Computer Science and Systems Analysis

Computer Science and Systems Analysis

Technical Reports

Miami University *Year* 1988

Experiments on Tunable Indexing

Fazli Can* Esen Ozkarahan†

*Miami University, commons-admin@lib.muohio.edu

†Miami University, commons-admin@lib.muohio.edu

MIAMI UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE & SYSTEMS ANALYSIS

TECHNICAL REPORT: MU-SEAS-CSA-1988-004

Experiments on Tunable Indexing

Fazli Can and Esen Ozkarahan

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

Experiments on Tunable Indexing

by

Fazli Can
Systems Analysis Department
Miami University
Oxford, Ohio 45056

Esen Ozkarahan
Computer Science
Arizona State University
Tempe, Arizona 85287

Working Paper #88-004

April 1988

EXPERIMENTS ON TUNABLE INDEXING

Fazli CAN
Dept. of Systems Analysis
Miami University
Oxford, OH 45056

Esen A. OZKARAHAN
Dept. of Computer Science
Arizona State University
Tempe, AZ 85287

ABSTRACT

The effectiveness and efficiency of an Information Retrieval (IR) system depends on the quality of its indexing system. Indexing can be used in inverted file systems or in cluster-based retrieval. In this article, a new concept called tunable indexing is introduced. With tunable indexing the number of clusters of a document clustering system can be varied to any desired value. Also covered are the computation of Term Discrimination Value (TDV) with the cover coefficient (CC) concept and its use in tunable indexing. A set of experiments has shown the consistency between the CC based TDVs and the TDVs determined with the known methods. The main use of tunable indexing has been observed in determining the parameters of a clustering system.

1. INTRODUCTION

Indexing can roughly be described as a process which determines descriptors of documents in a database. Although the definition of indexing is simple, the task of indexing is not. Document descriptors or index terms, or terms for short, cannot be selected arbitrarily since the performance of an IR system is critically dependent on them [LANC75, p. 138]. Once we form an indexing system we can then (a) define the contents of documents, (b) determine the topics of a document database, and (c) find the relatedness of user request (query) and individual documents of a database.

Indexing is used both in cluster- and inverted file-based IR systems [OZKA86a, SALT83, VANR79]. In cluster-based retrieval systems, clustering (i.e., the process of putting similar documents into the same group) enables efficient operation of text retrieval systems inclusive of full text search systems [OZKA84, OZKA86a]. The partitioning (clustering) of document space also helps increase the effectiveness of IR systems [SALT83].

IR systems use the two generally accepted measures of effectiveness which are recall and precision. Recall is the proportion of retrieved relevant documents with respect to the total number of relevant documents in the database. Precision is the proportion of retrieved relevant documents with respect to the total number of retrieved documents. The users of an IR system can be sensitive to either recall or precision or both. Therefore, an indexing subsytem in an IR system should be tuned according to the needs of its users. Consequently, in a recall-sensitive user environment we pick terms with broad coverage (i.e., terms which appear in many documents). In the precision-sensitive case we pick specific terms (i.e., terms which appear in small number of documents) for the description of documents. In the mixed mode of expectations the indexing system must find a good combination of general and specific terms.

Indexing can be performed automatically and manually. Manual indexing depends on experts [LANC75, SALT83]. In this article we are based on automatic indexing, i.e., index terms are extracted from the documents by means of computer programs.

Various contributions to the ongoing research on the theory of indexing can be found in sources such as *Journal of the American Society for Information Science, Information Processing* and *Management, Journal of Documentation*, and ACM SIGIR conferences. The survey in [BORK77] provides a brief overview of the indexing theories of Jonker, Heilprin, Landry, and Salton and his co-workers. A detailed illustration of the theory of Salton and his co-workers can be found in [SALT75a, SALT75b]. Various probabilistic approaches for indexing are proposed in [YU76, SALT81, COOP78]. The term relevance measure (work of Robertson, Spark Jones, Van Rijsbergen, Harper, etc.), which is another probabilistic approach, can be found in [VANR79]. An artificial intelligence approach to the indexing problem is discussed in [KOL83]. For self-adapting document approaches, where indexing is determined by inquirers' relevance assessments, the reader can refer to [GORD85, BOOK86]. A new, yet computationally expensive, approach is the generalized vector space model [WONG87]. It expresses the single terms of indexing in terms of atomic concepts.

In Section 2 we will introduce the TDV concept. The computation of TDVs with the CC concept and the tunable indexing concept will be introduced in Section 3. We then introduce a new methodology called tunable indexing. It involves selection of proper index terms that will yield the desired number of clusters within a database. The utility of tunable indexing will also be discussed.

Previously we have shown [CAN87a] that CC approach for TDV calculation yields results compatible with an approximation technique [SALT75a] which is implemented in [CRAW75]. In that study the compatibility experiments of the two approaches were done using only one document description, D, matrix. In Section 4, we will present the results of the compatibility experiments performed with sixteen larger D matrices. The experiments show that our approach produces compatible results not only with respect to the approximation technique but also with the exact technique [WILL85]. Furthermore, the experiments show that the degree of compatibility between the CC approach and the exact approach is higher than that of the approximation approach with the exact approach. The various experiments will show that the tunable indexing concept works, i.e., it produces the desired number of clusters.

2. THE TDV CONCEPT

The TDV concept is introduced in [SALT75a, SALT83]. In the TDV concept the significance of an index term is measured by its effect on distinguishability of documents.

To illustrate the concept and its computation let us first introduce our notation. Let D be an m by n document database description matrix. m indicates the cardinality of the database: $\{d_1, d_2, \ldots, d_m\}$, and n is the cardinality of the indexing vocabulary $T= \{t_1, t_2, \ldots, t_n\}$. Accordingly, an entry, d_{ij} $(1\leq i \leq m, 1 \leq j \leq n)$ of D matrix indicates the importance of t_j in d_i. Indexing can either be binary or weighted. In the case of binary indexing d_{ij}

can be either 1 or 0 indicating the existence of t_j in d_i or non-existence of t_j in d_j, respectively. In weighted indexing d_{ij} represents weight of the term such as the frequency of t_j in d_i. Obviously in a D matrix, a row with all zeros (i.e., an undefined document) or a column with all zeros (i.e., an unused term) cannot exist.

In calculating TDV the average similarity of documents, Q' can be used. It is

$$Q' = 2 / \left[m \times (m-1) \right] \times \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} s(d_i, d_j) \qquad (0 \le Q' \le 1, 0 \le s \le 1) \quad (2.1)$$

Q' is also referred to as "average document space density." In Eq. (2.1), $s(d_i, d_j)$ indicates the similarity between d_i and d_j. For the similarity measure we may use the cosine similarity coefficient

$$s(d_i, d_j) = \left[\sum_{k=1}^{n} d_{ik} \times d_{jk} \right] \times \left[\sum_{k=1}^{n} d_{ik}^2 \times \sum_{k=1}^{n} d_{jk}^2 \right]^{-\frac{1}{2}} \quad (2.2)$$

To reduce the computational cost of Q' we may use the database centroid, G, instead of individual documents where the individual entries of G, g_j $(1 \le j \le n)$ can be defined as $(d_{1j} + d_{2j} + \ldots + d_{2m}) / m$, i.e., the average weight of t_j in the database. We can then define approximate document space density as follows [SALT75a, SALT83]:

$$Q = \left[\sum_{i=1}^{m} s(d_i, G) \right] \times m^{-1} \qquad (0 < Q \le 1 \text{ for } 0 < s \le 1) \quad (2.3)$$

Document databases with greater separation of document description vectors will have lower Q (density) values. The reverse is true for document description vectors with less separation, i.e., document databases with similar documents would lead to higher Q values.

Let us consider a term t_j; if we delete t_j, this will change the indexing vocabulary, T, the document space density, Q, and the database centroid, G. The new G, G_j, would be $(g_1, g_2, \ldots, g_{j-1}, g_{j+1}, \ldots, g_n)$. The new document space density Q_j will be calculated using Eq. (2.3) and replacing G by G_j.

The difference $(Q_j - Q)$ reflects the change due to deletion of term t_j. If the assignment of t_j separates the documents from each other, then it will decrease the document space density (Q). Consequently, the removal of t_j makes the documents closer to each other by increasing the document space density, Q_j and causing $Q_j > Q$. This will make $(Q_j - Q)$ greater than zero. The difference $(Q_j - Q)$ is referred to as the discrimination value of t_j, TDV_j, [SALT75a, SALT83].

A TDV has the following properties: (a) $TDV_j > 0$ for a good discriminator t_j (i.e., the assignment of t_j makes the documents more distinguishable from each other); (b) $TDV_j = 0$ for an indifferent term t_j (i.e., the assignment of t_j does not contribute to the distinguishability of documents); (c) $TDV_j < 0$ for a poor discriminator t_j (i.e., a term which makes the documents less distinguishable from each other).

The use of Eq. (2.1) yields exact TDVs; hence, it will be called "exact similarity method" and abbreviated as ESM. Eq. (2.3) leads to approximate TDVs; therefore, it will be called "approximate, centroid method" and will be abbreviated as ACM.

Willett [WILL85] and Crawford [CRAW75] defined efficient algorithms for the implementation, respectively, of ESM and ACM using Eq. (2.2). The experiments discussed in Section 4 will show that the TDVs computed by the CC concept and ESM have better consistency than that of ACM with ESM. This is significant since ACM tries to approximate ESM.

In the IR literature TDVs have been used for various purposes [SALT75a, SALT83, SALT86]: some of which are: (a) to construct a document weighting function $d_{ij} = TDV_j \times f_{ij}$, where f_{ij} is the number of occurrences of t_j in d_i; (b) to construct "term phrases" and "thesaurus classes" which increase effectiveness of an IR system.

Our specific purposes for using the TDV concept are (a) to construct a document weighting function $d_{ij} = DSV_i \times TDV_j \times f_{ij}$ where DSV is a concept similar to TDV indicating the document significance value of d_i; (b) to implement tunable indexing in connection with the number of clusters indicated by the CC concept. Our experience to date have indicated that the DSV approach is helpful in improving the effectiveness of an IR system [URAL86, CAN87b].

3. CALCULATION OF TDVs BY THE CC CONCEPT AND TUNABLE INDEXING

The CC concept has been described in various publications [CAN83, CAN84, CAN85a, CAN85b]. For readers who are unfamiliar with this concept we have provided an introduction in the appendix.

3.1 TDV Calculation by CC

We can use CC related variables in the computation of TDVs. Consider the notion of document decoupling. It is easy to realize that the concepts of document space density (Q) and average decoupling of documents (δ or number of clusters, n_c) are inverse to each other. Table 1 shows the interpretation of the related quantities with respect to TDV (where δ and δ_h are the average decoupling of documents before and after the deletion of t_h; similarly n_c and n_{ch} are the number of clusters in the database).

Table 1. Effects of the type of index term(t_h) on the values of Q, δ and (n_c).

	Quantity		
Type of t_h	Q vs Q_h	δ vs δ_h	n_c vs n_{ch}
Good discriminator ($TDV_h > 0$)	$Q < Q_h$	$\delta > \delta_h$	$n_c > n_{ch}$
Indif. discriminator ($TDV_h = 0$)	$Q = Q_h$	$\delta = \delta_h$	$n_c = n_{ch}$
Poor discriminator ($TDV_h < 0$)	$Q > Q_h$	$\delta < \delta_h$	$n_c < n_{ch}$

According to the CC concept, a TDV is defined as ($n_c - n_{ch}$), i.e., the change in the number of clusters after the deletion of term t_h [CAN85a]. In [CAN87a] it is shown that an exact TDV can be computed according to the CC concept as follows:

$$TDV_h = \sum_{i=1}^{f_h} \left[\delta_i - \alpha_i^h \times \left[\frac{\delta_i}{\alpha_i} - d_{ih}^2 \times \beta_h \right] \right] \tag{3.1}$$

where f_h is the cardinality of D_h and $D_h = \{d_i | d_i \in D \wedge d_{ih} \neq 0\}$, i.e., D_h is the set of document containing the term t_h. As we have defined in the appendix, α_i and β_h indicate the reciprocal of row-i and column-h sums of the D matrix respectively. And $\alpha_i^h = (\alpha_i^{-1} - d_{ih})^{-1}$, i.e., α_i^h is the reciprocal of row-i sum excluding d_{ih}.

For a binary D matrix $d_{ih}^2 = d_{ih}$ and $\alpha_i^{-1} - (\alpha_i^h)^{-1} = 1$ if $d_{ih} = 1$, for $1 \leq i \leq m$, $1 \leq h \leq n$. Hence, for a binary D matrix, Eq. (3.1) will take the following form:

$$TDV_h = \sum_{i=1}^{f_h} \alpha_i^h \times (\beta_h - \delta_i) \tag{3.2}$$

It should be mentioned that the TDV values obtained with the CC concept are exact values. This is because their calculation does not involve any approximation. In this paper, the CC method for TDV calculation will be referred to as C^2DVM.

In [CAN87a] we have shown that computational complexity of C^2DVM is O(6t)≈ O(t) where t is number of nonzero entries in D matrix. This compares favorably with the computational cost of ESM and ACM [CRAW75, WILL85], which are O(mt) and O(9t), respectively. Even though the order of magnitude scales are the same 6 is less than 9 and much less than m.

3.2 Tunable Indexing: Concept and Implementation

In designing and implementing an IR system we may want to vary various parameters. If we are using clustering we may need to tune average cluster size due to external constraints. In the CC based clustering we can predict the cluster size analytically [CAN85b]. However, for better precision or recall the IR system administrator may want to decrease or increase cluster size. Also, for efficiency of the underlying computer system we may want to optimize cluster(partition) size with respect to paging and memory constraints. In hierarchical clustering, we may also want to control the number of clusters at each level of the hierarchy.

In the tunable indexing scheme, the administrator can specify the number of clusters that he(she) wants to have within a database. Assume that N_c is the number of clusters that the administrator wants to have within a database. (Note that the number of clusters, n_c, computed by the CC concept must be within the range: $1 \leq n \leq \min(m, n)$; this property must also be observed by N_c. Another restriction on N_c will be stated later when we define the tunable indexing algorithm.) According to the analytical relationships observed [CAN87a] between indexing and clustering once we fix N_c then we must vary indexing to obtain the desired partition pattern.

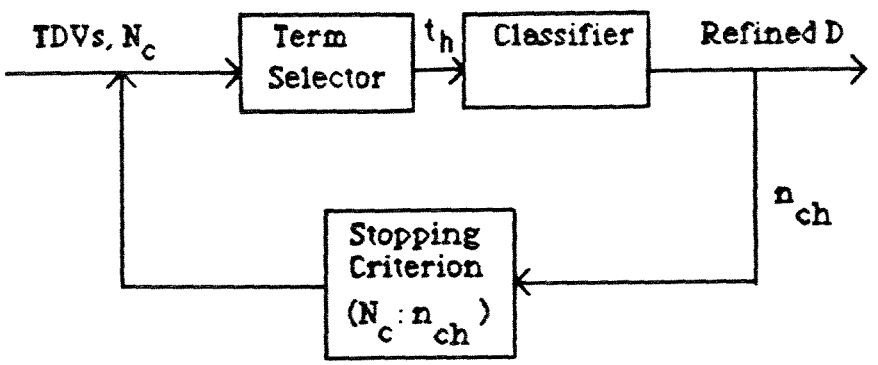

Figure 1. Tunable indexing concept

The system view of our tunable indexing concept is illustrated in Figure 1. The tunable indexing process will be activated whenever N_c is different from n_c which is the number of clusters implied by the original D matrix. By applying tunable indexing we will obtain a refined D matrix that will yield the desired number of clusters: N_c. In the figure, term selector is responsible from the choice of index terms. Classifier is the CC based clustering subsystem. The tuning procedure takes the following forms depending upon the relationship between N_c and n_c (i.e., required and theoretical).

a) $N_c = n_c$: the current description of the D matrix satisfies the administrator's requirement. This means no tuning, hence changing of D, is required.

b) $N_c > n_c$: the current number of clusters resulting from the original D matrix is smaller than the administrator's requirement. In this case the terms which

reduce the number of clusters should be deleted from the index vocabulary T. These terms have negative TDVs. Therefore, term selector will add, one by one, the terms with positive TDVs. In the figure these newly added terms are indicated by t_h. The document vectors extended by t_h will be passed to Classifier. The number of clusters, n_{ch}, implied by the resulting T= $\{t_1, t_2, \ldots t_h\}$ will be passed to the decision maker, which implements the stopping criterion by comparing N_c and n_{ch}. The result is to stop or continue the term selection process. In other words, the system modifies itself, or learns, so that it reaches the desired number, N_c, of clusters.

c) $N_c < n_c$: the current number of clusters resulting from the original D matrix is greater than the administrator's requirement. In this case, the terms which increase the number of clusters should be deleted from T. These terms have positive TDVs.

Let us start with a solution for case (c):

Algorithm for index tuning for the case $N_c < n_c$:

[a] *Compute* TDV_h $(1 \leq h \leq n)$

Sort terms in ascending order according to their TDVs

h= 0

T= ϕ / null set */*

[b] *repeat*

h= h + 1

T= T U $\{t_h\}$ / where t_h is the h'th term of the sorted list */*

until all documents are defined by at least one term

Compute n_{ch} / n_{ch} is the number of clusters in the D_h matrix of size m by h */*

$N_{min} = n_{ch}$

[c] *while $n_{ch} < N_c$ and $h < n$*

h= h + 1

T= T U $\{t_h\}$

compute n_{ch}

endwhile

The tunable indexing algorithm assumes an initial D matrix of size m by n. The initial D matrix is used in the computation of TDVs. Step (b) provides the definition of all documents by at least one term. We must first define all documents and this of course is necessary independent of the value of N_c. At the end of step (b) T contains h terms with the lowest TDVs. Accordingly, the number of clusters resulting from the D matrix would be the minimum possible, N_{min}, number of clusters. As we increase the size of T in step (c) the number of clusters n_{ch} resulting from the changing D matrix will increase. The algorithm will terminate whenever $n_{ch} = N_c$, $n_{ch} > N_c$, or h= n.

The solution for case (b), i.e., $N_c > n_c$, is very similar to the previous solution. However, in step (a) we will sort the terms in descending order, and at the end

of step (b), n_{ch} will be N_{max}, i.e., maximum possible number of clusters. This is because T contains the terms with highest TDVs and, by definition, these are the terms which distinguish the documents most. In this solution the condition of the while statement in step (c) must be "$n_{ch} > N_c$ and $h < n$."

The foregoing explanation indicates the value range $1 \leq N_{min} \leq N_c \leq N_{max} \leq \min(m, n)$ for N_c. It should be noticed that the above algorithms are approximations, since at the time of adding t_h to T there is no guarantee that t_h will be the term with the minimum TDV among the terms added, in the case of $N_c > n_c$, and similarly, the term with the maximum TDV among the terms added in the case of $N_c < n_c$. For n terms there are n! possible permutations for the addition sequence and it is not possible to consider all possible sequences to reach the best addition sequence. However, the foregoing approximation methods seem reasonable and their expected behavior is shown in Figure 2. In this figure h and n_{ch} indicate the cardinality of indexing vocabulary, T, and the number of cluster implied by matrix D_h, respectively. And n_c is the number of clusters in the original D matrix, n is the size of T for original D, N_{max} is the maximum possible number of clusters, and N_{min} is the minimum possible number of clusters. N_{max} and N_{min} are obtained by tunable indexing. Therefore, when we apply the tunable indexing algorithm for N_c is greater than n_c, we are expecting a gradual decrease from N_{max} to n_c as we increase the size of T. For the reverse case, i.e., when N_c is less than n_c, we are expecting a gradual increase from N_{min} to n_c.

The experiments of the next section will show that the tunable indexing algorithms could be used to obtain a D matrix that defines documents in the required detail to obtain desired number of clusters.

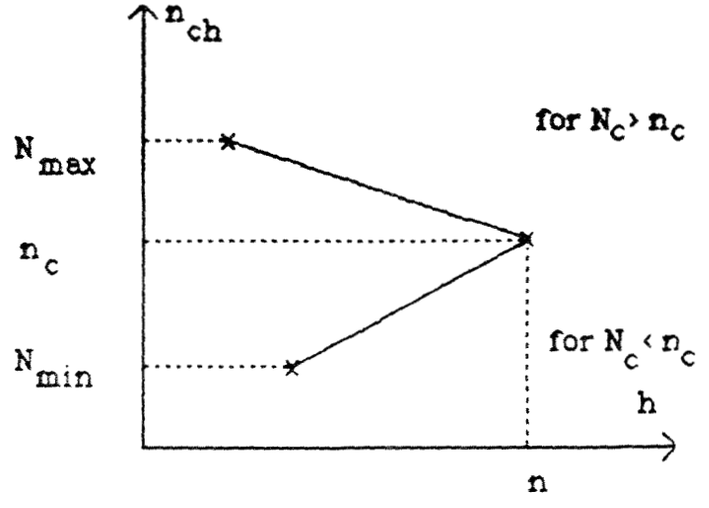

Figure 2. Expected behavior of tunable indexing algorithms

Now let us consider the implementation aspects of tunable indexing in terms of CC based TDVs. Let f_{h+1} be the cardinality of D_{h+1}, where $D_{h+1} = \{d_i | d_i \in D \wedge$

$d_{i,h+1} \neq 0\}$, i.e., D_{h+1} is the set of documents containing the term at the rank h+1. Then the number of clusters, $n_{c,h+1}$, in refined D can be expressed as

$$n_{c,\,h+1} = n_{ch} + \sum_{i=1}^{f_{h+1}} \left[\delta_{i,h+1} - \delta_{i,h} \right] \tag{3.3}$$

Eq. (3.3) can be rewritten as follows, by using the identity given in Eq. (3.5)

$$n_{c,\,h+1} = n_{ch} + \sum_{i=1}^{f_{h+1}} \left[\alpha_{i,h+1} \times \left[\frac{\delta_{i,h}}{\alpha_{i,h}} + d^2_{i,h+1} \times \beta_{h+1} \right] - \delta_{i,h} \right] \tag{3.4}$$

$$\delta_{i,h+1} = \alpha_{i,h+1} \times \left[\frac{\delta_{i,h}}{\alpha_{i,h}} + d^2_{i,h+1} \times \beta_{h+1} \right] \tag{3.5}$$

where $\delta_{i,h}$ and $\delta_{i,h+1}$ indicate the decoupling coefficient of d_i with $T= \{t_1, t_2, \ldots, t_h\}$ and $T= \{t_1, t_2, \ldots, t_{h+1}\}$, respectively. And

$$\alpha_{i,h} = \left[\sum_{j=1}^{h} d_{ij} \right]^{-1} \quad \text{and} \quad \alpha_{i,h+1} = \left[\sum_{j=1}^{h+1} d_{ij} \right]^{-1}$$

For a binary D matrix, Eq. (3.4) becomes

$$n_{c,h+1} = n_{c,h} + \sum_{i=1}^{f_{h+1}} \left[\alpha_{i,h+1} \times \left[\beta_{h+1} - \delta_{i,h} \right] \right] \tag{3,6}$$

A reasonable storage overhead for keeping intermediate δ_i values decreases the computational complexity of tunable indexing [CAN87a].

Notice that Eqs. (3.1) and (3.5), and Eqs. (3.2) and (3.6) are similar and yet not identical to each other. This is because in tunable indexing our concern is the change in the number of clusters when we "extend" the indexing vocabulary of size "h" by the h+1st term of the sorted list. In the TDV case, we calculate the change in the number of clusters when we "decrease" the indexing vocabulary of size "n" by one term.

4. EXPERIMENTAL EVALUATION

The experiments described in this section are performed to verify (a) the CC based TDV calculation methodology (C^2DVM), and (b) tunable indexing. In (a) and (b) we will, respectively, show that C^2DVM produces compatible results as compared to ACM (approximate centroid method), and ESM (exact similarity method) and tunable indexing can be used to obtain an indexing vocabulary for a D matrix that results in the desired number of clusters.

4.1 Document Database and Indexing Policy

In order to perform the experiments we need to start with the initial D matrices which will be used in the calculation of TDVs, and later, in tunable indexing. The document database is the TODS database which contains titles, abstracts, and keywords of 214 articles from the journal called ACM Transactions on Database Systems. This database has been used in our previous studies and its indexing methodology is described in [OZKA86b]. A stem becomes a term if it satisfies the following two conditions: (a) it must not appear in the stop list (containing the frequent words of English language and database literature); (b) the number of documents containing the stem must be within the range f_{min} through f_{max}. The weight d_{ij}, of terms within documents, is determined as the frequency of a document in a specific document (i.e., $d_{ij} = f_{ij}$). Hence, d_{ij} indicates the number of occurrences of t_j in d_i, ($1 \leq i \leq 214$, $1 \leq j \leq n$). In this case, n is a function of f_{min} and f_{max}. The lower f_{min} and higher f_{max} imply the higher value of n. The opposite, i.e., the higher f_{min} and lower f_{max} implies the lower value of n. For experimentation we need to have various D matrices and this is obtained by using different f_{min} (2, 3, 4) and f_{max} (20, 30, 40) values. The characteristics of the D matrices for different f_{min}, f_{max} combinations are shown in Table 2.

Table 2. Statistics of the D matrices generated according to the frequency constraints: (f_{min}, f_{max})

f_{min}, f_{max}	n	t	x_d	t_g	x_{dw}	t_{gw}	n_c
2, 40	1060	7446	34.79	7.02	53.02	10.70	34
3, 40	757	6840	31.96	9.04	49.15	13.90	27
4,40	604	6381	29.82	10.56	46.13	16.34	24
2, 30	1045	6916	32.32	6.62	48.90	10.01	36
3, 30	742	6310	29.49	8.50	45.04	12.99	29
4, 30	589	5851	27.34	9.93	42.01	15.26	26
2, 20	988	5537	25.87	5.60	37.64	8.15	43
3, 20	685	4931	23.04	7.20	33.78	10.55	35
4, 20	532	4472	20.90	8.41	30.75	12.37	30

The meaning of the column headers in Table 2 is as follows. n: |T|, t: the total number of non-zero entries in D, x_d: the average number of distinct terms per document (depth of indexing), t_g: the average number of distinct documents per term (term generality), x_{dw}: the average total weight per document (weighted depth of indexing), t_{gw}: the average total weight per term

(weighted term generality), and n_c: the number of clusters indicated by the CC concept.

Relaxed conditions (i.e., smaller f_{min} and higher f_{max}) lead to larger D matrices in terms of n and t (m is always fixed and equal to 214 in all the experiments). Let us compare $D_{2\text{-}40}$ (where the first subscript, 20, corresponds to f_{min} and second, 40, corresponds to f_{max}) with $D_{4\text{-}20}$. $D_{2\text{-}40}$ is a 214 by 1060 matrix with 7446 non-zero entries. On the other hand, $D_{4\text{-}20}$ is a 214 by 532 matrix with 4472 non-zero entries.

4.2 Experiments for Checking the Consistency of the Three TDV Calculation Methods

The experiments of this section will measure the consistency between CC based TDV calculation method (C^2DVM) with the other known methods, i.e., ESM and ACM. Specifically, the consistency measurements will be performed between

a) C^2DVM and ACM,

b) C^2DVM and ESM, and

c) ACM and ESM

Item (a) measures the compatibility or agreement of the two approaches that involve relatively small number of computations. Items (b) and (c) measure the agreement between these two TDV calculation techniques (C^2DVM, ACM) involving small number of computations and the one (ESM) that involves a high number of computations. In these comparisons, one would normally expect to obtain higher consistency in (c) than the ones obtained for items (a) and (b). This is because ACM is an approximation for ESM. However, the opposite, i.e., higher consistency between C^2DVM and ACM, and C^2DVM and ESM were observed in the experiments that will follow.

In the consistency experiments, we will use sixteen weighted D matrices. Nine of these matrices are described in Table 2. The additional seven D matrices that were generated in tunable indexing are also used in the consistency experiments. The statistics of these additional matrices, which will be referred to as D' matrices, are provided in Table 4.C.

It is obvious that the three approaches may not assign identical TDVs to identical terms. This has been seen to be true even when the TDVs were calculated with two different similarity coefficients using the exact similarity approach [WILL85]. Therefore, to evaluate the consistency, the terms were sorted in descending order according to their TDVs thus giving the rank of 1 and n to the terms with the highest and lowest TDV, respectively.

In the consistency experiments, a bivariate correlation analysis is performed on the experimental data using the ranks of the terms. For this purpose we used the SPSS statistical package [NIE75]. In the bivariate correlation analysis,

we used Spearman's rank order correlation coefficient (denoted as r_s). In this analysis, if the value of r_s is close to zero, then there is little or no linear relationship between the two variables being correlated. Whenever $r_s = 1$, there is a perfect linear relationship between the two variables (in our case, the ranks of the terms in the sorted lists). On the other hand, when $r_s = -1$, there is a perfect inverse relationship. This means that terms which have high ranks in one sorted list tend to have low ranks in the other sorted list.

Associated with the bivariate correlation, a linear regression line is obtained to locate the best-fitting straight line for the two approaches being correlated. This is done using the scattergram facility of SPSS, obtaining a plot and an equation of the regression line as the result. In this article we will provide the equation of the regression line since it also serves as a summary of the scattergram plot. To give an idea the scattergram plot for matrix $D'_{4\text{-}40}$ is given in Figure 3. In this plot an asterisk indicates an observation, and a number (twos, threes, and four) indicates more than one (i.e., two, three, or four) observations at that point. In Figure 2 the horizontal (X) and vertical (Y) axes correspond to ESM and ACM, respectively. Hence, an observation (X_i, Y_i) in the data set indicates that there is a term with ranks X_i and Y_i with respect to ESM and ACM.

The scattergram outputs showed that all three methods are more consistent on the high-ranked terms than they are on the low-ranked terms. This observation shows that the methods are more consistent in determining the TDV of the worst terms, i.e., the terms with low (negative) TDV. (Remember that terms are sorted in descending order according to their TDVs.)

Now let us consider the regression lines. Let R_{C^2DVM}, R_{ACM}, and R_{ESM} denote the ranks obtained by use of the C^2DVM, ACM, and ESM, respectively. In the experimentsthe following regression lines were obtained indicating the relationships among R_{C^2DVM}, R_{ACM}, and R_{ESM}:

$$R_{C^2DVM} = b_1 \times R_{ACM} + a_1$$

$$R_{C^2DVM} = b_2 \times R_{ESM} + a_2$$

$$R_{ACM} = b_3 \times R_{ESM} + a_3$$

where b_i and a_i $(1 \leq i \leq 3)$ are the slope and intercept of the regression lines. Obviously if the results of any two methods are perfectly identical, say C^2DVM and ACM, then we would obtain $R_{C^2DVM} = R_{ACM}$, or in general b_i and a_i will be closer to 1 and 0, respectively $(1 \leq i \leq 3)$. Rather than displaying the parameters of equations, i.e., a_1, a_2, a_3 and b_1, b_2, b_3 in tabular form a plot of them is provided in Figure 4 for easy interpretation.

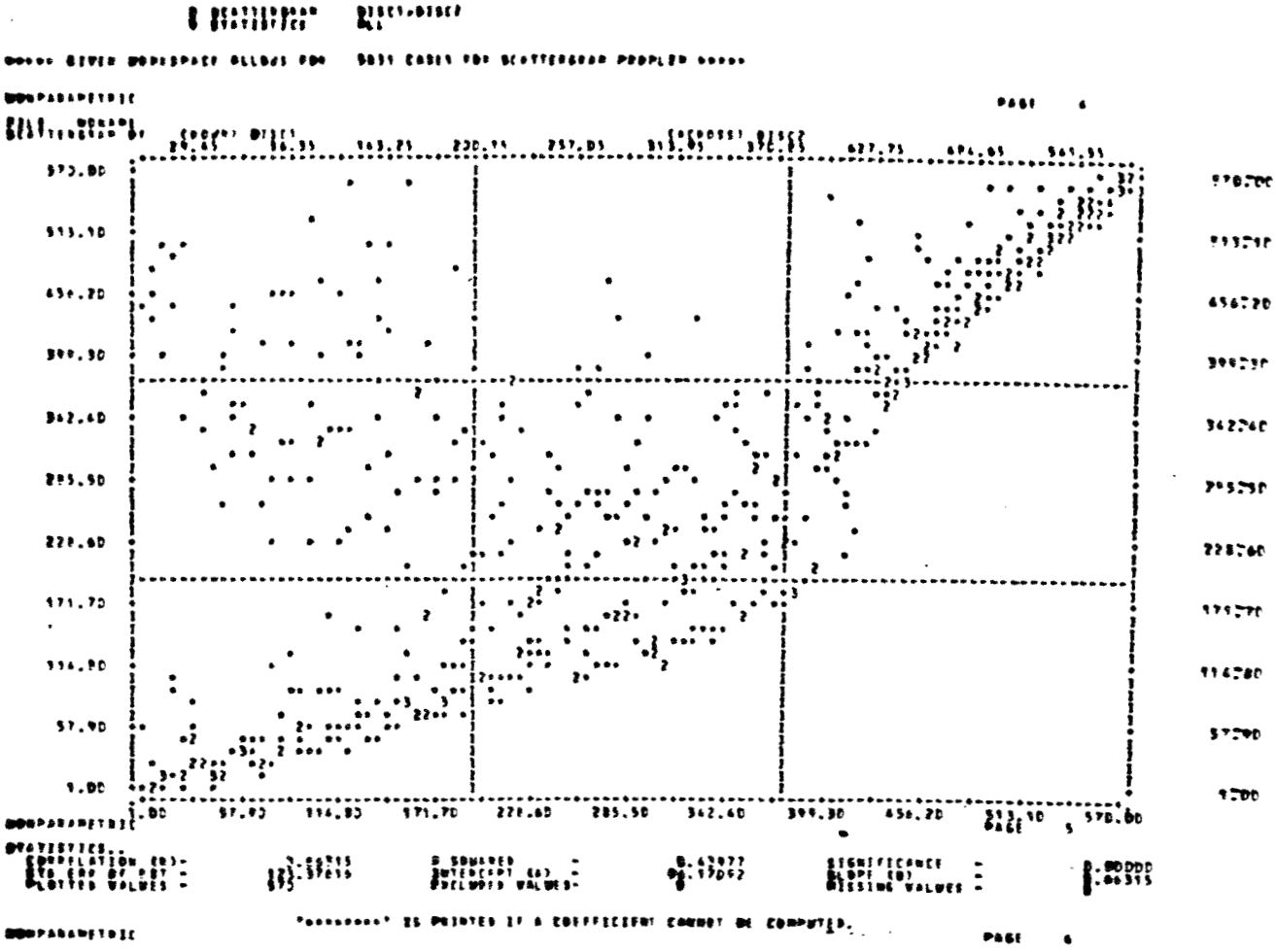

Figure 3. The scattergram which shows the consistency of ACM and ESM for the matrix $D'_{4\text{-}40}$

The validity of each regression line is indicated by the Pearson product-moment correlation value r_i $(1 \leq i \leq 3)$. Since we are using ordinal level data (i.e., the ranks of terms) in our analysis the slope (b_i) of the regression line, Pearson product-moment correlation (r_i), and Spearman's correlation coefficient (r_{si}) yield identical values. The analytical validity of these identities can be easy show by using the formulas given in [NIE75].

In all of the experiments, the statistical significances obtained are 0.001, 0.00000 for r_{si} and r_i $(1 \leq i \leq 3)$, respectively. Such low values for the statistical significances indicate the validity of the experimental results.

Now, let us consider the consistency aspects. In Figure 4 it is easy to see that the consistency between C^2DVM and ACM is always better than the consistency between ESM and ACM. This is because $b_1 > b_3$ and $a_1 < a_3$ hold for all of the sixteen D matrices used. Notice that consistency between C^2DVM and ACM is remarkably high, because b_1 values are very high the smallest b_1 occurrence being 0.803. Similarly, the consistency between C^2DVM and ACM is better than

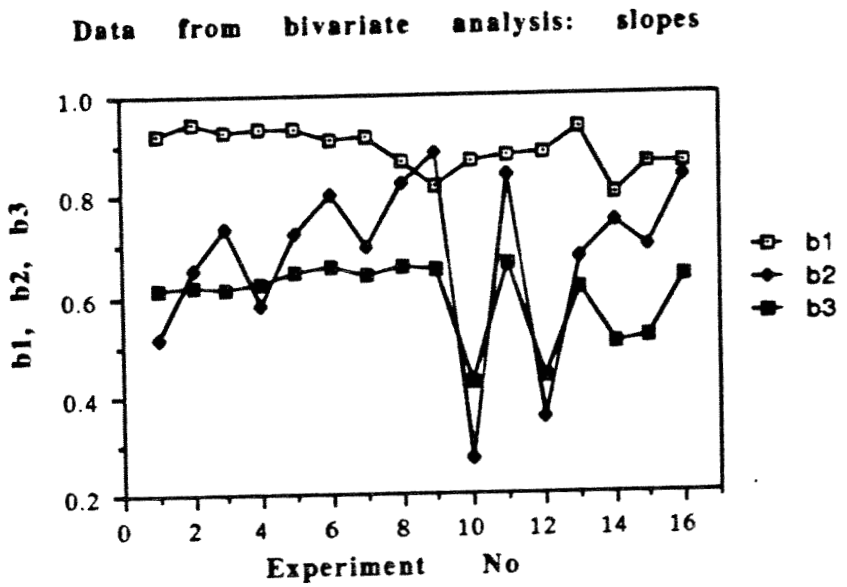

Figure 4.a Results of the bivariate analysis tests with slopes: b_1, b_2, b_3

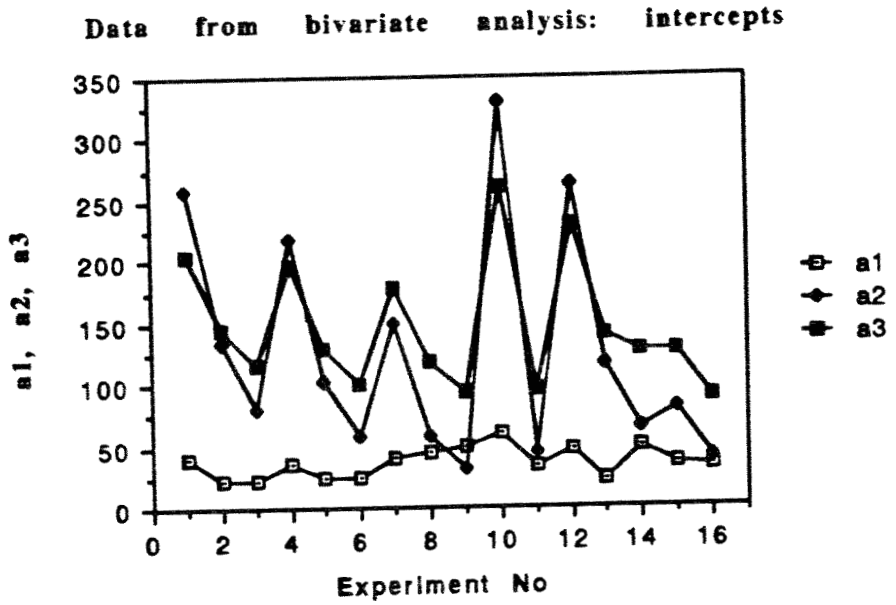

Figure 4.b Results of the bivariate analysis tests with intercepts: a_1, a_2, a_3

Figure 4. Results of the bivariate analysis experiments
($R_{C^2DVM} = b_1 \times R_{ACM} + a_1$, $R_{C^2DVM} = b_1 \times R_{ESM} + a_1$, $R_{ACM} = b_1 \times R_{ESM} + a_1$)

Table 3. Correspondence between experiments and D matrices for Figure 4

Exp. No	1	2	3	4	5	6	7	8
D matrix	D_{2-40}	D_{3-40}	D_{4-40}	D_{2-30}	D_{3-30}	D_{4-30}	D_{2-20}	D_{3-20}
Exp. No	9	10	11	12	13	14	15	16
D matrix	D_{4-20}	D'_{2-40}	D'_{4-40}	D'_{2-30}	D'_{3-30}	D'_{2-20}	D'_{3-20}	D'_{4-20}

the consistency between C^2DVM and ESM (i.e., when compared b_1 and b_2 in fifteen of the sixteen observations b_1 comes out greater than b_2 the only exception being the case of $D_{2\text{-}40}$).

As we stated previously, ACM had been introduced to reduce the computational requirements of ESM and as an approximation to ESM. However, our experiments show that C^2DVM, which is computationally the most effective among the three, is a better approximation to ESM as observed in the twelve observations with $b_2 > b_3$ and $a_2 < a_3$ out of the total sixteen. To confirm this the Wilcoxon matched-pairs signed-ranks test [DANI78] has been performed by using the sixteen b_2 and b_3 (the slopes of the regression lines) values of Figure 4. The one sided test has indicated that C^2DVM-ESM is more consistent than ACM-ESM with the critical value (P) of less than 0.022.

4.2 Tunable Indexing

The experiments on tunable indexing have been performed using the nine D matrices defined in Table 2. The experiments are performed in the following two ways (for the complete algorithm refer to Section 3.2):

For $N_c < n_c$: In step (a) of the tunable indexing algorithm terms are sorted in ascending order according to TDVs. For this case step (b) of the algorithm yields N_{min}. In step (c) of the algorithm we expect a gradual increase in n_{ch} that finally reaches $n_{ch} = n_c$ when h= n, n_c being the number of clusters of the original D matrix.

For $N_c > n_c$: Terms are sorted in descending order according to TDVs. This time step (b) of the tunable indexing algorithm will yield N_{max}. In step (c) of the algorithm we expect a gradual decrease in the number of clusters.

As can be seen, the number of clusters N_{min} and N_{max} can be observed with different indexing vocabularies for the same initial D matrix. The statistics of the D matrices that would result in N_{min} and N_{max} number of clusters are given in Tables 4.A and 4.B, respectively.

Tables 4.A and 4.B show that for $D_{2\text{-}40}$ we obtain N_{min}= 15 and N_{max}= 80 by using 211 and 432 terms. In order to obtain N_{min} we used the terms with lowest TDVs , i.e., the terms with high term generality that appear in large number of documents. Therefore, the D matrix has fewer number of terms. For example, for $D_{2\text{-}40}$ and $D_{3\text{-}40}$ the D matrix can be defined with 211 and 201 terms, respectively, which correspond to the cardinality of the indexing vocabulary T. For these matrices, the term generalities (t_g) are also rather high. For example, for $D_{2\text{-}40}$ and $D_{3\text{-}40}$ they are 18.32 and 18.62 respectively. Obviously this will satisfy the recall sensitive users because the documents are more similar to each other. And this generates larger clusters increasing recall in a cluster-based retrieval environment [SALT78, CAN87b].

Table 4. Statistics of D matrices generated with tunable indexing

A) For minimum possible number of clusters (N_{min})

D matrix	n	t	x_d	t_g	x_{dw}	t_{gw}	N_{min}
$D_{2\text{-}40}$	211	3865	18.06	18.32	29.33	29.75	15
$D_{3\text{-}40}$	201	3742	17.49	18.62	28.19	30.01	14
$D_{4\text{-}40}$	177	3402	15.90	19.66	25.59	31.65	13
$D_{2\text{-}30}$	212	3511	16.41	16.56	26.37	26.62	16
$D_{3\text{-}30}$	202	3404	15.91	16.85	25.24	26.74	16
$D_{4\text{-}30}$	178	3120	14.58	17.53	23.27	27.98	15
$D_{2\text{-}20}$	203	2574	12.03	12.68	17.96	18.94	20
$D_{3\text{-}20}$	194	2510	11.73	12.94	17.35	19.13	20
$D_{4\text{-}20}$	172	2296	10.73	13.35	15.87	19.75	19

B) For maximum possible number of clusters (N_{max})

D matrix	n	t	x_d	t_g	x_{dw}	t_{gw}	N_{max}
$D_{2\text{-}40}$	432	1275	5.96	2.75	10.27	5.09	80
$D_{3\text{-}40}$	241	1058	4.96	4.39	9.17	8.15	57
$D_{4\text{-}40}$	182	1055	4.93	5.80	9.14	10.75	45
$D_{2\text{-}30}$	403	1164	5.44	2.89	9.48	5.03	82
$D_{3\text{-}30}$	258	1096	5.12	4.25	9.14	7.59	58
$D_{4\text{-}30}$	187	1051	4.91	5.62	9.87	10.16	45
$D_{2\text{-}20}$	342	929	4.34	2.72	7.57	4.73	87
$D_{3\text{-}20}$	343	1416	6.62	4.13	10.49	6.54	58
$D_{4\text{-}20}$	183	982	4.59	5.37	8.09	9.46	47

C) For $N_c = m/\log_2 m = 27.64$

D matrix	n	t	x_d	t_g	x_{dw}	t_{gw}	N_c
$D'_{2\text{-}40}$	906	6953	32.49	7.67	47.64	11.25	27.66
$D_{3\text{-}40}$	757	6840	31.96	9.04	49.15	13.90	27.46
$D'_{4\text{-}40}$	570	5355	25.02	9.39	37.18	13.96	27.58
$D'_{2\text{-}30}$	814	6223	29.08	7.64	42.25	11.11	27.66
$D'_{3\text{-}30}$	720	6223	29.08	8.64	43.71	12.99	27.69
$D_{4\text{-}30}$	570	5355	25.02	9.39	37.18	13.96	27.58
$D'_{2\text{-}20}$	506	4218	19.71	8.34	27.89	11.79	27.66
$D'_{3\text{-}20}$	522	4238	19.80	8.12	27.35	11.21	27.64
$D'_{4\text{-}20}$	489	4226	19.75	8.64	27.97	12.24	27.71

The opposite case is valid for N_{max}. In other words, to observe N_{max} the terms with high TDVs are used and these terms have low term generality. Hence, more terms are needed to obtain a D matrix that yields N_{max}. However, comparing $D_{2\text{-}40}$ and $D_{3\text{-}40}$ (see Table 4.B), we can see that D matrices are defined by 432 and 241 terms, respectively. We see that 241 << 432 since their corresponding minimum t_gs are 2 and 3 respectively. This means that whenever T is extended, a newly added term will appear in at least two and three documents for the respective D matrices. This is because to reach N_c we need smaller T set with $D_{3\text{-}40}$ than with $D_{2\text{-}40}$.

In the tunable indexing experiments n_{ch} has varied from N_{min} (N_{max}) to n_c as expected with respect to Figure 2. The change in n_{ch} with respect to h (where h is the number of terms used in the description of D matrix) is plotted in Figure 5. Because the results for all nine matrices of Table 2 are similar we will display only the results of matrices $D_{2\text{-}30}$, $D_{3\text{-}30}$, and $D_{4\text{-}30}$. These plots show that the experimental behavior of the algorithm agrees with the theoretical expectation. In other words, if we begin $n_{ch} = N_{min}$ the inclusion of more terms in T increases the number of clusters, i.e., in general $n_{ci} < n_{cj}$ for $i < j$ On the other hand when we start with $n_{ch} = N_{max}$ the inclusion of more terms decreases the number of clusters and in general $n_{ci} > n_{cj}$ for $i < j$.

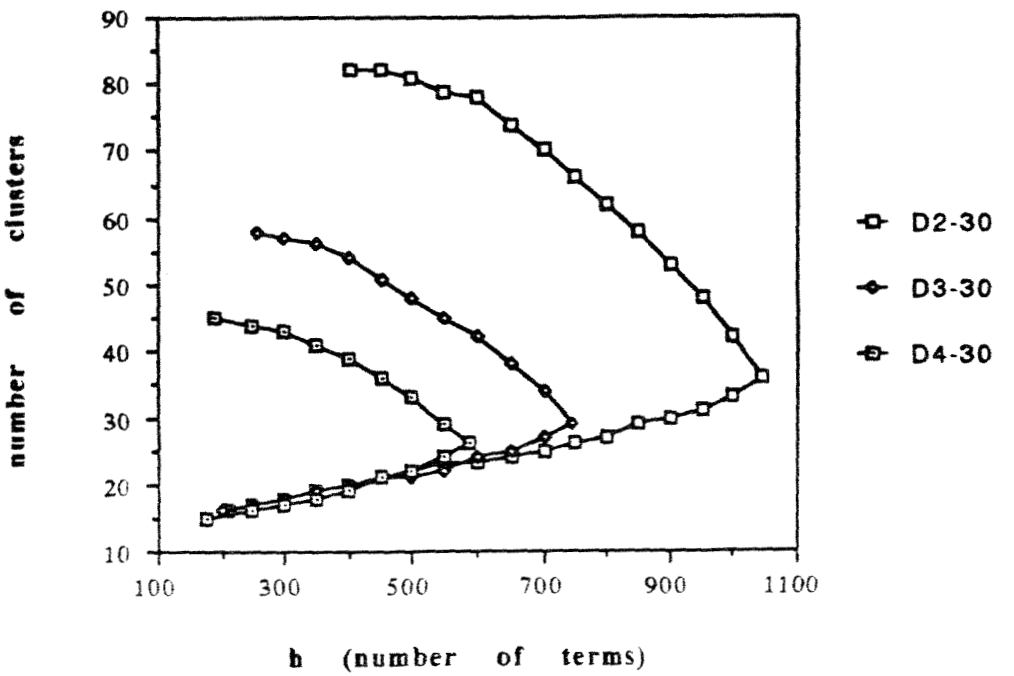

Figure 5. n_{ch} vs h for $D_{2\text{-}30}$, $D_{3\text{-}30}$, and $D_{4\text{-}30}$

Let the number of clusters desired by the system administrator be $N_c = m/\log_2 m$. Then for m= 214 (our database size), N_c= 27.64. (In the complexity

analysis of clustering algorithms, it has been assumed [SALT75c] that for a database of size m, the average cluster size would be $\log_2 m$.) N_c is reached either when going up from N_{min} to n_c or when going down from N_{max} to n_c. The characteristics of the D matrices generated for $N_c = m/\log_2 m$ (27.64) are given in Table 4.C. In the N_c column, Table 4.C shows that N_c is closest to 27.64 which is obtained from the corresponding matrix. The table shows that with different initial D matrices we obtain different resultant D matrices that yield the desired number, N_c, of clusters. Notice that $D_{3\text{-}40}$ of Table 4.C is identical to $D_{3\text{-}40}$ of Table 2, i.e., the original D matrix gives the desired number of clusters. Similarly, $D_{4\text{-}40}$ and $D_{4\text{-}30}$ of Table 4.C are identical.

5. CONCLUSION

It is shown that the CC based TDV calculation approach yields results that are compatible with the two other approaches available in the literature. In the experiments described, sixteen document description matrices are generated from a document database consisting of the titles, abstracts, and keywords of the 214 ACM TODS publications.

In the various tunable indexing experiments it is shown that CC based TDVs can be used to obtain an indexing vocabulary that yields the desired number of clusters. The tunable indexing concept can be used (a) to achive a high discrimination value model to enhance precision, or a low discrimination value model to increase recall, (b) to control the computational complexity of the CC based clustering methodology, and (c) to obtain clusters whose sizes are dictated by external (such as paging and memory) constraints. Also, in hierarchical clustering we may want to control the number of clusters at each level of the hierarchy. Tunable indexing can be used in achieving this.

Currently, we are evaluating the effect of tunable indexing on retrieval effectiveness.

REFERENCES

[BOOK86] Bookstein, A. "Performance of Self-Taught Documents: Exploiting Co-Relevance Structure in a Document Collection." In *Proceedings of the 9th Annual International ACM-SIGIR Conference* (September 1986), ACM, New York, 244-248.

[BORK77] Borko, H. "Toward a Theory of Indexing." *Information Processing and Management*. 13 (1977), 355-365.

[CAN83] Can, F., Ozkarahan, E. A. "A Clustering Scheme." In *Proceedings of the 6th Annual International ACM-SIGIR Conference* (1983), ACM, New York, 115-121.

[CAN84] Can F., Ozkarahan, E. A. "Two Partitioning Type Clustering Algorithms." *Journal of the American Society for Information Science*. 35, 5, (September 1984), 268-276.

[CAN85a] Can, F. *A New Clustering Scheme for Information Retrieval Systems Incorporating the Support of a Database Machine*. Ph. D. Dissertation. Dept. of Computer Engineering, Middle East Technical University, Ankara, 1985.

[CAN85b] Can, F., Ozkarahan, E. A. "Concepts of the Cover Coefficient Based Clustering Methodology." In *Proceedings of the 8th Annual International ACM-SIGIR Conference* (June 1985), ACM, New York, 204-211.

[CAN87a] 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*. 38, 3 (May 1987), 171-183.
[CAN87b] Can, F., Ozkarahan, E. A. "Concepts and Effectiveness of the Cover Coefficient Based Clustering Methodology for Text Databases." Submitted to *ACM Transactions on Database Systems*.
[CAN87c] Can, F., Ozkarahan, E. A. "A Dynamic Cluster Maintenance System for Information Retrieval." In *Proceedings of the 10th Annual International ACM-SIGIR Conference* (June 1987), ACM, New York, 123-131.
[CAN88] Can, F., Ozkarahan, E. A. "Dynamic Cluster Maintenance." *Information Processing and Management* (to appear).
[CRAW75] Crawford, R. G. "The Computation of Discrimination Values." *Information Processing Management*. 11 (1975), 249-253.
[COOP78] Cooper, W. S., Maron, M. E. "Foundations of Probabilistic and Utility-Theoretic Indexing." *Journal of the Association for Computing Machinery*. 25, 1 (January 1978), 67-80.
[DANI78] Daniel, W. W. *Applied Nonparametric Statistics*. Houghton Mifflin Co., Boston, 1978.
[GORD85] Gordon, M. D. "A Learning Algorithm Applied to Document Redescription." In *Proceedings of the 9th Annual International ACM-SIGIR Conference* (June 1985), ACM, New York, 179-186.
[KOLO83] Kolodner, J. L. "Indexing and Retrieval Strategies for Natural Language Fact Retrieval." *ACM Transctions on Database Systems*. 8, 3 (September 1983), 434-464.
[LANC75] Lancaster, F. W. *Information Retrieval Systems: Characteristics, Testing and Evaluation, 2nd ed.* Wiley, New York, 1975.
[NIE75] Nie, N. H., et. al. *SPSS Statistical Package for the Social Sciences, 2nd ed.* McGraw-Hill, New York, 1975.
[OZKA84] Ozkarahan, E. A., Can, F. "An Integrated Fact/Document Information System for Office Automation." *Information Technology: Research and Development*. 3, 3 (1984), 142-156.
[OZKA86a] Ozkarahan, E. *Database Machines and Database Management*. Prentice Hall, Englewood Cliffs NJ, 1986.
[OZKA86b] Ozkarahan, E. A., Can, F. "An Automatic and Tunable Indexing System." In *Proceedings of the 9th Annual International ACM-SIGIR Conference* (September 1986), ACM, New York, 234-243.
[SALT75a] Salton, G. "A Theory of Indexing." Regional Conference Series in Applied Mathematics, No 18, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1975.
[SALT75b] Salton, G., Wong, A., Yang, C. S. "A Vector Space Model for Automatic Indexing." *Communications of the ACM*. 18, 11 (November 1975) 613-620.
[SALT75c] Salton, G. *Dynamic Information and Library Processing*. Prentice Hall, Englewood Cliffs NJ, 1975.
[SALT78] Salton, G., Wong, A. "Generation and Search of Clustered Files." *ACM Transactions on Database Systems*. 3, 4 (December 1978), 321-346.
[SALT81] Salton, G., Wu, H., Yu, C. T. "The Measurement of Term Importance in Automatic Indexing." *Journal of the American Society for Information Science*. 32, 3 (May 1981) 175-186.
[SALT83] Salton, G., McGill, M. J. *Introduction to Modern Information Retrieval*. McGraw Hill, New York, 1983.
[SALT86] 37. Salton, G. "Another Look at Automatic Text-Retrieval Systems." *Communications of the ACM*. 29, 7 (July 1986). 648-656.
(1973), 499-513.

[URAL86] 42. Ural, M. H. *Performance Evaluation of the Cover Coefficient Based Clustering and Cluster Maintenance Methodology in Information Retrieval.* MSc Thesis, Dept. of Electrical and Electronics Engineering, Middle East Technical University, Ankara, 1986.

[VANR79] Van Rijsbergen, C. J. *Information Retrieval, 2nd ed.* Butterworths, London, 1979.

[WILL85] Willett, P. "An Algorithm for the Calculation of Exact Term Discrimination Values." *Information Processing and Management.* 21, 3(1985), 225-232.

[WONG87] Wong, S. K. M., Ziarko, W., Raghavan, V. V., Wong, P. C. N. "On Modelling of Information Retrieval Concepts in Vector Spaces." *ACM Transactions on Database Systems.* 12, 2 (June 1987) 299-321..

[YU76] Yu, C. T., Salton, G. "Precision Weighting - An Effective Automatic Indexing Method." *Journal of the ACM.* 23, 1 (January 1976), 76-88.

APPENDIX

The Cover Coefficient Concept

The cover coefficient (CC) concept was originally introduced for document clustering purposes [CAN83]. In this concept the document description matrix (refer to Section 2 for its definition) is mapped onto an m by m C (cover coefficient) matrix. Each entry, c_{ij}, of the C matrix is the result of a double-stage probabilistic experiment [CAN87b] and indicates the extent with which document$_i$ (d_i) is covered by d_j.

A C matrix is formed by using the matrices S and S' and the relationship $C = S \times S'^{T}$, where S'^{T} is the transpose of the matrix S'. The entries of the S and S' matrices are defined as follows:

$$s_{ij} = d_{ij} \times \left[\sum_{k=1}^{n} d_{ik}\right]^{-1}, \quad s'_{ij} = d_{ij} \times \left[\sum_{k=1}^{m} d_{kj}\right]^{-1} \qquad 1 \le i \le m,\ 1 \le j \le n$$

s_{ij} and s'_{ij} indicate, respectively, the significance of t_j for d_i or probability of selecting t_j from d_i, and significance of d_i for t_j or probability of selecting d_i from t_j. The entries of the C matrix are defined as follows:

$$c_{ij} = \sum_{k=1}^{n} s_{ik} \times s'_{kj} = \sum_{k=1}^{n} (\text{probability of selecting } t_k \text{ from } d_i) \times (\text{probability of selecting } d_j \text{ from } t_k)$$

From the definition of the S and S' matrices

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

where α_i and β_k are the reciprocals of row$_i$ and column$_k$ sums, respectively. Each entry of C is a covering coefficient among documents and $(c_{i1} + c_{i2} + \ldots + c_{in}) = 1$ for $1 \le i \le m$.

The diagonal entries c_{ii} of the C matrix indicate the extent with which d_i is covered by itself and called the uniqueness or decoupling coefficient, δ_i, of d_i.

$c_{ii} > c_{ij}$ if $i \neq j$ for a binary D matrix; however, this condition does not hold for a weighted D matrix.

If none of the terms of d_i is used by any other document then $\delta_i = 1$, i.e., d_i is unique or completely "decoupled" from the other documents of the document database; otherwise, $\delta_i < 1$ meaning that d_i is coupled with one or more documents of the database. Thus the values of δ_i fall in the range $0 < \delta_i \leq 1$.

The sum of the off-diagonal entries of row_i is referred to as the coupling coefficient ψ_i of d_i. ψ_i indicates the extent of coupling of d_i with the other documents in the database. $\psi_i = 1 - \delta_i$ and from the definition of δ_i, the value range of ψ_i is $0 \leq \psi_i < 1$.

The overall or average decoupling coefficient of the database, δ, is the average of δ_i values where $1 \leq i \leq m$. It is hypothesized that the number of clusters within a database, n_c, can be obtained as

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

It is shown that the value range for n_c indicated by δ is $1 \leq n_c \leq \min(m, n)$. The average number of documents within a cluster would be $m/n_c = m/(\delta \times m) = 1/\delta$. The concept of decoupling coefficient indicates that the increase in δ or individual δ_i ($1 \leq i \leq m$) will increase the number of clusters and hence decrease their average size. (Remember that the number of clusters is nothing but the summation of individual decoupling coefficients.) When generating clusters n_c number of documents are selected as cluster seeds and non-seed documents are assigned to seeds which cover them maximally.

The use of CC related concepts for clustering and cluster maintenance can be found in various publications [CAN87c, CAN88, OZKA86a].