED INFORMATION RETRIEVAL PERFORMANCE

by Timothy Lee Schorr

The purpose of this study is to investigate the effect of tunable indexing on the structure and information retrieval performance of a clustered document database. The generation of all cluster structures and calculation of term discrimination values is based upon the Cover Coefficient-Based Clustering Methodology. Information retrieval performance is measured in terms of precision, recall, and e-measure. The relationship between term generality and term discrimination value is quantified using the Pearson Rank Correlation Coefficient Test. The effect of tunable indexing on index term distribution and on the number of target clusters is examined.

Table of Contents

List of Figures

List of Tables

1 Introduction

Academic, corporate, and government organizations are increasingly dependent upon very large databases for accessing vital information. Frequently, these databases contain full-text documents in either formatted or unformatted form. A user typically retrieves information from a document database by providing the system with a number of key words (query) which indicate the content of documents which the user wishes to view. The database's associated search engine accepts the user query and performs some type of database search in an effort to find documents which are relevant to the query. Some, or all, of the candidate documents are then presented to the user for further perusal. The system's determination of a document's relevance to a query is usually based on the similarity of the query to the document, or some portion of the document (e.g. the abstract). In general, the similarity value reflects the number of terms common to the query and the document, although some term normalization considerations are made. Documents and queries are represented by their constituent terms, referred to as *indexing terms*, and the entire collection of indexing terms for the database is called the *indexing vocabulary*.

A number of database search methods exist, and the efficiency of the search method often determines the overall efficiency of the system. The document database and its associated search engine are referred to collectively as an information retrieval system (IRS). A schematic IRS is presented in Figure 1. An excellent overview of modern information retrieval concepts and systems is provided in [10] and [12].

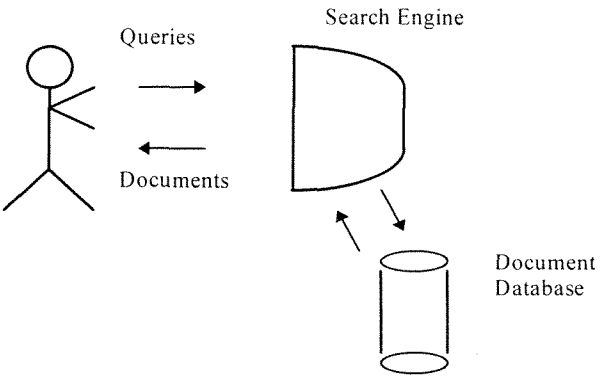


Figure 1. Schematic Information Retrieval System (IRS)

The performance of information retrieval systems is generally measured in terms of *precision, recall,* and another criterion known as *e-measure*[15]. Precision is defined as the ratio of retrieved relevant documents to the total number of documents retrieved. Recall is defined as the ratio of retrieved relevant documents to the total number of relevant documents in the database. The e-measure is defined in terms of precision and recall, as follows:

$$e = 1 - 1/\ \alpha(1/\text{precision}) + (1 - \alpha)1/\text{recall}\ ,$$

where $0 \leq e \leq 1$, α is the importance of precision with respect to recall, and is defined as:

$$\alpha = 1/(\beta^2 + 1),$$

where β is the importance of recall with respect to precision. For example, if equal emphasis is placed upon precision and recall, then $\beta = 1$ and $\alpha = 1/2$. If no importance is placed on recall, then $\beta = 0$, $\alpha = 1$, and e-measure simplifies to e = 1 - precision.

Similarly, if no importance is placed on precision, then $\beta = \infty$, $\alpha = 0$, and e = 1 - recall. In the case of precision and recall, an increasing value indicates improved IRS performance, while the e-measure decreases in value with improved IRS performance.

There are many ways of physically and logically constructing a document database. A simple method is to store a term description of each document and conduct a full-search (FS) of each document in response to a query. This is often called a brute force search method. Since document databases are generally quite large, FS methods are quite inefficient with regard to processing time.

Another approach to database structure is to form clusters (sometimes hierarchical) of similar documents. With this approach, an inter-document similarity value is calculated, and documents with relatively high mutual similarities are grouped together into clusters. Again, the similarity value generally reflects the number of terms common to a pair of documents. The resulting database consists of a number of document clusters, where each cluster represents a collection of documents having a strong mutual association. Each cluster also has a representative document called a *centroid.* The centroid is not necessarily an actual document, rather it is a system generated document, contrived in such a way as to represent an average document within the cluster. In response to a user query, a similarity measure between the query and each cluster centroid is determined. A full search is then performed only on documents within the cluster(s) having the most similar centroid(s). A clustered document collection is often referred to as a partitioned document space. Clustered document databases may offer improved performance

because the number of documents which are subjected to FS is greatly reduced. The efficiencies of several information retrieval methods are presented in [1] and [9].

This study deals exclusively with clustered document databases. Its purpose is to demonstrate how an indexing vocabulary can be tailored to achieve better IR performance. More specifically, it will attempt to show that an indexing vocabulary consisting of terms having the highest term discrimination values yields a cluster structure which delivers superior IR performance. The paper is organized as follows: section 2 provides background information and clustering concepts, section 3 deals with the concept of tunable indexing and term discrimination value, section 4 states the experimental hypotheses, section 5 details the experimental procedure, section 6 provides the study's results, and section 7 summarizes the study's conclusions and provides suggestions for future research.

2 Indexing and Clustering in Information Retrieval Systems

2.1 Indexing

In an information retrieval system, a document is identified by its constituent terms. The process of identifying a document by its individual terms is called indexing, and the terms used by an IRS to identify member documents are collectively referred to as the *indexing vocabulary*. For a document database containing *m* documents (D_1, D_2, D_3, ...D_m) the indexing vocabulary will consist of *n* terms, and can be described by the vector $T = (t_1, t_2, t_3, ...t_n)$. It is then possible to describe a document, D_i, by an *n* dimensional *document vector*: $D_i = (d_{i1}, d_{i2}, d_{i3}, ... d_{in})$, where d_{ij} indicates the weight of the j^{th} indexing term in the i^{th} document. The entire document database can then be identified by an *m* x *n* matrix, referred to as the *D matrix*. This approach is referred to as the vector space model [13]. The terms may be weighted according to their importance, or frequency of occurrence, or they may be unweighted (binary), thus restricting d_{ij} to 1 or 0. Typically, any document D_i will be defined by several indexing terms, and similarly any given indexing term t_j will be present in several different documents. The average number of unique terms per document is the *depth of indexing* x_d, while the average number of unique documents per indexing term is *term generality* t_g. Appendix 1 provides a sample D matrix containing 5 documents defined by 6 unique index terms, and illustrates how x_d and t_g are determined.

Intuitively, some of a document's terms are more descriptive of the document's content than others. For example, a document pertaining to information retrieval systems would

contain, but not be well described by, the following words: and, to, the, this, etc. Conversely, the same document would contain, and be much better described by, the following words: information, retrieval, system, document, etc. Therefore, indexing vocabularies (and document vectors) usually exclude commonly used, non-descriptive terms. The same logic applies to queries and query vectors.

Given two document vectors, it is possible to determine a similarity measure between them, $s(D_i, D_j)$. One method of determining $s(D_i, D_j)$ relies upon the cosine function and defines the similarity measure as:

$$s(D_i, D_j) = \cos(D_i, D_j) = \frac{D_i \bullet D_j}{\| D_i \| \| D_j \|,} = \frac{\sum_{k=1}^{n} d_{ij} \bullet d_{jk}}{\sqrt{\sum_{k=1}^{n} d_{ik}^2} \sqrt{\sum_{k=1}^{n} d_{ik}^2}}$$

A user query may also be represented by a vector, $Q_i = (q_{i1}, q_{i2}, q_{i3}, \ldots q_{in})$. Where q_{ij} indicates the weight of the j^{th} indexing term in the i^{th} query. Again, the terms may be weighted or binary, and a similarity value, $s(Q_i, D_j)$, may be calculated between query vector Q_i and document vector D_j. An example similarity calculation is provided in Appendix 1.

2.2 Clustering and the Cover Coefficient-Based Clustering Methodology

The essential idea in clustering is that similar documents are grouped together to form clusters. The underlying reason is known as the "clustering hypothesis", which states that

"closely associated documents tend to be relevant to the same requests"[3]. Therefore, grouping similar documents provides a browsing tool and accelerates the user's search process. This hypothesis validates the clustering of documents in a database. The search strategy in a cluster-based document database, known as *cluster-based retrieval (CBR),* is to first compare a query vector with each cluster centroid. Detailed query to document comparison is then performed only in selected clusters; generally, the most similar x clusters.

The clustering algorithm used in this study is known as the *Cover Coefficient-based Clustering Methodology (C^3M)*[2,3,5]. The C^3M algorithm is of the *partitioning type*, meaning that a document appears in only 1 cluster. Also, the resulting cluster structure is non-hierarchical, and is *seed based.* That is, each cluster contains a seed document, or simply a *seed,* which attracts other relevant documents to itself. The seed acts like a nucleation site for the cluster. The C^3M algorithm takes a probabilistic approach to defining the inter-document relationships. These relationships are described by an m x m C matrix, whose elements convey document/term couplings. More formally:

A D matrix that represents the document database $\{D_1, D_2, ... D_m\}$ described by the index terms $T = \{t_1, t_2, ... t_n\}$ is given. The Cover-Coefficient matrix, C, is a document-by-document matrix whose entries c_{ij} $(1 \leq i, j \leq m)$ indicate the probability of selecting any term of D_j from D_i[3].

In other words, the C matrix indicates the relationship between documents based on a 2-stage probability experiment. The experiment randomly selects terms from documents

in 2 stages. First, one arbitrarily chooses a term t_k from document D_i, then tries to select document D_j from this term. That is, check if document D_j contains t_k. Each row of the C matrix summarizes the results of this 2-stage experiment. This can be better understood by analogy. Suppose we have many urns, and each urn contains different numbers of balls of different colors. Then what is the probability of selecting a ball of a particular color? To find this probability, we first must randomly select an urn, then randomly select a ball from this urn. In terms of the D matrix, we have the following: From the terms (urns) of D_i, choose one at random. Each term appears in many documents, or each urn contains many balls. From the selected term, try to draw D_j, or from the selected urn try to draw a ball of a particular color. What is the probability of getting D_j, or what is the probability of selecting a ball of a particular color? This is the probability of selecting any term of D_j from D_i. An example of deriving the C matrix from a given D matrix is illustrated in Appendix 2. It is worth noting that the diagonal entries of the C matrix, c_{ii}, represent the probability of selecting document D_i from any term in document D_i. Therefore, c_{ii} $(1 \le i \le m)$ is a measure of the uniqueness of document D_i, and is referred to as the *decoupling coefficient*, δ_i. On the other hand, the sum of the off-diagonal entries for document D_i represents the coupling of D_i with the other documents in the collection. This sum is referred to as the *coupling coefficient*, ψ_i, where $\psi_i = 1 - \delta_i$ $(1 \le i \le m)$.

Cluster seed documents must have proper degrees of uniqueness and inter-document coupling (i.e. proper values of δ_i and ψ_i). A good cluster seed document strikes a

balance between being relatively unique within the collection, yet not being entirely composed of highly unique index terms. Cluster seed documents are selected based upon *cluster seed power,* P_i, where

$$P_i = \psi_i \; x \; \delta_i \; x \; \sum_{j=1}^{n} d_{ij}.$$

The summation provides normalization to P_i, and for a binary matrix will simply be the number of terms in d_i. Documents having the highest cluster seed power are selected as cluster seeds, and any remaining document D_i is assigned to the cluster containing the seed document D_j for which c_{ij} is greatest.

Finally, it can be shown that the number of clusters, n_c, in a document collection equals the summation of all δ_i values. Intuitively, this is best understood by considering 2 separate document collections. The first is comprised of *m* unique documents (i.e. all c_{ii} values = 1 and all c_{ij} values are 0). In this case, each document represents an individual cluster so that $n_c = m$. Also, since each $c_{ii} = 1$,

$$\sum_{i=1}^{m} c_{ii} = \sum_{i=1}^{m} \delta_i = m = n_c.$$

The second collection consists of *m* identical documents (i.e. all c_{ii} values = $1/m$ and all c_{ij} values = $1/m$). In this case, all documents are clustered into one group so $n_c = 1$, and

$$\sum_{i=1}^{m} c_{ii} = \sum_{i=1}^{m} \delta_i = 1 = n_c.$$

In each of these cases we see that the summation of decoupling coefficients equal n_c. It was shown in [3] that for all cases,

$$n_c = \sum_{i=1}^{m} \delta_i .$$

2.3 Term Weighting and Matching Functions

Term weighting is a means of expressing the importance of the occurrence of a term in a document. Weighting schemes generally have three components: Term Frequency Component (TFC), Collection Frequency Component (CFC), and Normalization Component (NC). The weight of a term in a document or query (represented by ω_{dj} and ω_{qj}, $1 \leq j \leq n$, respectively) is then determined by the product (TFC x CFC x NC), and is expressed in the form *document weighting scheme · query weighting scheme*. The best weighting schemes for the test databases of this study, INSPEC and NPL, were established in [1] and [11], respectively. For INSPEC the scheme is *txc · txx*, while NPL employs *bxx · bpx*. An explanation of these letter designations is provided in Appendix 3. The matching function for a query-document pair is then given by:

$$\text{similarity}(Q, D) = \sum_{k=1}^{n} \omega_{dk} \text{ x } \omega_{qk}.$$

The matching function for a centroid-query pair is the same as that for a query-document pair. Also, the centroid weighting schemes are the same as those used for documents.

3 Concepts of Tunable Indexing

3.1 Concept of Term Discrimination Value

An ideal IRS would respond to any given query by retrieving *only* documents which are relevant to the query, and by retrieving *all* documents which are relevant to the query; thereby yielding precision and recall values of 1.0. The system would easily discriminate between relevant and non-relevant documents. In practice, however, the process of discriminating relevant from non-relevant documents is difficult and imperfect. Intuitively, the process of document discrimination becomes easier as the documents themselves become more unique. To demonstrate this, consider 2 document vectors D_i and D_j, and their similarity value $s(D_i, D_j)$: where $s(D_i, D_j)$ increases as the 2 documents contain more and more common terms. If $s(D_i, D_j)$ is relatively high, it may be quite difficult to discriminate between D_i and D_j. Correspondingly, it would be difficult to formulate a query which retrieves either D_i or D_j, but not the other. On the other hand, if $s(D_i, D_j)$ is relatively low, then it becomes comparatively simple to discriminate between D_i and D_j, and one can easily formulate a query which retrieves one but not the other. Extending this logic to an entire document collection, one can see that in order to improve precision and recall it is necessary to lower the average inter-document similarity for the entire collection. More formally, it is desirable to minimize the following:

$$F = \sum_{i=1}^{m} \sum_{j=1}^{m} s(D_i, D_j) \quad \text{where } i \neq j. \qquad (1)$$

When eq. (1) is minimized, the average similarity between document pairs is smallest and each document may be retrieved without also necessarily retrieving its neighbors. Also, in a collection where there are several relevant documents for a given query, it will be possible to retrieve all relevant documents, while rejecting the non-relevant documents. Thus, high precision and recall outputs are assured.

These concepts are easily and naturally applied to clustered document databases. A cluster structure having widely separated centroids and high intra-cluster similarity will optimize precision and recall outputs. When considered in aggregate, such a structure would be referred to as having highly decoupled, highly cohesive clusters.

The computational cost of eq. (1) can be lowered significantly by computing a centroid G for the entire document collection. Each centroid entry g_j ($1 \leq j \leq n$) of G is then defined as the average weight of t_j in all the *m* documents:

$$g_j = (1/m) \sum_{i=1}^{m} d_{ij}.$$

The approximate document *space density*, Q, can then be defined as follows:

$$Q = (1/m) \sum_{i=1}^{m} S(d_i, G). \qquad (2)$$

Accordingly, document collections with greater (lesser) separation of document description vectors will have lower (greater) Q value. It follows that the careful selection of indexing terms can impact the space density value for the entire document collection [13].

Term discrimination value is used to measure how an indexing term affects the overall separation of a document collection[4,6]. The deletion of any term t_j from T will change the indexing vocabulary and the description of documents. Since Q is a function of document descriptions, such a change will also change the document space density for the entire collection. The deletion of t_j ($1 \leq j \leq n$) will set d_{ij} ($1 \leq i \leq m$) and g_j to null. The new value of Q, Q_j, will be as follows:

$$Q_j = (1/m) \sum_{i=1}^{m} S(d_i^j, G_j),$$

where $d_i^j = (d_{i1}, d_{i2}, \ldots, d_{i,j-1}, d_{i,j+1}, \ldots, d_{in})$ and
$G_j = (g_1, g_2, \ldots, g_{j-1}, g_{j+1}, \ldots g_n)$.

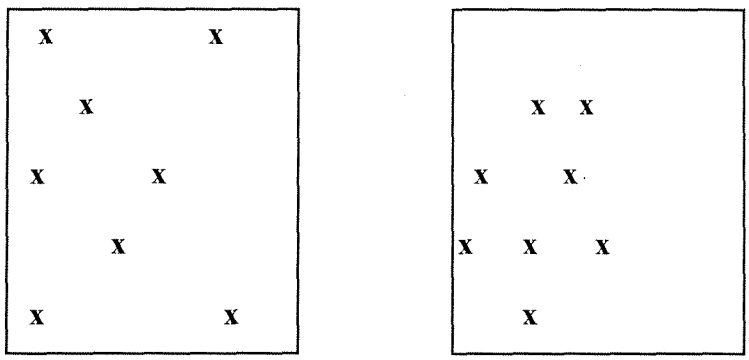
The difference Q_j - Q reflects the change due to the deletion of t_j . For example, if the use of t_j in the indexing vocabulary increases the separation of documents, then its effect will be to decrease the document space density. Consequently, the deletion of t_j will decrease the separation of documents, increasing the document space density. It follows that Q_j - Q will be greater than zero. Figure 2 demonstrates the effect of deleting such a term. The difference Q_j - Q is defined as the *term discrimination value of* t_j , TDV_j. TDV_j has the following properties:

(a) $TDV_j > 0$ for a good discriminator t_j,

(b) $TDV_j \approx 0$ for an indifferent term t_j,

(c) $TDV_j < 0$ for a poor discriminator t_j.

Selecting the terms of an indexing vocabulary based on their TDV_j has the potential to enhance the performance of a cluster based IR system in terms of precision, recall and e-measure. Using only the best discriminators, a structure of widely separated, highly cohesive clusters should be obtained.



(a) **(b)**

Figure 2. Shows separation of documents when a term t_j with $TDV_j > 0$ is (a) included in the indexing vocabulary and (b) deleted from the indexing vocabulary.

3.2 Cover Coefficient Determination of TDV

The concept of a decoupling coefficient, δ_i, was previously discussed. The average decoupling coefficient δ is defined as:

$$\delta = \sum_{i=1}^{m} \delta_i / m.$$

The document space density (Q) is similar to the overall decoupling coefficient δ. If document descriptions are more distinguishable (i.e. Q is low), then the documents are more decoupled from one another (i.e. δ is high). Conversely, when Q is high (i.e. the documents are less distinguishable) δ will be low. Using this concept, we may use the CC method to compute TDV's [4]. Assuming the deletion of term t_j does not alter the number of documents (i.e. each document is defined by at least 2 terms), then we may use $\delta = \sum \delta_i$ by ignoring the divisor *m*. We know, however, that $\sum_{i=1}^{m} \delta_i$ is simply the number of clusters which are expected to exist within the collection. In a manner analogous to computing TDV_j as $Q_j - Q$, we may compute TDV_l as $n_c - n_{cl}$, where n_c and n_{cl} are, respectively, the number of clusters before and after deleting term t_l. In this context, TDV_l has the following properties:

(a) $n_c > n_{cl}$ for terms which are good discriminators,

(b) $n_c < n_{cl}$ for terms which are poor discriminators,

(c) $n_c \approx n_{cl}$ for terms which have no description significance.

This shows that the concepts of space density (Q) and average decoupling of documents (δ) are inversely related. It is worth mentioning that the diagonal entries of the C matrix, c_{ii} ($1 \leq i \leq m$), are not related to $s(D_i, D_i)$, since $s(D_i, D_i) = 1$, while $c_{ii} = 1$ only if D_i is entirely unique. Also, if all documents in a collection are entirely unique, then all $\delta_i = 1$, $n_c = m$, and all TDV's will be equal to zero. To repeat, TDV_l will be determined by

$n_c - n_{cl}$, where good, poor, and indifferent discriminators will have positive, negative, and near zero values, respectively.

It is important to note that TDV's are relative for a given document collection. In other words, valid comparisons of TDV's cannot be made between different document collections even if the same term is considered in each collection. Furthermore, while different means of calculating TDV's will not assign identical values to a given term, the different methods should demonstrate consistency in the **relative values** assigned. In fact, the degree of consistency of the CC method with accepted similarity based calculation methods was documented for a small database in [4]. That study showed that the consistency of the CC based TDV's with these other methods is excellent for determining poor discriminators (negative TDV), while there is some divergence between all methods in determining good discriminators. The study also showed the CC method to be more efficient in terms of computational complexity than other accepted methods. For other methods of calculating TDV see [6,7,8,14].

3.3 Tunable Indexing

We may apply the concepts of TDV and CC to control the number of clusters, N_c, within a document collection. *Tunable indexing* is the process of selecting appropriate indexing terms to control N_c, and the selection of index terms is based upon individual TDV. Since terms having relatively high TDV make documents more unique, their addition to the indexing vocabulary will increase N_c, while terms with relatively low TDV decrease N_c. When all terms are included in the indexing vocabulary, the original D matrix is used

and the natural number of clusters n_c exists (i.e. $N_c = n_c$). The tunable indexing procedure is outlined as follows[4]:

(1) Determine TDV_l for each index term t_l, using the CC methodology as previously described,

(2) Sort terms according to their TDV's ,

(3) Beginning with the term having the highest (lowest) TDV_l , select index terms until each document is defined by at least 1 term. This will yield the maximum (minimum) value for N_c , and n_{mod} terms are used for indexing,

(4) Continue adding index terms until desired N_c is reached, or until all terms have been used. When all terms are used, $N_c = n_c$.

This algorithm produces a general graph of N_c versus number of indexing terms used l as given in Figure 3. Point A of Figure 3 indicates the maximum possible number of clusters, $N_{c\,max}$. Point A is reached when, beginning with the term having the highest TDV_l and working toward terms having lower TDV_l, terms are successively added to the D matrix until all documents are defined by at least 1 term. At this point, the D matrix (referred to as the *maximal D matrix*) contains n_{max} terms and is of dimension m x n_{max}. The portion of the curve between points A and B depicts the effect of adding to the maximal D matrix, terms with successively lower TDV_l. Point C of Figure 3 indicates the minimum possible number of clusters, $N_{c\,min}$. Point C is reached when, beginning with the term having lowest TDV_l and working toward terms having higher TDV_l, terms are successively added to the D matrix until all documents are defined by at least 1 term. This time the D matrix (referred to as the *minimal D matrix*) contains n_{min} terms and is of

dimension m x n_{min}. The portion of the curve between points C and B illustrates the effect of adding to the minimal D matrix, terms with successively higher TDV_l. Point B represents the natural number of clusters n_c which exists when all indexing terms are used for cluster generation.

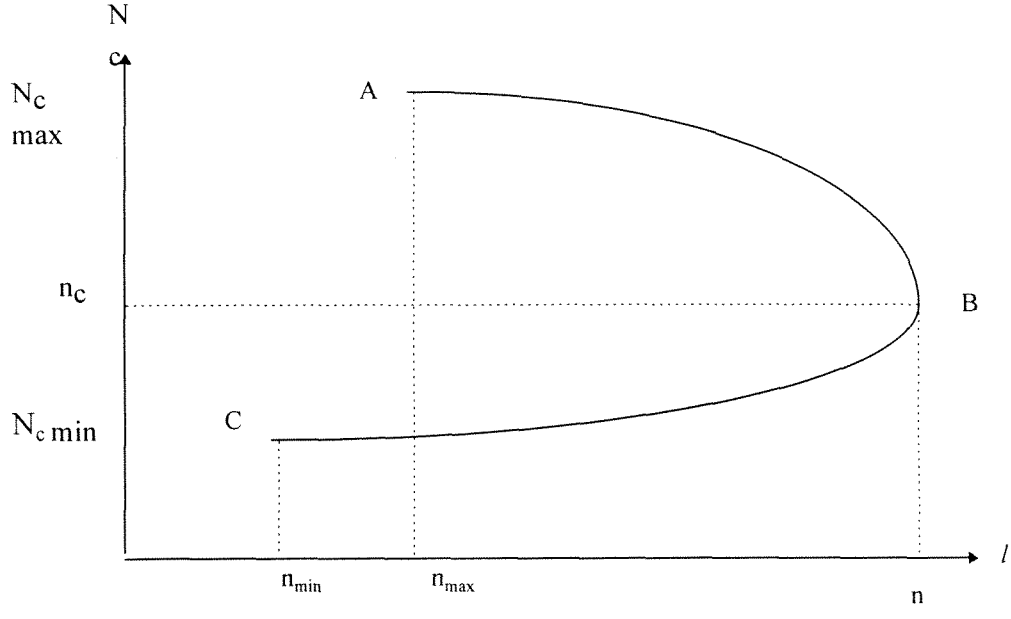


Figure 3. General Graph of Number of Clusters vs. Number of Terms

A brief explanation of the relationship between n_{min} and n_{max} is appropriate. As shown in Figure 3, $n_{min} < n_{max}$. Intuitively, this makes sense since n_{min} is arrived at by using terms of low TDV_l to describe the document collection. Such terms usually have relatively high t_g, so that fewer terms are necessary to achieve the minimal D matrix.

Terms having high TDV_l generally have lower t_g, therefore, more such terms are required to reach the maximal D matrix.

4 Hypotheses

4.1 Hypothesis of the Relationship of TDV to Term Generality

Terms with high term generality are, by definition, found in a large number of different documents. Such terms do little to distinguish any of these documents from any of the others. One would expect then, that terms with high t_g would have low TDV. On the other hand, terms with low t_g are found in few documents, and are important in distinguishing these documents from all other documents. It follows that terms having low t_g should have high TDV. This is the first hypothesis to be tested.

This hypothesis is somewhat at odds with what has been observed with similarity based measures of TDV[14]. In that case, terms with exceptionally high t_g were found to have very low TDV, terms with moderate t_g had relatively high TDV, and terms with very low t_g had TDV's of near zero.

4.2 Hypothesis of the Effect of Tunable Indexing on Information Retrieval Performance

Term discrimination value can be determined for each of the *n* indexing terms. Since terms with relatively high TDV make individual documents more distinguishable, they tend to increase δ, as well as the number of clusters N_c in the collection. Conversely, terms with relatively low TDV make individual documents less distinguishable, tend to

decrease δ, and decrease the number of clusters N_c in the collection. The maximum (minimum) value for N_c will be realized when the indexing vocabulary consists of the n_{max} (n_{min}) terms having the highest (lowest) TDV. Also, N_c may be varied on the range from $N_{c\,min}$ to $N_{c\,max}$.

Intuitively, some cluster structures should be superior to others and deliver correspondingly superior information retrieval performance. Since improved performance depends upon the system's ability to distinguish between relevant and nonrelevant documents, it is reasonable to hypothesize that the cluster structure in which documents are most unique should give the best information retrieval performance. Therefore, the IR performance for the case in which $N_c = N_{c\,max}$, and cluster generation is based on the n_{max} index terms having the highest TDV's, should be superior to the other cluster structures. Contrarily, the cluster structure in which documents are least unique should yield the worst information retrieval performance. This structure will be realized when $N_c = N_{c\,min}$ and the n_{min} index terms with lowest TDV's are used for cluster generation. This is the second hypothesis to be tested.

4.3 Hypothesis of the Effect of Tunable Indexing on the Number of Target Clusters

A cluster which contains at least 1 relevant document for a query is called a *target cluster*. The number of target clusters accessed in response to query *j* is denoted by $n_{tc\text{-}j}$, while the average number of target clusters accessed in response to a query set is represented by n_{tc} (i.e. $n_{tc} = \sum_{j=1}^{n} n_{tc\text{-}j}/n$). The range of $n_{tc\text{-}j}$ is determined by the lesser of 2 values: the number of relevant documents to query *j* (denoted rd_j), or the number of clusters N_c. For example,

if query *j* has 15 relevant documents and $N_c = 30$, then $n_{tc\text{-}j}$ may be at most 15. Alternatively, if query *j* has 15 relevant documents and $N_c = 10$, then $n_{tc\text{-}j}$ may be at most 10. It follows that n_{tc} is limited by the lesser of the average number of relevant documents for a query set (rd) or N_c for the cluster structure.

Intuitively, one expects that as a clustering structure improves, the ratio of average target clusters to the limiting value of average target clusters (i.e. n_{tc}/rd or n_{tc}/N_c) should decrease. It follows that as the cluster structure is modified by tunable indexing, the structures with lower values of n_{tc}/rd or n_{tc}/N_c should deliver correspondingly higher precision and recall and lower e-measure values. Since it is anticipated that as N_c increases IR performance will improve, it is expected that as N_c increases the value of n_{tc}/rd (or n_{tc}/N_c) will decrease, reaching a minimum when $N_c = N_{c\,max}$. This is the third hypothesis tested.

4.4 Hypothesis of the Effect of Tunable Indexing on Indexing Term Distribution

The number of unique clusters in which an indexing term exists is called the *cluster generality*. The cluster generality for index term *j* is denoted by c_{gj}, and the average cluster generality for all index terms is denoted by c_g (i.e. $c_g = \sum_{j=1}^{n} c_{gj}/n$). A discussion of c_{gj} and c_g is quite similar to that of $n_{tc\text{-}j}$ and n_{tc}, respectively. A significant difference is that c_{gj} is limited by the lesser of the term generality for term *j* (t_{gj}) and N_c. To illustrate this point, consider a case where t_j appears in 15 unique documents (i.e. $t_{gj} = 15$) and $N_c = 30$. Clearly, c_{gj} would be limited to a value of 15. On the other hand, if $t_{gj} = 30$ and $N_c = 15$, then c_{gj} is at most 15.

Again, it is expected that as cluster structures improve, this improvement should be manifest in a lower ratio of c_g/t_g or c_g/N_c. Furthermore, improved values of precision, recall, and e-measure should be associated with cluster structures having low values of c_g/t_g (or c_g/N_c). It is therefore expected that as N_c increases c_g/t_g (or c_g/N_c) should decrease, reaching a minimum when $N_c = N_{c\,max}$. This is the fourth hypothesis to be tested.

5 Experimental Procedure

5.1 Databases and Computing Environment

This study uses 2 databases, INSPEC and NPL, which have been used in numerous other research efforts. The INSPEC database contains documents pertaining to computer and electrical engineering topics, while NPL documents deal with topics in physics. Database characteristics are shown in Table 1.

Table 1. Summary of Databases

Symbol	Meaning	INSPEC	NPL
m	number of documents in database	12,684	11,429
n	number of distinct terms in database	14,573	7,491
n_c	number of clusters using CC	477	360
x_d	average depth of indexing	32.5	20.0
t_g	average term generality	28.3	30.5
--	number of queries provided with database	77	100
--	average number of terms/query	15.82	7.16
r_d	average number of relevant documents/query	33.03	20.83

All programs developed to support this study are written in the Pascal/VS language version 2.0, and are compiled and executed on Miami University's IBM mainframe.

5.2 Experiments in Tunable Indexing

A program was written which calculates the term discrimination value for each indexing term of the document database. Additionally, this program determines the term generality for each indexing term. The output of this program consists of 2 text files. The first file contains term numbers and their associated term discrimination values, sorted in ascending order according to TDV. The second file contains term numbers and their associated term generalities, sorted in ascending order according to term generality.

The file of term numbers and their associated TDV's provided the input for a second program which, starting with the indexing term of highest or lowest TDV, would select the minimum number of terms (n_{min} or n_{max}) necessary to define all m documents by at least 1 term. Therefore, starting with the term of highest (lowest) TDV, the term list which would yield $N_{c\,max}$ ($N_{c\,min}$) clusters was established. Having thus obtained the list of index terms which would define the 3 critical points of Figure 3 on page 18 (points A, B, C), additional terms were added (either in ascending or descending order of TDV's) to achieve the portions of Figure 3 connecting points A and B, as well as points C and B. Index terms were added to the maximal and minimal D matrices in discrete blocks, each comprising 20% of the difference either $n - n_{min}$ or $n - n_{max}$. Each modified matrix was used as input to a clustering program, and the corresponding cluster structures generated.

Finally, both text files were used as input to a statistical analysis program. Since one file contained term numbers ranked according to TDV, and the other file contained term numbers ranked according to term generality, the relationship between TDV and t_g could be measured using a Pearson Rank Correlation Coefficient.

5.3 Experiments in Information Retrieval

An information retrieval program was developed to measure the performance of each of the aforementioned cluster structures in terms of precision, recall, and e-measure. For each cluster structure, the entire set of database queries was used (100 queries for NPL, 77 queries for INSPEC), and average values of the performance parameters were obtained. Furthermore, cluster centroid length was varied in order to examine its effect on information retrieval performance. The selection of terms to be used in cluster centroids is based on term generality. Specifically, a cluster centroid of length l will contain the l terms with highest t_g within the cluster of interest. Centroid lengths of 125, 250, 500, and 750 terms were used for each cluster structure. The same matching function is used for both centroid-query and document-query matching.

To demonstrate the validity of this study's cluster structures, a modified version of Yao's theorem for calculating block accesses is used [16]. This modification allows Yao's theorem to be applied to clustered data collections, as was proven in [3]. The modified theorem follows:

Theorem: *Consider a partition of m documents with n_C number of non-overlapping clusters and with each cluster having a size of $|C_j|$ for $1 \leq i \leq n_C$. If k documents are randomly selected from m documents, the probability P_j that cluster C_j will be selected is given by*

$$P_j = \left[1 - \prod_{j=1}^{k} (m_j - i + 1)/(m - i + 1) \right] \quad \text{where } m_j = m - |C_j|.$$

Accordingly, for a randomly generated cluster structure and a query with k relevant documents, the number of target clusters is given by the summation ($P_1 + P_2 + \ldots P_{nc}$). Thus, the number of target clusters for any random strucuture and an associated query are easily determined. Tables 2 and 3 present the average number of target clusters for each C^3M generated cluster structure, as well as the average number of target clusters for the corresponding random cluster structures, $n_{tc\text{-}r}$. As can be seen, the C^3M structures are always better than the random structures. This data is in good agreement with that presented in [3].

Table 2. Comparison of Actual Average Number of Target Clusters (n_{tc}) with the Average Number of Target Clusters for a Random Structure ($n_{tc\text{-}r}$) for INSPEC.

N_c	45	106	155	222	312	477	517	543	563	580	598
n_{tc}	14.21	19.49	20.56	22.81	23.49	24.44	24.57	24.68	24.68	24.64	24.83
$n_{tc\text{-}r}$	20.46	27.70	29.67	31.62	32.80	33.77	34.06	34.21	34.33	34.38	34.44

Table 3. Comparison of Actual Average Number of Target Clusters (n_{tc}) with the Average Number of Target Clusters for a Random Structure ($n_{tc\text{-}r}$) for NPL.

N_c	351	352	354	356	358	360	369	382	393	403	412
n_{tc}	14.53	14.53	14.56	14.56	14.56	14.57	14.66	14.88	14.84	14.87	14.92
$n_{tc\text{-}r}$	18.92	18.92	18.92	18.93	18.93	18.94	18.98	19.08	19.16	19.21	19.24

6 Results

6.1 Relationship between Term Generality and Term Discrimination Value

It was hypothesized that terms which have relatively high term generalities should have correspondingly low term discrimination values. This is because such terms are relatively common and do little to make their associated documents unique. Equivalently, terms with relatively low term generalities were hypothesized to have relatively high term discrimination values.

A Pearson rank correlation test was conducted between t_g and TDV for INSPEC indexing terms. The resulting rank correlation coefficient was 0.33 with a null hypothesis probability of 0.0001. These numbers indicate a definite, moderately strong relationship between t_g and TDV. Specifically, that terms with high TDV have low t_g. It is believed that the value of the rank correlation coefficient is lowered by the large number of data entries (n = 14573) and the fact that a large number of entries assume the same rank. For example, nearly half of all the terms have a term generality of 1. Even so, these results do support the hypothesis that as an indexing term's t_g increases, its TDV decreases.

The Pearson rank correlation test was not conducted for NPL. The indexing vocabulary of this database is highly controlled (e.g. t_g = 1 for over ½ the terms), so it was believed the results of such a test would not be reliable or realistic.

6.2 The Effect of Tunable Indexing on Information Retrieval Performance

The minimum number of terms necessary to define all m documents of the test databases was determined. As described earlier, terms were selected in either ascending or descending order of TDV. As expected, the maximal D matrix consisting of the n_{max} indexing terms with highest term discrimination values produced the maximum number of clusters $N_{c\,max}$. Similarly, the minimal D matrix consisting of the n_{min} indexing terms with lowest term discrimination values produced the minimum number of clusters $N_{c\,min}$. Table 4 summarizes these results for both databases. As additional indexing terms were added to the maximal or minimal D matrix, the resulting number of clusters approaches the natural number of clusters n_c. Table 5 summarizes the pertinent parameters of these cluster structures for INSPEC, while Table 6 does the same for NPL.

Table 4. Summary of $N_{c\,max}$, $N_{c\,min}$, n_{max}, and n_{min} for Databases

Database	n_{max} for maximum number of clusters	n_{min} for minimum number of clusters	N_c max	N_c min
INSPEC	14540	832	598	45
NPL	7472	7453	412	351

Table 5. INSPEC Cluster Parameters

	$N_{c\,min}$ to n_c					n_c				n_c to $N_{c\,max}$	
Number of terms used	832	3580	6328	9076	11824	14573	14568	14561	14554	14547	14540
N_c	45	106	155	222	312	477	517	543	563	580	598
x_d	25.3	29.8	30.5	31.2	31.9	32.5	31.0	29.7	28.5	27.7	26.9
t_g	385.6	105.5	61.0	43.6	34.3	28.3	26.9	25.8	24.8	24.1	23.5
Average cluster size	281.9	119.7	81.8	57.1	40.7	26.6	24.5	23.4	22.5	21.9	21.2

Table 6. NPL Cluster Parameters

	N_c to n_c					n_c			n_c to $N_{c\,max}$		
Number of terms used	7453	7461	7469	7477	7485	7491	7488	7484	7480	7476	7472
N_c	351	352	354	356	358	360	369	382	393	403	412
x_d	20.0	20.0	20.0	20.0	20.0	20.0	19.5	18.9	18.5	18.0	17.5
t_g	30.6	30.6	30.5	30.5	30.5	30.5	29.7	28.8	28.2	27.5	26.8
Average cluster size	32.6	32.5	32.3	32.1	32.0	31.8	31.0	29.9	29.1	28.4	27.7

It was postulated that a cluster structure in which documents are better distinguished from one another would produce better information retrieval performance. Furthermore, it was thought that this cluster structure would be produced by using the maximal D matrix

containing the n_{max} index terms with highest TDV, resulting in $N_{c\,max}$ clusters. The experimental data validate this postulation for both INSPEC and NPL databases. Table 7 summarizes precision values obtained from the INSPEC experiments using a centroid of length 250 terms, and N_c values of $N_{c\,min}$, $N_{c\,max}$ and n_c. To ensure the validity of comparison between these three cluster structures, the precision values presented are associated with the number of target clusters necessary to access a constant number of documents. Approximately 10% of the database is selected, since it has been shown that precision values begin to saturate at this point[3]. Also, since precision is defined as the ratio of retrieved relevant documents to the total number of documents retrieved, meaningful comparison of precision values requires fixing the number of retrieved documents. This is consistent with many IR systems which offer the option of selecting the number of documents to be returned to the user. Accordingly, 1200 documents are selected for INSPEC. The number of target clusters necessary to access 1200 documents for $N_{c\,min}$, n_c and $N_{c\,max}$ are 5, 45, and 57, respectively. Table 7 provides precision values for INSPEC when 10, 20, and 30 documents are returned, while Table 8 provides INSPEC recall values when 10, 20, and 30 documents are returned. Table 9 presents e-measure data for INSPEC when $\beta = 1$ (i.e. equal importance is given to precision and recall), and 10, 20, and 30 documents are returned. Tables 10 through 12 provide the same data for NPL. The NPL data is based on the number of target clusters necessary to access 1100 documents, again approximately 10% of the database. The number of target clusters necessary to access 1100 documents are 34, 35, and 40 for N_{cmin}, n_c, and $N_{c\,max}$, respectively.

Table 7. INSPEC Precision Values with 10% of the Documents Accessed

Number of Documents retrieved	10	20	30
N_c min	.258	.210	.174
n_c	.288	.224	.184
N_c max	.299	.228	.190

Table 8. INSPEC Recall Values with 10% of the Documents Accessed

Number of Documents retrieved	10	20	30
N_c min	.082	.129	.157
n_c	.095	.142	.169
N_c max	.100	.150	.182

Table 9. INSPEC e-measure Values ($\beta = 1.0$)

Number of Documents retrieved	10	20	30
N_c min	.890	.860	.850
n_c	.870	.850	.840
N_c max	.860	.840	.840

Table 10. NPL Precision Values with 10% of the Documents Accessed

Number of Documents retrieved	10	20	30
N_c min	.240	.179	.145
n_c	.247	.189	.152
N_c max	.246	.193	.162

Table 11. NPL Recall Values with 10% of the Documents Accessed

Number of Documents retrieved	10	20	30
N_c min	.144	.192	.225
n_c	.145	.199	.230
N_c max	.144	.200	.240

Table 12. NPL e-measure Values ($\beta = 1.0$)

Number of Documents retrieved	10	20	30
N_c min	.850	.840	.850
n_c	.850	.840	.840
N_c max	.850	.830	.830

Although the data in Tables 7 through 12 show improved information retrieval performance parameters for the cluster structure with $N_c = N_{c\,max}$, the improvement does not at first glance seem significant. This is especially true for the NPL data. The data, however, may be viewed in another manner which more clearly demonstrates the improvement in information retrieval performance. Specifically, for each cluster structure ($N_{c\,max}$, n_c, $N_{c\,min}$), one should consider how many documents must be searched to achieve a given level of precision, recall, or e-measure. This will be the number of target clusters multiplied by the average cluster size. Table 13 shows for INSPEC the number of target clusters and the associated documents required to achieve a precision value of .275, when 10 documents are returned and a centroid of 250 terms is used. Table 14 contains the same information for NPL, except that a precision value of .200 is used since .275 is not attainable for all NPL cluster structures. Table 13 shows a 28.85% decrease in the number of searched documents from the case where $N_c = n_c$ to the case in which $N_c = N_{c\,max}$. Similarly, Table 14 shows a 22.46% decrease for NPL.

Table 13. Target Clusters and Associated Documents Required to Achieve Precision of .275 (INSPEC)

number of clusters	number of target clusters	number of documents
$N_{c\,min}$	11	3100
n_c	19	506
$N_{c\,max}$	17	360

Table 14. Target Clusters and Associated Documents Required to Achieve Precision of .200 (NPL)

number of clusters	number of target clusters	number of documents
$N_{c\ min}$	9	293
n_c	9	285
$N_{c\ max}$	8	221

Although not presented here, the trends shown in Tables 7 through 14 are immune to varying centroid lengths (centroid lengths of 125, 250, 500, and 750 terms were used). Similarly, although the data presented has focused on only 3 cluster structures ($N_{c\ min}$, n_c, $N_{c\ max}$), all trends in precision, recall and e-measure values are consistent over the intermediate cluster structures. This supports the hypothesis that as documents within the collection become increasingly unique, IR performance improves, and as the documents become less unique, IR performance degrades.

6.3 Effect of Tunable Indexing on the Number of Target Clusters

It was hypothesized that as cluster structure improves n_{tc}/rd (or n_{tc}/N_c) should decrease, and that this ratio should be minimal for the case where $N_c = N_{c\ max}$. Also, it was believed that as the value of n_{tc}/rd decreases (for all cases, with both INSPEC and NPL, rd is found to be more limiting than N_c, therefore only n_{tc}/rd will be used: INSPEC rd = 33.0 documents/query, while for NPL rd = 20.8 documents/query), the IR performance of the associated cluster structure should increase. Contrary to this hypothesis, however, it is

found that as the number of clusters increases, the ratio n_{tc}/rd also increases. Furthermore, the highest value of n_{tc}/rd is associated with the cluster structure giving the best IR performance. Tables 15 and 16 summarize this data for INSPEC and NPL, respectively.

6.4 Effect of Tunable Indexing on Indexing Term Distribution

It was hypothesized that c_g/t_g (or c_g/N_c) should decrease as N_c increases, and that the minimum value of c_g/t_g (in all cases, for both INSPEC and NPL, c_g values are limited by t_g rather than N_c: INSPEC t_g = 28.29, NPL t_g = 30.45) would be associated with the cluster structure providing the best IR performance. Again, however, it was found that c_g/t_g increases as the number of clusters increases, and the maximum value of c_g/t_g corresponds to the cluster structure yielding the best IR performance. Table 17 summarizes these results for INSPEC, while Table 18 summarizes for NPL.

Table 15. Summary of INSPEC Target Cluster Data

	$N_{c\,min}$ to n_c					n_c			n_c to $N_{c\,max}$		
N_c	45	106	155	222	312	477	517	543	563	580	598
n_{tc}	14.21	19.49	20.56	22.81	23.49	24.44	24.57	24.68	24.68	24.64	24.83
n_{tc}/rd	.43	.59	.62	.69	.71	.74	.74	.75	.75	.75	.75

Table 16. Summary of NPL Target Cluster Data

	$N_{c\,min}$ to n_c					n_c		n_c to $N_{c\,max}$			
N_c	351	352	354	356	358	360	369	382	393	403	412
n_{tc}	14.53	14.53	14.56	14.56	14.56	14.57	14.66	14.88	14.84	14.87	14.92
n_{tc}/rd	.68	.68	.70	.70	.70	.70	.70	.71	.71	.71	.72

Table 17. Summary of INSPEC Cluster Generality Data

	$N_{c\,min}$ to n_c					n_c			n_c to $N_{c\,max}$		
N_c	45	106	155	222	312	477	517	543	563	580	598
c_g	6.08	8.88	10.3	11.96	13.39	15.12	15.50	15.67	15.86	15.96	16.09
c_g/t_g	.21	.31	.36	.42	.47	.53	.55	.55	.56	.56	.57

Table 18. Summary of NPL Cluster Generality Data

	$N_{c\,min}$ to n_c					n_c			n_c to $N_{c\,max}$		
N_c	351	352	354	356	358	360	369	382	393	403	412
c_g	14.84	14.84	14.85	14.86	14.88	14.90	15.04	15.26	15.43	15.55	15.64
c_g/t_g	.49	.49	.49	.49	.49	.49	.49	.50	.51	.51	.51

7 Conclusions and Suggestion for Future Research

The Pearson Rank Correlation Coefficient test conducted on the INSPEC indexing vocabulary supports the theory that a moderately strong relationship exists between an indexing term's term generality and its term discrimination value. Results suggest that as term generality increases, term discrimination value decreases. Since this finding is somewhat contradictory to previous works [14], further investigation is warranted. It would be of specific interest to test the degree of this relationship independently for each of the three categories of indexing terms: those with high, low, and nearly zero TDV.

It has been shown that the structure of a clustered document database can be predictably controlled through careful selection of the indexing vocabulary. The process of selecting indexing terms based upon their individual term discrimination values (tunable indexing) allows the number of clusters to be varied from a minimum value to a maximum value.

The minimum number of clusters exists when the D matrix is defined by the indexing terms having the lowest term discrimination values. As anticipated, the associated information retrieval performance, measured in terms of precision, recall, and e-measure, is the poorest of all observed cluster structures. The number of documents which must be searched in order to achieve a given level of precision/recall exceeds that associated with both the natural cluster structure, and the cluster structure containing the maximum number of clusters. Furthermore, in accessing a constant number of documents, this structure provides the lowest values of precision and recall, and the highest e-measure value.

The maximum number of clusters exists when the D matrix is defined by the indexing terms having the highest term discrimination values. Again as expected, this cluster structure provides the best information retrieval performance when compared to the other structures. In accessing a constant number of documents, this structure yields much better values of precision, recall and e-measure, as compared to structures with fewer clusters. In order to achieve a given level of precision, recall, or e-measure, the structure containing the maximum number of clusters requires substantially fewer clusters to be accessed, greatly reducing the number of documents which must be searched.

To conclusively show that tunable indexing is capable of effecting information retrieval performance, it would be worthwhile to dictate the number of clusters created by the C^3M program independent of the tunable indexing process. In this way, one could determine if the variation in information retrieval performance is at least partially attributable to simply varying the number of clusters.

It was hypothesized that the cluster structure yielding the best observed information retrieval performance would have the lowest ratio of actual average target clusters to either total number of clusters, or average number of relevant documents per query. This, however, did not prove to be true. In fact, the highest ratio was associated with the cluster structure giving the best information retrieval performance, and the ratio increases slightly with increasing number of clusters. It was also believed that the ratio of average cluster generality to either total number of clusters, or average term generality would reach its minimal value for the cluster structure associated with the best observed

information retrieval performance. Again, this belief was proven incorrect, the ratio increases slightly with increasing number of clusters and achieved its maximum value when information retrieval performance peaked.

All cluster structures were created using only those indexing terms selected during the tunable indexing process. However, there was never such a restriction applied to the creation of queries or cluster centroids; such a restriction was felt to be too artificial, especially for queries. In retrospect, applying such a restriction to the centroids may not have compromised the realism of the research and may have produced different results. It would be worthwhile to investigate this.

Appendix 1: Sample D matrix and Example Similarity Calculation

$$D = \begin{array}{c|cccccc|c} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & \\ \hline & 1 & 3 & 0 & 0 & 2 & 0 & d_1 \\ & 0 & 2 & 0 & 0 & 2 & 1 & d_2 \\ & 1 & 3 & 0 & 1 & 0 & 0 & d_3 \\ & 2 & 0 & 3 & 2 & 1 & 2 & d_4 \\ & 0 & 0 & 2 & 1 & 0 & 0 & d_5 \end{array}$$

m = number of documents = 5
n = number of index terms = 6

depth of indexing for document 1 = $x_{d1} = 3$ ($x_{d2} = 3$, $x_{d3} = 3$, $x_{d4} = 5$, $x_{d5} = 2$)
average depth of indexing = $x_d = 3.2$
term generality for term 1 = $t_{g1} = 3$ ($t_{g2} = 3$, $t_{g3} = 2$, $t_{g4} = 3$, $t_{g5} = 3$, $t_{g6} = 2$)
average term generality = $t_g = 2.7$

$$s(D_1, D_2) = \cos(D_i, D_j) = \frac{D_i \bullet D_j}{\| D_i \| \| D_j \|}$$

$$= \frac{(1 \times 0) + (3 \times 2) + (0 \times 0) + (0 \times 0) + (2 \times 2) + (1 \times 0)}{(1 + 9 + 4) \times (4 + 4 + 1)}$$

$$= 10 / 126 = \mathbf{.079}$$

Appendix 2: Example C matrix

For the generation of the example C matrix the following document description, D, matrix will be used.

$$D = \begin{array}{c|cccccc|c} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & \\ \hline & 1 & 0 & 1 & 0 & 0 & 1 & d_1 \\ & 0 & 0 & 1 & 1 & 1 & 0 & d_2 \\ & 1 & 1 & 0 & 1 & 0 & 0 & d_3 \\ & 0 & 1 & 1 & 0 & 1 & 1 & d_4 \\ & 1 & 0 & 1 & 0 & 1 & 0 & d_5 \end{array}$$

From the D-matrix we see that $m = 5$ and $n = 6$.
To obtain the C-matrix we use the following formula:

$$c_{ij} = \alpha_i \times \sum_{k=1}^{n} (d_{ik} \times \beta_k \times d_{jk}), \quad 1 \le i, j \le m,$$

where α_i and β_k are the reciprocals of the i^{th} row sum and the k^{th} column sum, respectively.

Accordingly, c_{51} is determined as follows:

$$\alpha_5 = 1/3, \ \beta_1 = 1/3, \beta_2 = 1/2, \beta_3 = 1/4, \beta_4 = 1/2, \beta_5 = 1/2, \beta_6 = 1/2.$$

$$c_{51} = 1/3 \times (1 \times 1/3 \times 1 + 0 \times 1/2 \times 0 + 1 \times 1/4 \times 1 + 0 \times 1/2 \times 0 + 1 \times 1/2 \times 0 + 0 \times 1/2 \times 1)$$

$$c_{51} = \mathbf{.195}$$

The resulting C-matrix follows:

$$C = \begin{vmatrix} .362 & .083 & .111 & .250 & .194 \\ .125 & .375 & .250 & .125 & .125 \\ .111 & .167 & .444 & .167 & .111 \\ .187 & .063 & .125 & .438 & .187 \\ .194 & .083 & .111 & .250 & .362 \end{vmatrix}$$

To generate the cluster structure we proceed as follows:

Recall that δ_i (the coupling coefficient for document i) = c_{ii} from the C matrix.
Therefore, $\delta_1 = .362$, $\delta_2 = .375$, $\delta_3 = .444$, $\delta_4 = .438$, and $\delta_5 = .362$.
Also recall that ψ_i (the decoupling coefficient for document i) = $1 - \delta_i$.
Therefore, $\psi_1 = .638$, $\psi_2 = .625$, $\psi_3 = .556$, $\psi_4 = .562$, and $\psi_5 = .638$.

To select cluster seeds, we must determine the cluster seed powers, P_i, for all documents:

$$P_i = \delta_i \times \psi_i \times \sum_{j=1}^{n} d_{ij}.$$

Therefore, $P_1 = .693$, $P_2 = .703$, $P_3 = .741$, $P_4 = .985$, and $P_5 = .693$.

Since it can be shown that $n_c = \delta \times m$ ($\delta = \sum_{i=1}^{m} \delta_i/m$), $n_c = .3962 \times 5 = 1.981 \cong 2$.

So we must have 2 clusters and of course 2 cluster seeds. The documents having the 2 highest seed powers become the cluster seeds, d_4 and d_3.

To cluster the remaining 3 documents we refer back to the C matrix. Considering d_1, it is seen that $c_{14} > c_{13}$ (d_1 is covered better by d_4 than by d_3), so d_1 is clustered with d_4. Using this procedure the following clusters are obtained (seeds are indicated in bold).

Cluster 1 = {$\mathbf{d_3}$, d_2} Cluster 2 = {$\mathbf{d_4}$, d_1, d_5}

Appendix 3: Term Weighting Components

Term Frequency Components		Meaning
b	1.0	binary weight equal to 1.0 for terms present in vector (term frequency is ignored)
t	*tf*	raw term frequency
n	$.5 + .5(tf / \max tf)$	augmented normalized term frequency

Collection Frequency Components		Meaning
x	1.0	no change in weight
f	$\log N/n$	inverse collection frequency, where N = number of documents in collection and *n* = number of documents to which a term is assigned.
p	$\log(N - n/n)$	probabilistic inverse collection frequency factor

Normalization Components		Meaning
x	1.0	no normalization
c	$1/ (\sum_{vector} w_i^2)^{1/2}$	cosine normalization, where w_i is the weight of the i^{th} term

More explanation for the term weight components is provided in [11].

References

1. Can, F., On the efficiency of best-match cluster searches. *Information Processing & Management,* Vol. 30, No. 3, (1994) pp. 343-361.

2. Can, F., Incremental clustering for dynamic information processing. *ACM Transactions on Information Systems,* Vol. 11, No. 2, April 1993, pp. 143-164.

3. Can, F., Ozkarahan, E. A., Concepts and effectiveness of the cover-coefficient based clustering methodology for text databases. *ACM Transactions on Database Systems*, Vol. 15, No. 4, December 1990, pp. 483-517.

4. Can, F., Ozkarahan, E. A., Computation of term/document discrimination values by use of the cover coefficient concept. *Journal of the American Society for Information Science*, Vol. 38, No. 3, May 1987, pp. 171-183.

5. Can, F., Ozkarahan, E. A., Concepts of the cover-coefficient-based clustering methodology. In *Proceedings of the 8th Annual ACM-SIGIR Conference*, June 1985, ACM, New York, pp. 204-211.

6. Crawford, R. G., The computation of discrimination values. *Information Processing & Management*, Vol. 11 (1975), pp. 249-253.

7. Crouch, C. J., An analysis of approximate versus exact discrimination values. *Information Processing & Management*, Vol. 24, No. 1 (1988), 5-16.

8. El-Hamdouchi, A., Willett, P., An improved algorithm for the calculation of exact term discrimination values. *Information Processing & Management*, Vol. 24, No. 1 (1988), pp. 17-22.

9. Faloutsos, C., Access methods for text. *ACM Computing Surveys*, Vol. 17, No. 1, March 1985, pp. 49-74.

10. Salton, G. (1989), *Automatic text processing: The transformation, analysis, and retrieval of information by computer*. Reading, MA: Addison-Wesley.

11. Salton, G., Buckley, C., Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, Vol. 24, No. 5 (1988), pp. 513-523.

12. Salton, G., McGill, M. J. (1983), *Introduction to modern information retrieval*. New York: McGraw-Hill.

13. Salton, G., Wong, A., Yang, C. S., A vector space model for automatic indexing. *Communications of the ACM*, Vol. 18, No. 11, November 1975, pp. 613-620.

14. Willett, P., An algorithm for the calculation of exact term discrimination values. *Information Processing & Management*, Vol. 21, No. 3 (1985), pp. 225-232.

15. van Rijsbergen, C. J. (1979), *Information retrieval* (2nd edition). London: Butterworths.

16. Yao, S. B., Approximating block accesses in database organizations. *Communications of the ACM*, Vol. 27, No. 20, April 1977, pp. 260-261.