

# Computation of Term/Document Discrimination Values by Use of the Cover Coefficient Concept

Fazlı Can\*

Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey

Esen A. Ozkarahan

Department of Computer Science, Arizona State University, Tempe, AZ 85287

Indexing in information retrieval (IR) is used to obtain a suitable vocabulary of index terms and optimum assignment of these terms to documents for increasing the effectiveness and efficiency of an IR system. The concept of term discrimination value (TDV) is one of the criteria used for index-term selection. In this article a new concept called the cover coefficient (CC) will be used in computing TDVs. After a brief introduction to the theory of indexing and the CC concept, an efficient way of computing TDVs by use of the CC concept, index-term selection, and weight modification are discussed. It is also shown that the computational cost of the CC approach in the calculation of TDVs is favorably comparable to the cost of a different approach that uses similarity coefficients. Furthermore, the TDVs obtained by the CC approach are consistent with those of the latter approach.

## I. Introduction

Indexing can be described as means of describing documents by a set of terms whose resolving power (i.e., discrimination of documents from each other) is as high as possible. For very large document databases it is essential to employ indexing. Indexing is used both in inverted-file- and cluster-based IR systems. In cluster-based retrieval systems, clustering (i.e., the process of putting similar documents into the same group) is used to increase the effectiveness of the system [1]. The partitioning (clustering) of the document space would enable the retrieval systems, including the full-text retrieval systems, to operate efficiently. However, the task of indexing cannot be done arbitrarily, because it affects the performance of an IR system dramatically. For example, a study which has been made on the MEDLARS system by Lancaster shows that about 50% of system failures in terms of recall and precision are related to indexing [2, p. 138;

ref. 27], where recall (precision) is the number of retrieved relevant documents divided by the total number of relevant documents in the collection (by the total number of retrieved documents).

Having an indexing system we can then (a) define the contents of documents, (b) determine the topics of a document collection, and (c) find the relatedness of user request (query) and individual documents of a collection.

Although indexing can be performed manually as well as automatically by extracting index terms from documents, we will concentrate only on automatic indexing. Accordingly, we will abstract the task of automatic indexing by a 5-tuple  $I = [M, C, T, W, D]$ , where

- M*: the indexing methodology (i.e., the decision criteria used in the selection of index terms and in the assignment of index terms to documents).
- C*: the document collection containing the documents to be indexed,  $C = \{d_i | 1 \leq i \leq m\}$ .
- T*: the set of index terms  $T = \{t_i | 1 \leq i \leq n\}$  (i.e., indexing vocabulary for the description of these  $m$  documents).
- W*: the possible weights to be assumed by index terms in documents (i.e., the relative importance of terms in individual documents). If  $W = \{0, 1\}$  then indexing is binary. If  $W = R$  (where  $R$  is the set of non-negative real numbers), then indexing is weighted.
- D*: the abstract description of documents as a matrix of size  $m \times n$ , as determined by the cardinalities of the sets  $C$  and  $T$ , that is,  $|C| = m$ ,  $|T| = n$ . An individual entry of  $D$ ,  $d_{ij}$  ( $1 \leq i \leq m$ ,  $1 \leq j \leq n$ ), indicates the relative importance of  $t_j$  in  $d_i$ , where  $d_{ij} \in W$ .

In indexing, there are two basic measures that are especially important. These are depth of indexing (indexing exhaustivity) and term generality (term breadth) [1, 3]. For  $W = \{0, 1\}$ , i.e., binary indexing, depth of indexing can be measured by the number of terms used for the description of

\*Present address: Miami University; Oxford, OH 45056.

Received June 20, 1985; revised October 2, 1985; accepted May 9, 1986.

© 1987 by John Wiley & Sons, Inc.

individual documents. Accordingly, the depth of indexing for  $d_i, x_{d_i}$ , is equal to  $\sum_{i=1}^n d_i$ . Similarly, using binary indexing, term generality can be measured by the number of documents described by individual terms. Hence, generality of term  $t_j, t_{g_j}$  is equal to  $\sum_{j=1}^m d_j$ . (Similar definitions can be made for weighted indexing.) The average depth of indexing ( $x_d$ ) and term generality ( $t_g$ ) can then be defined as  $\sum_{i=1}^m x_{d_i}/m$  and  $\sum_{i=1}^n t_{g_i}/n$ , respectively [4]. It has been shown that depth of indexing (term generality) affects the recall (precision) of an IR system. Appropriate levels of indexing exhaustivity and term generality would lead to better recall and precision values [1, 3, 5]. Accordingly, the following are some important issues concerning an indexing task ( $I$ ):

- (a) What is the best  $T$  for a particular  $C$ ?
- (b) What should be the importance of  $t_j$  in  $d_i$  (i.e., the value of  $d_{ij}$ )?
- (c) What is the appropriate level of  $x_{d_i}$  ( $1 \leq i \leq m$ ) and  $x_d$ ?
- (d) What is the appropriate level of  $t_{g_i}$  ( $1 \leq i \leq n$ ) and  $t_g$ ?

Research on indexing has resulted in some interesting theories. For a brief overview of the indexing theories of Jonker, Heilprin, Landry, and Salton and his co-workers the reader may refer to [5]. Detailed illustrations of the theories of Salton and co-workers can be found in [6, 7]. A probabilistic approach for better indexing, the term precision measurement, is formally treated in [8]. IR experiments on term precision are described in [9, 10]. Another probabilistic approach called the term utility measurement is discussed in [10, 11]. The term relevance measure (work of Robertson, Sparck Jones, Van Rijsbergen, Harper, etc.) [1] is yet another probabilistic approach to estimate the weight of index terms. The probabilistic approaches, such as term precision and term utility measure, may have some practical difficulties, since they require the occurrence characteristics of query terms in the relevant and nonrelevant documents; hence they are more suitable to query reformulation. However, there are some studies to overcome this obstacle [12].

In the remainder of this article, the concepts of term discrimination value (TDV) and cover coefficient (CC) are briefly introduced. An efficient way to compute TDVs by use of the CC concept is presented in detail. This is followed by a section on experiments performed which show that the TDVs computed according to the CC concept are consistent with those computed by use of the cosine similarity measure. Furthermore, the computational cost of the CC concept-based approach compares favorably with that of the similarity-based approach. The use of the CC concept in connection with the TDV concept for index-term selection and weight modification of index terms is also discussed.

## II. The Term-Discrimination Value Concept

An indexing theory proposed by Salton and his co-workers uses the TDV concept [6, 10]. In this theory, an optimum indexing vocabulary is obtained according to sig-

nificance of terms. For computing term significance several weighting measures have been proposed, such as term frequency, signal-to-noise ratio, variance, dynamic information value, and discrimination value [3, 6].

In a document space, TDV is used to measure an index term's contribution to how well documents are separated (distinguished) from each other. The separation of the documents of a collection can be measured by looking at the document space density (i.e., average similarity of documents)  $Q'$  [3, 6].

$$Q' = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^m S(d_i, d_j) \quad (0 \leq Q' \leq 1 \text{ for } 0 \leq S \leq 1),$$

where  $S(d_i, d_j)$  is the similarity between  $d_i$  and  $d_j$ , which can be calculated according to the cosine coefficient [3] as

$$S(d_i, d_j) = \frac{\sum_{k=1}^n d_{ik} \times d_{jk}}{[\sum_{k=1}^n (d_{ik})^2 \times \sum_{k=1}^n (d_{jk})^2]^{1/2}}.$$

The document space density calculation according to the above formula involves a very large number of computations. The computational cost can be lowered by first creating a centroid  $G$  for the document collection. Each centroid entry  $g_j$  ( $j = 1, \dots, n$ ) of  $G$  is then defined as the average weight of  $t_j$  in all of the  $m$  documents:  $g_j = (1/m) \sum_{i=1}^m d_{ij}$ . Then the (approximate) document space density can be defined as follows [3, 6]:

$$Q = \frac{1}{m} \sum_{i=1}^m S(d_i, G) \quad (0 < Q \leq 1 \text{ for } 0 < S \leq 1).$$

Accordingly, document collections with greater (lesser) separation of document description vectors will have lower (higher)  $Q$  value. When all documents are distinct from each other (i.e., no two documents have a common term), then  $Q'$  will be 0 and  $Q$  will have a value close to 0. If all documents are identical, then both  $Q'$  and  $Q$  will be equal to 1. The appropriate selection of index terms and assignment of them to individual documents [i.e., appropriate  $x_{d_i}$  ( $1 \leq i \leq m$ ) and  $t_{g_j}$  ( $1 \leq j \leq n$  values)] will yield an optimum value for the document space density of a collection. Accordingly, better discrimination of documents will increase the effectiveness of an IR system [10].

Deletion of a term from  $T$  will change the indexing vocabulary and hence the description of documents. Since  $Q$  is a function of document descriptions, such a change will also change the document space density  $Q$  of a collection. The deletion of  $t_j$  ( $1 \leq j \leq n$ ) will set  $d_{ij}$  ( $1 \leq i \leq m$ ) and hence  $g_j$  to null. The new value of  $Q, Q_j$  will be the following:

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

where

$$d_i^j = (d_{i1}, d_{i2}, \dots, d_{i,j-1}, d_{i,j+1}, \dots, d_{in}),$$

$$G_j = (g_1, g_2, \dots, g_{j-1}, g_{j+1}, \dots, g_n).$$

The difference  $Q_j - Q$  reflects the change due to deletion of term  $t_j$ . If the assignment of  $t_j$  separates the documents more from each other, then its assignment will decrease the document space density. Consequently, the removal of  $t_j$  makes the documents closer to each other, thereby increasing the document space density and leading to the situation  $Q_j > Q$ . Accordingly,  $Q_j - Q$  will be greater than zero. The difference  $Q_j - Q$  is defined as the discrimination value of  $t_j$ ,  $TDV_j$  [3, 6].

TDV has the following properties:

- (a)  $TDV_j > 0$  for a good discriminator  $t_j$  (i.e., a term which makes the documents more distinguishable from each other),
- (b)  $TDV_j \approx 0$  for an indifferent term  $t_j$  (i.e., the assignment of  $t_j$  to documents does not affect the separation of documents), and
- (c)  $TDV_j < 0$  for a poor discriminator  $t_j$  (i.e., a term which makes the documents less distinguishable from each other).

After determining the TDVs, entries in the  $D$  matrix (i.e., the matrix used for the calculation of TDVs) are modified as follows [3, 6]:

$$d'_{ij} = TDV_j \times d_{ij}.$$

Such a modification in the matrix will increase the effectiveness of an IR system in terms of recall and precision [6, 7]. TDV can also be used to optimize the indexing vocabulary ( $T$ ) of a collection ( $C$ ) to improve the effectiveness of an IR system [6].

### III. The Cover Coefficient Concept

The cover coefficient (CC) concept was originally introduced for document clustering purposes [13–17]. The document collection is defined by the  $D$  matrix which is already described. The entries of the  $D$  matrix,  $d_{ij}$  ( $1 \leq i \leq m$ ,  $1 \leq j \leq n$ ) satisfy the following conditions:

$$(a) \quad \sum_{j=1}^n d_{ij} > 0, \quad 1 \leq i \leq m$$

(each document has at least one term), and

$$(b) \quad \sum_{i=1}^m d_{ij} > 0, \quad 1 \leq j \leq n$$

(each term is assigned to at least one document).

The  $D$  matrix is mapped onto an  $m \times m$   $C$  (cover coefficient) matrix. Each  $c_{ij}$  entry of the  $C$  matrix indicates the extent with which document<sub>*i*</sub> ( $d_i$ ) is covered by document<sub>*j*</sub> ( $d_j$ ).

The  $C$  matrix is formed by using the matrices  $S$  and  $S'$  as  $C = S \times S'^T$ . Entries of the  $S$  and  $S'$  matrices are defined as follows:

$$s_{ij} = d_{ij} / \sum_{k=1}^n d_{ik}, \quad s'_{ij} = d_{ij} / \sum_{k=1}^m d_{kj},$$

where  $1 \leq i \leq m$ ,  $1 \leq j \leq n$ .

$s_{ij}$  ( $s'_{ij}$ ) indicates the significance of  $t_j(d_i)$  for  $d_i(t_j)$ . The entries of the  $C$  matrix are defined as follows:

$$\begin{aligned} c_{ij} &= \sum_{k=1}^n s_{ik} \times s'_{kj} \\ &= \sum_{k=1}^n (\text{significance of } t_k \text{ in } d_i) \\ &\quad \times (\text{significance of } d_j \text{ for } t_k), \end{aligned}$$

where  $S'^T$  indicates the transpose of the matrix  $S'$ .

From the definition of  $S$  and  $S'$  matrices,

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

where  $\alpha_i$  and  $\beta_k$  are the reciprocal of the row<sub>*i*</sub> and column<sub>*k*</sub> sums as shown below:

$$\alpha_i = \frac{1}{\sum_{j=1}^m d_{ij}}, \quad 1 \leq i \leq m,$$

$$\beta_k = \frac{1}{\sum_{j=1}^m d_{jk}}, \quad 1 \leq k \leq n.$$

Each entry of  $C$  is a covering coefficient among documents and  $\sum_{j=1}^m c_{ij} = 1$  for  $1 \leq i \leq m$ . Diagonal entries  $c_{ii}$  ( $1 \leq i \leq m$ ) of the  $C$  matrix indicate the extent with which  $d_i$  is covered by itself and called the uniqueness or decoupling coefficient  $\delta_i$  of  $d_i$ .  $c_{ii} \geq c_{ij}$  if  $i \neq j$  for a binary  $D$  matrix; however, this condition does not necessarily hold for a weighted  $D$  matrix [13, 14, 16]. This is very simple to prove. Consider  $c_{ii}$  and  $c_{ij}$ :

$$c_{ii} = \alpha_i \times \sum_{k=1}^n d_{ik}^2 \times \beta_k,$$

$$c_{ij} = \alpha_i \times \sum_{k=1}^n d_{ik} \times d_{jk} \times \beta_k.$$

If  $d_{jk} \geq d_{ik}$  and if  $d_{jk} > d_{ik}$  for at least one  $k$  ( $1 \leq k \leq n$ ), then  $c_{ij} > c_{ii}$ .

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

The sum of the off-diagonal entries of row<sub>*i*</sub> is referred to as the coupling coefficient  $\psi_i$  of  $d_i$ .  $\psi_i$  indicates the extent of coupling of  $d_i$  with the other documents in the collection.  $\psi_i = 1 - \delta_i$  and from the definition of  $\delta_i$ , the value range of  $\psi_i$  is  $0 \leq \psi_i < 1$  [13, 14, 16]. The overall or average decoupling coefficient of the collection is

$$\delta = \sum_{i=1}^m \delta_i / m = \text{tr}C / m \quad (0 < \delta \leq 1).$$

It is hypothesized that the number of clusters  $n_c$  within a collection is  $\delta \times m = \text{tr}C$ . The value range implied for

$n_c$  by  $\delta$  is  $1 \leq n_c \leq \min(m, n)$  [16, 18]. The average number of documents within a cluster would be  $m/n_c = m/(\delta \times m) = 1/\delta$ . The concept of decoupling coefficient implies that increase in  $\delta$  or in an individual  $\delta_i$  ( $1 \leq i \leq m$ ) will increase the number of clusters ( $n_c = \delta \times m = \sum_{i=1}^m \delta_i$ ) and hence decrease the average size of clusters. The use of such cover-coefficient-related concepts for clustering purposes is illustrated in [13, 14, 16].

Estimation of the number of clusters within a collection is one of the unresolved problems of clustering [19]. The computation of  $n_c$  in the CC-based methodology can be considered an heuristic. Currently we are working to show the validity of  $n_c = \delta \times m$  in the CC-based clustering methodology [20].

A similar  $n \times n$  matrix called  $C'$ , which has similar properties as the  $C$  matrix, can be constructed for index terms.  $C'$  is defined as the product  $S'^T \times S$ . The individual entries of the  $C'$  matrix are defined as follows:

$$c'_{ij} = \sum_{k=1}^n s'_{ik} \times s_{kj} = \sum_{k=1}^m (\text{significance of } d_j \text{ for } t_k) \times (\text{significance of } t_k \text{ in } d_i).$$

From the definition of  $S$  and  $S'$  matrices,

$$c'_{ij} = \beta_i \sum_{k=1}^m d_{ki} \times \alpha_k \times d_{kj} \quad (1 \leq i \leq n, 1 \leq j \leq n).$$

The concepts of decoupling and coupling coefficients of  $t_j$ , respectively,  $\delta'_j$  and  $\psi'_j$ , are the counterparts of the same concepts defined for documents. Hence  $\delta'_j = c'_{ij}$ , and  $\psi'_j = 1 - \delta'_j$ . The concepts of overall decoupling and coupling coefficients are also defined for terms and represented by  $\delta'$  and  $\psi'$ , respectively.

The number of clusters implied by  $\delta'$  is  $n'_c = \delta' \times n$ . It is shown that  $n'_c = n_c$  [13, 16]. The  $C'$  matrix can be used for constructing clusters of index terms and more effective indexing vocabularies [18]. The diagonal entries of the  $C'$  matrix are used for the construction of centroid vectors for document clusters and in the calculation of seed powers,  $p_i$  ( $1 \leq i \leq m$ ) of individual documents. The cluster seed power is determined as

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

and

$$p_i = \delta_i \times \psi_i \times \sum_{j=1}^n (d_{ij} \times \delta'_j = \psi'_j)$$

in the case of binary and weighted  $D$  matrices, respectively. The  $C$  matrix and the documents which have the highest cluster seed power are then used in the implementation of the two partitioning-type clustering algorithms [13, 16].

The CC concept also reveals some relationships between indexing and clustering. For binary indexing these relationships are discussed in [16, 18]. The effects of the total number of term assignments,  $t = \sum_{i=1}^m \sum_{j=1}^n d_{ij}$ , the average

depth of indexing  $x_d$ , and the average term generality  $t_g$  on  $n_c$  are expressed as  $n_c = t/(x_d \times t_g)$ . The effects of collection size ( $m$ ), vocabulary size ( $n$ ), and  $t$  on  $n_c$  are expressed by  $n_c = (m \times n)/t$ .  $n_c$  can also be written as a function of  $(m, t_g)$  and  $(n, x_d)$  as  $n_c = m/t_g = n/x_d$ . That is, the average sizes ( $d_c$  and  $d'_c$ ) of a document cluster and term cluster, respectively, are equal to  $t_g$  and  $x_d$ .

It should be emphasized that the foregoing relationships between indexing and clustering were first observed experimentally [15], then shown analytically [16, 18]. In the experiments, certain conditions were set for a word (more correctly a stem) to be an index term [15, 16]. The said relationships can be verified by using the well-known IR collections or creating some artificial  $D$  matrices as described in [21].

We should also note that the relationships are derivable from each other and are intuitively correct. Consider the relationship  $n_c = n/x_d$ . It is known that an increase in  $x_d$  leads to high recall and low precision [22]. To obtain the same effect in the cluster-based retrieval, one must have larger clusters [22]. This is provided by  $n_c = n/x_d$ .

#### IV. Use of the CC Concept for Computing TDV

Document space density ( $Q$ ) is similar to the overall decoupling coefficient  $\delta$  of documents in the following sense. If document descriptions are more distinguishable (i.e., if  $Q$  is low), then this also means that the documents are more decoupled from each other (i.e.,  $\delta$  will be high). In the reverse situation (i.e., when the documents are less distinguishable)  $Q$  will be high and  $\delta$  will be low. After observing this fact, we may use the CC concept in computing TDVs.

As shown in the foregoing, the overall decoupling coefficient  $\delta$  of a collection is  $\sum_{i=1}^m \delta_i/m$ . If deletion of a term  $t_j$  does not lower the number of documents (i.e., if each document is defined by at least two terms), then we can use  $\sum_{i=1}^m \delta_i$  by ignoring the divisor  $m$ .  $\sum_{i=1}^m \delta_i$  is nothing but the number of clusters we assume should exist within the collection. In a similar way to the original cosine similarity approach for computing  $TDV_l$  ( $1 \leq l \leq n$ ), we can use the number of clusters  $n_c$  and  $n_{cl}$  before and after deleting  $t_l$ . That is, we delete  $t_l$  and observe the change in  $n_c$  to compute the discrimination value of  $t_l$ . It is expected that the deletion of terms that are good discriminators (i.e., significant terms) would decrease the number of clusters by decreasing the decoupling coefficient of documents, i.e.,  $n_{cl} < n_c$ . The decrease in the number of clusters indicates that the documents of the  $D$  matrix are more similar to each other. Conversely, the deletion of terms with less significance, or correspondingly poor discriminators, would increase the number of clusters, i.e.,  $n_{cl} > n_c$ . The terms which do not have any significance for document description (i.e., the nondiscriminators) would not change the number of clusters in the collection, i.e.,  $n_{cl} \approx n_c$ .

This shows that the concepts of document space density ( $Q$ ) and average decoupling of documents ( $\delta$ ) are inverse to

TABLE 1. Effects of type of index term ( $t_l$ ) on the values of  $Q$ ,  $\delta$ , or  $n_c$ .

Type of $t_l$	Quantity	
	$Q$ vs $Q_l$	$\delta$ vs $\delta_l$ (or $n_c$ vs $n_{cl}$ )
Good discriminator ( $TDV_l > 0$ )	$Q < Q_l$	$\delta > \delta_l$ ( $n_c > n_{cl}$ )
Indifferent discriminator ( $TDV_l \approx 0$ )	$Q \approx Q_l$	$\delta \approx \delta_l$ ( $n_c \approx n_{cl}$ )
Poor discriminator ( $TDV_l < 0$ )	$Q > Q_l$	$\delta < \delta_l$ ( $n_c < n_{cl}$ )

each other. Table 1 shows the interpretation of the related quantities with respect to TDV.

We would like to point out that the diagonal matrix elements of the  $C$  matrix  $c_{ii}$  ( $1 \leq i \leq m$ ), are not "related" to  $S(d_i, d_i)$ , since  $S(d_i, d_i) = 1$ . On the other hand,  $c_{ii} = 1$ , if and only if  $d_i$  is unique, i.e.,  $\sum_{k=1}^n d_{ik} \times d_{jk} = 0$  ( $1 \leq j \leq m, i \neq j$ ). We would obtain  $n_c = m$  (i.e., number of clusters equal to number of documents) if all  $d_i$  are unique (i.e.,  $\delta_i = 1$  and  $\delta = 1$ ).

TDV of  $t_l$  ( $1 \leq l \leq n$ ) is defined as the difference

$$TDV_l = n_c - n_{cl}.$$

By this formula, good, poor, and indifferent discriminators will, respectively, have a TDV of positive, negative, or approximately zero values. As we will see in B of Section VII, however, the  $TDV_l$  is expressed as the ratio  $n_c/n_{cl}$  in adjusting term weights in the  $D$  matrix.

To compute  $TDV_l$  we need to have  $n_c$  and  $n_{cl}$ . These are indicated by the decoupling coefficients as follows:

$$n_c = \sum_{i=1}^m \delta_i = \sum_{i=1}^m \alpha_i \times (d_{i1}^2 \times \beta_1 + d_{i2}^2 \times \beta_2 + \dots + d_{in}^2 \times \beta_n),$$

$$n_{cl} = \sum_{i=1}^m \delta_i^l = \sum_{i=1}^m \alpha_i^l \times (d_{i1}^2 \times \beta_1 + d_{i2}^2 \times \beta_2 + \dots + d_{i,l-1}^2 \times \beta_{l-1} + d_{i,l+1}^2 \times \beta_{l+1} + \dots + d_{in}^2 \times \beta_n).$$

In the formula of  $n_{cl}$ , the superscripts and subscripts of  $\delta_i^l$  and  $\alpha_i^l$  notate the absence of  $t_l$  in  $d_i$  ( $1 \leq i \leq m$ ). Also,  $\alpha_i^l$  can be easily defined as

$$\alpha_i^l = \left( \sum_{j=1}^n d_{ij} \right)^{-1} = (\alpha_i^{-1} - d_{il})^{-1}, \quad j \neq l.$$

As can be seen from  $n_c$  and  $n_{cl}$ , the two formulas differ only in the term belonging to  $t_l$ . Therefore,  $n_{cl}$  can also be expressed as

$$n_{cl} = \sum_{i=1}^m \alpha_i^l \times \left( \frac{\delta_i}{\alpha_i} - d_{il}^2 \times \beta_l \right).$$

In this formula, the term  $-d_{il}^2 \times \beta_l$  eliminates the contribution of  $t_l$  to  $n_c$  via its individual term generality ( $\beta_l = 1/t_{gl}$ ).  $\delta_i/\alpha_i$  eliminates the contribution of  $t_l$  to  $n_{cl}$  due to its effect on depth of indexing ( $\alpha_i = 1/x_{di}$ );  $\alpha_i^l$  reintroduces the effect of the modified depth of indexing. It is obvious that the above formula for  $n_{cl}$  is extremely ineffi-

cient. Notice that not all of the documents contain the deleted term  $t_l$ . After this observation we may use the value of  $n_c$  to calculate  $n_{cl}$ :

$$n_{cl} = n_c + \sum_{i=1}^{f_l} \left( \alpha_i^l \times \frac{\delta_i}{\alpha_i} - d_{il}^2 \times \beta_l \right) - \delta_i,$$

for all  $d_i \in D_l$ ,

where  $f_l = |D_l|$  and  $D_l = \{d_i | d_i \in D \wedge d_{il} \neq 0\}$ , i.e.,  $f_l$  is the document frequency of  $t_l$ . In this formula, first the contribution of a document to the number of clusters is deleted (by  $-\delta_i$ ) and then the contribution of the same document is reintroduced by disregarding the existence of  $t_l$ .

From the above formula, and from the definition of ( $TDV_l = n_c - n_{cl}$ ), the discrimination value of  $t_l$  can be expressed by the following formula:

$$TDV_l = \sum_{i=1}^{f_l} \left[ \delta_i - \alpha_i^l \times \left( \frac{\delta_i}{\alpha_i} - d_{il}^2 \times \beta_l \right) \right].$$

Obviously,  $TDV_l$  is nothing but the change in the number of clusters after the deletion of  $t_l$ .

For the sake of completeness and to aid programming, a simple algorithm for the computation of  $TDV_l$  ( $1 \leq l \leq n$ ) is given in the following:

```

for  $d_i$  ( $1 \leq i \leq m$ )
  compute (sum of row $_i$ ,  $\alpha_i$ )
endfor
for  $t_l$  ( $1 \leq l \leq n$ )
  compute  $\beta_l$ 
endfor
for  $d_i$  ( $1 \leq i \leq m$ )
  compute  $\delta_i$ 
endfor
for  $t_l$  ( $1 \leq l \leq n$ )
   $TDV_l = 0$ 
  for  $d_i$  ( $1 \leq i \leq m$ , and  $d_{il} > 0$ )
     $\alpha_i^l = (\text{sum of row}_i - d_{il})^{-1}$ 
     $TDV_l = TDV_l + [\delta_i - \alpha_i^l \times (\delta_i/\alpha_i - d_{il}^2 \times \beta_l)]$ 
  endfor
endfor

```

## V. The Computational Aspects of the CC Concept in the Calculation of TDVs

The computational aspects of the CC approach in determining TDVs will be illustrated in terms of both  $\delta_i$  and  $DV_l$ . For this purpose, the total number of operations (additions, subtractions, multiplications, and divisions) will be computed.

- (a) The computation of  $\delta_i$  ( $1 \leq i \leq m$ ):  $\alpha_i^{-1}$  ( $1 \leq i \leq m$ ) and  $\beta_j$  ( $1 \leq j \leq n$ ) require  $2 \times m \times n$  additions and  $n$  divisions for  $\beta_j$ . The number of additions,  $2 \times m \times n$ , can be reduced to  $2t$  by using an appropriate data structure such as an inverted file. The com-

putation for  $\delta_i$  ( $1 \leq i \leq m$ ) will require  $t$  additions and  $(m + 2t)$  multiplications ( $2t$  counts for  $d_{ij}^2 \times \beta_j$ ), and  $m$  divisions to obtain  $\alpha_i$  from  $\alpha_i^{-1}$  ( $1 \leq i \leq m$ ). The values maintained for each document, for step  $b$ , are  $\alpha_i^{-1}$  (row-sum $_i$ ) and  $\delta_i$ . For each term  $\beta_j$  is maintained.

- (b) The computation of  $DV_l$  ( $1 \leq l \leq n$ ): In this computation, all  $DV_l$  values ( $1 \leq l \leq n$ ) can be initialized as 0 and then  $DV_l$  values can be found by considering the documents one by one (i.e., the effect of a document on each  $DV_l$  for the terms used by the document). This means that  $\delta_i \times \alpha_i^{-1}$  will be computed only once. This involves  $m$  multiplications for  $m$  documents. The computation of  $DV_l$  involves  $f_l$  additions,  $(2 \times f_l)$  subtractions,  $(3 \times f_l)$  multiplications, and  $f_l$  divisions (for  $\alpha_i^l$ ). ( $f_l$  is the number of documents which contain  $t_l$ .) For all terms this will make  $t$  additions,  $2t$  subtractions,  $(m + 3t)$  multiplications, and  $(m + t)$  divisions.

The overall computational requirement can then be summarized as follows: additions,  $(3t) + (t) = 4t$ ; subtractions,  $(0) + (2t) = 2t$ ; multiplications,  $(m + 2t) + (m + 3t) = 2m + 5t$ ; and divisions,  $(m + n) + (m + t) = 2m + n + t$ .

An algorithm that computes exact TDVs by using the cosine similarity of each document with the other documents before and after deleting a term is proposed by Willett [23]. By exact and approximate calculations we mean use of actual document and centroid vectors, respectively. This algorithm has a complexity of  $O(mt)$ . The overall complexity of a computation is generally expressed in terms of the multiplications and divisions involved. The exact calculation of the TDVs by using the CC concept has a complexity of  $O(6t)$ . Usually we would say that  $O(mt) \approx O(t)$  and similarly  $O(6t) \approx O(t)$ . But it should be noticed that  $6 \ll m$ . That is to say, the CC approach is much more efficient than the cosine similarity approach.

Another efficient algorithm that computes TDVs by use of the cosine similarity coefficient is proposed by Crawford [24]. In this algorithm, approximate values for term discrimination values are computed by use of the centroid approach which is similar to the one presented in Section II.

In [23] it is stated that the exact calculation of TDVs using the Dice, overlap, or Jaccard coefficients [3] gives results comparable with the exact calculation that uses the cosine similarity coefficient. Therefore, it can be assumed that the same similarity coefficients would again yield comparable results if the approximation method is used. A comparison of the computation of TDVs with respect to CC and cosine, Dice, overlap, and Jaccard coefficients using Crawford's approach is presented in Table 2. In the Crawford's article only a segment of the algorithm which contains large number of computations was analyzed. However, in Table 2 the computations involved in all of the steps of the algorithm are considered.

In Crawford's algorithm two items are saved for each document. In order to decrease the computational requirement of his algorithm, we will assume that one more item

TABLE 2. The amount of computations involved for TDVs with respect to the CC, cosine, Dice, overlap, and Jaccard coefficients. ( $m$  and  $n$  are ignored, since  $m, n \ll t$ .) (W, for weighted D; B, for binary D.)

Operation	CC		Cosine		Dice		Overlap		Jaccard	
	W	B	W	B	W	B	W	B	W	B
Addition	4t	4t	4t	4t	5t	5t	4t	4t	5t	5t
Subtraction	2t	t	4t	4t	4t	4t	4t	4t	6t	6t
Multiplication	5t	t	5t	2t	2t	0	2t	0	2t	0
Division	t	0	t	t	t	t	t	t	t	t
Square root	0	0	t	t	0	0	0	0	0	0

is saved for each document. That is  $\cos(d_i, G)$ , i.e., the cosine similarity of  $d_i$  with the centroid. To increase the speed of the algorithm, one item for each term will be saved in addition to the above items. That is  $g_j^2$  ( $1 \leq j \leq n$ ), i.e., the square of the centroid entry corresponding to  $t_j$ . These are all the parameters that can be saved to increase the efficiency of the algorithm. (Similar assumptions are made for the other similarity coefficients that are listed in Table 2.)

By use of the assumptions that are stated in the foregoing, the detailed analysis of Crawford's approach in terms of the number of computations is given as follows: Step 0,  $t$  additions; Step 1.1,  $t$  additions,  $t$  multiplications; Step 1.2,  $t$  additions,  $t$  multiplications; Step 1.3, minor [operations can be expressed in terms of  $m$  which can be ignored ( $m \ll t$ )]; Step 2.1, minor; Step 2.2,  $t$  additions,  $4t$  subtractions,  $3t$  multiplications,  $t$  divisions,  $t$  square-root operations; Step 2.3, minor.

If we assume a binary  $D$  matrix, the computational requirement of both approaches will drop. For the CC approach, the  $DV_l$  ( $1 \leq l \leq n$ ) will be reformulated. Consider the individual entries of  $DV_l$  (notice that  $d_{ij}^2 = d_{ij}$  and  $\alpha_i^{-1} - (\alpha_i^l)^{-1} = 1$  if  $d_{ij} = 1$  for  $1 \leq i \leq m, 1 \leq j \leq n$  for a binary  $D$  matrix):

$$\begin{aligned} & \left[ \delta_i - \alpha_i^l \times \left( \frac{\delta_i}{\alpha_i} - d_{ij} \times \beta_j \right) \right] \\ &= \left[ \delta_i - \frac{\alpha_i^l}{\alpha_i} \times \delta_i + d_{ij} \times \beta_j \times \alpha_i^l \right] \\ &= \left[ \left( \frac{\alpha_i - \alpha_i^l}{\alpha_i} \right) \times \delta_i + d_{ij} \times \alpha_i^l \times \beta_j \right]. \end{aligned}$$

If  $\sum_{j=1}^n d_{ij} = x$  (row-sum $_i$ ) then

$$\begin{aligned} (\alpha_i - \alpha_i^l)/\alpha_i &= (1/x - 1/(x - 1))/(1/x) \\ &= -1/(x - 1) = -\alpha_i^l. \end{aligned}$$

Hence, for a binary  $D$  matrix TDV for  $t_l$  is expressed as follows:

$$TDV_l = \sum_{i=1}^{f_l} \alpha_i^l \times (d_{ij} \times \beta_j - \delta_i).$$

The multiplication  $d_{ij} \times \beta_j$  will drop, since  $d_{ij}$  is a binary  $D$  matrix. Therefore, the computational requirement

becomes the following: Step a: The computation for  $\alpha_i^{-1}$  ( $1 \leq i \leq m$ ),  $\beta_j$  ( $1 \leq j \leq n$ ) requires  $2t$  additions and  $t$  divisions; the computation for  $\delta_i$  ( $1 \leq i \leq m$ ) requires  $t$  additions,  $m$  multiplications ( $d_{ij}^2 \times \beta_j$  will drop since  $d_{ij}$  is binary), and  $m$  divisions to obtain  $\alpha_i$ . Step b: from the above definition of  $\text{TDV}_i$  for a binary  $D$  matrix, it is easy to see that the computational requirement is  $t$  additions,  $t$  subtractions, and  $t$  multiplications. In this step  $\alpha_i^l$  will be computed only once for  $d_i$  ( $1 \leq i \leq m$ ), since  $\alpha_i^l = \alpha_i^k$  for ( $1 \leq k \leq n$ ),  $d_{ij} = d_{ik} = 1$ , and  $k \neq l$ .

For a binary  $D$  matrix, the computational requirements of Crawford's algorithm will also drop. Step 0,  $t$  additions; Step 1.1,  $t$  additions; Step 1.2,  $t$  additions; Step 1.3, minor; Step 2.1, minor; Step 2.2,  $t$  additions,  $4t$  subtractions,  $2t$  multiplications,  $t$  divisions, and  $t$  square-root operations.

A comparison of the computations of TDVs, for a binary  $D$  matrix, with respect to all the measures indicated is also given in Table 2. As stated before, the overall complexity of a computation is generally expressed in terms of the multiplications and divisions involved. Accordingly, the complexity of the whole operation for a weighted  $D$  matrix is  $O(6t)$ ,  $O(6t)$ , and  $O(3t)$  for the cosine, CC, and all other similarity coefficients, respectively. Roughly, all of them have the same complexity of  $O(t)$ . (The cosine coefficient involves  $t$  square-root operations which involve multiplication and division operations.) For a binary  $D$  matrix, the CC approach is the most efficient even including the addition and subtraction operations.

This discussion indicates that the exact computation of the TDVs with the CC concept is much more efficient than the exact computation of TDVs with the cosine coefficient [23]. The computational efficiency of the CC-based approach compares favorably with that of the approximation approach [24]. This can be verified by referring to Table 2, which also shows computational comparisons based on different similarity coefficients.

## VI. Experiments for Checking Consistency of the Two Computations

In the experiments, articles from *ACM Transactions on Database Systems* were used to construct a binary  $D$  matrix. The detailed information about the collection and the indexing policy can be seen in [15, 16].

The experiments are performed on the  $D$  matrix (used in the T4 experiment of [15]). The characteristics of the  $D$  matrix are summarized in Table 3.

In the experiments TDVs of all terms are computed using both the CC and the cosine similarity coefficient methodologies. For the latter, because of its efficiency, Crawford's algorithm is used [24]. It is obvious that the two approaches may not assign identical discrimination values to individual terms. This is true even if we calculate the TDVs with two different similarity coefficients. Therefore, to evaluate the consistency, the terms are sorted in ascending order according to their discrimination values, thus showing the poorest

TABLE 3. Characteristics of the binary  $D$  matrix used in the experiments:

A. General information about the binary $D$ matrix.		
No. of documents ( $m$ )		167
No. of terms		539
Avg. no. of term/document ( $x_d$ )		24.94
Avg. no. of document/term ( $t_g$ )		7.73
B. Information about term generalities.		
Term generality range	Number of terms within the range	Total number of document occurrences
3-5	227	922
6-10	201	1505
11-15	66	833
16-20	28	499
21-25	12	268
26-29	5	138
C. Information on indexing exhaustivity of the individual documents.		
Indexing exhaustivity range	Number of documents within the range	Total number of terms in these documents
5	1	5
6-10	8	68
11-15	15	197
16-20	29	513
21-25	30	672
26-30	39	1082
31-35	23	750
36-40	12	434
41-45	7	298
46-50	2	95
51	1	51

and best terms as having the first and last ranks, respectively. The consistency of the two approaches can then be checked by comparing the ranks of terms in the two sorted lists. The overall ranks according to these two metrics might be rather consistent if we consider term groups. However, any two terms may not always have exactly the same rank in the two lists. Therefore, it would be very strict to expect equivalent ranks. We would therefore use range of ranks to check consistency. This will be illustrated in the experiments that follow.

### A. An Experimental Analysis on the Range of Rank Differences

As mentioned at the beginning of this section, TDVs for all terms are computed and ranked according to both methods. The absolute rank difference comparisons of the two methods are given in Table 4. The absolute rank difference is defined as

$$|\text{rank}_{\text{CC}}(t_i) - \text{rank}_{\text{CS}}(t_i)|$$

In this formula  $\text{rank}_{\text{CC}}(t_i)$  and  $\text{rank}_{\text{CS}}(t_i)$  denote the rank of

TABLE 4. The rank difference comparison of the two discrimination measures.

Absolute rank difference	No. of observations	Percent of observations	No. of observations $\leq$ rank difference	Percent of terms $\leq$ rank difference
0	16	3.0	16	3.0
1-5	143	26.5	159	29.5
6-10	94	17.4	253	46.9
11-15	56	10.4	309	57.3
16-20	57	10.6	366	67.9
21-25	35	6.5	401	74.4
26-30	26	4.8	427	79.2
31+	112	20.8	539	100.0

$t_i$  in the sorted lists corresponding to the CC- and the cosine similarity coefficient-based methods, respectively.

Table 4 states that the 16 terms (3.0%) of the total 539 terms have exactly the same ranks; 143 terms (i.e., 26.5% of all terms) have rank differences of 1 to 5 and 159 terms (29.5%) have rank differences less than or equal to 5. The table also shows that 309 terms (57.3%) have rank differences less than or equal to 15. Based on these values, the average rank difference for all terms is 19.39.

In order to interpret the rank differences, we must know the maximum possible average rank difference ( $r_{mpa}$ ). The value of  $r_{mpa}$  for  $n$  terms is expressed as  $n/2$  if  $n$  is even and  $(n^2 - 1)/2n$  if  $n$  is odd (see the appendix). If the average rank difference for all terms is closer or equal to  $r_{mpa}$  [or the ratio, (the average rank difference)/ $r_{mpa}$ , is closer or equal to 1], then this means that a term which has a high TDV according to a measure has a low TDV according to the other. This would indicate the inconsistency of the results of the methodologies in calculating the TDVs.

For  $n = 539$ ,  $r_{mpa} = (n^2 - 1)/2n = 269.50$ . Table 4 shows that 79.2% of the terms have a rank difference less than or equal to 30, which is only 11.1 percent of the maximum possible rank difference ( $30/269.50 = 0.111$ ). The average rank difference (19.39) is only 7.20% of the maximum possible rank difference ( $19.39/269.50 = 0.0720$ ). These results are strong indication of the consistency of the methodologies compared for the calculation of TDVs.

Another way of checking consistency based on rank differences can be to compare between the two sorted lists the average ranks of  $k$  terms at the beginning and at the end of the lists [6]. This is done for  $k = 25$  and is presented in Table 5.  $DV_{CC}$  and  $DV_{CS}$  indicate the term discrimination values for the CC and cosine similarity coefficient, respec-

tively. The ideal rank for the first 25 terms is 13 (i.e.,  $(1 + 2 + \dots + 25)/25$ ) and for the last 25 terms (terms 515 through 539) it is 527.0 (i.e.,  $(515 + 516 + \dots + 539)/25$ ). The first 25 terms of the discrimination measure according to the CC and the cosine similarity coefficient have average ranks of 13.6 (i.e., the average rank of the first 25 terms of the CC in the cosine list) and 14.4, respectively, with respect to each other. Similarly, the last 25 terms of the discrimination measure according to the CC and cosine similarity coefficient have average ranks of 517.1 and 516, respectively, with respect to each other. From these observations, it can be concluded that the term ranks are rather consistent for the first and last 25 terms. However, it can also be seen that both measures are more consistent for the first 25 terms. This is because the difference from the ideal rank is less than 1.5. For the last 25 terms, the difference from the ideal rank is about 10. Notice that the terms at the beginning and at the end of the sorted lists are the poor and good discriminators, respectively. Therefore, it can be observed that both measures are more (less) consistent in determining the poor (good) discriminators.

*B. An Experimental Analysis Based on the Bivariate Correlation Concept*

A bivariate correlation analysis is also conducted on the experimental data by using the ranks of the terms in the two sorted lists [25]. The Pearson's product-moment correlation coefficient (denoted as  $r$ ) and Spearman's rank-order correlation coefficient (denoted as  $r_s$ ) are used for this purpose. In these analyses, if the value of  $r(r_s)$  is close to zero, then there is little or no linear relationship between the two variables being correlated. Whenever  $r(r_s) = 1$ , there is a perfect linear relationship between the two variables (in our case, the ranks of the terms in the sorted lists). That is, identical terms tend to have identical ranks in the two lists. On the other hand, when  $r(r_s) = -1$ , there is a perfect inverse relationship. This means that terms which have high (low) ranks in one sorted list tend to have low (high) ranks in the other sorted list. Associated with the bivariate correlation, a linear regression is also applied to locate the best-fitting straight line for the two measures. This is done by using the scattergram facility of the SPSS package [25].

TABLE 5. Comparison of the average ranks for the first and last 25 terms between the two measures.

	$DV_{CC}$ Cover coefficient	$DV_{CS}$ Cosine similarity coefficient
First 25 terms	13.0	13.6
Last 25 terms	527.0	517.1
First 25 terms	14.4	13.0
Last 25 terms	516.0	527.0

A scattergram is obtained for the ranks of the terms. From this it is observed that the difference between the ranks assigned to a term by the two methods increases as the ranks of the terms increases. In other words, the two methods are more consistent on the low-ranked terms than they are on the high-ranked terms.

In this analysis, a regression line is also obtained. If  $R_{CC}$ ,  $R_{CS}$  denote the ranks obtained by use of the CC and the cosine similarity coefficient, respectively, then the following regression line can be constructed between  $R_{CC}$  and  $R_{CS}$ :

$$R_{CC} = 0.949 R_{CS} + 16.275.$$

(If both methods were perfectly consistent,  $R_{CC}$  would be equal to  $R_{CS}$ .)

For this analysis  $r(r_s)$  values of 0.949 (0.949) with the statistical significances of 0.00000 (0.001) are observed. This indicates a strong linear relationship between the two ranked lists. The very high value of  $r$  also indicates the validity of the regression line [25, p. 279].

From Table 5 and the scattergram output it is observed that the rank differences of terms increases as the ranks of terms increases. The large intercept of 16.275 is mostly due to the increasing differences in the high-order ranks.

To test the validity of this observation, the first 269 terms (half of the total) are considered and the following regression line is obtained:

$$R_{CC} = 0.965 R_{CS} + 3.734.$$

Compared to the line above, the intercept is much closer to zero and the slope is closer to one. For this experiment, the  $r(r_s)$  values are equal to 0.965 (0.965) with the same statistical significance values obtained for the previous case.

These observations prove that the methods are consistent in assigning ranks to terms.

## VII. Use of the TDV and CC Concepts for Indexing

In this section, TDV and CC will be used together to obtain the description of documents, i.e., they will be used for indexing. In A and B, respectively, new strategies for "tunable indexing" and finding the optimum weights for indexing terms will be illustrated.

### A. Tunable Indexing

In this section, the use of the TDV and CC concepts for tunable indexing will be presented. Tunable indexing means selection of appropriate terms which will yield the desired number of clusters within a collection.

The administrator of an IR system may specify the number of clusters that he (she) wants to have within a collection or may assume that in each cluster there should ideally be  $\log_2 m$  number of documents [2]. Assume that  $N_c$  is the number of clusters that the administrator wants to have within a collection. The selection of index terms can be done according to  $N_c$ . That is, one could choose the terms to

be used for indexing by observing the effect of the selected terms on the number of clusters.

Let  $n_c$  be the number of clusters implied by the original  $D$  matrix. The comparison of  $n_c$  with  $N_c$  would indicate one of the following term-selection methodologies:

- (a)  $N_c \approx n_c$ : The current definition of the  $D$  matrix satisfies the administrator's requirement. In this case no change in  $D$  is required.
- (b)  $N_c > n_c$ : The current number of clusters implied by the  $D$  matrix is smaller than the administrator's requirement. In this case the terms which lessen the number of clusters should be deleted from  $T$  (i.e., the set of terms used for the description of documents). These are terms which have negative TDVs.
- (c)  $N_c < n_c$ : The current number of clusters implied by the  $D$  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 are terms which have positive TDVs.

A solution for case (b) is given in the following.

### Algorithm for tunable indexing in the case of $N_c > n_c$ .

- (a) Compute  $TDV_l$  ( $1 \leq l \leq n$ )  
Sort terms in descending order according to their TDVs  
 $l = 0$   
 $T = \phi$
- (b) **repeat**  
 $l = l + 1$   
 $T = T \cup \{t_l\}$  /\*where  $t_l$  is the  $l$ th term of the sorted list\*/  
**until** all documents are defined by at least one term  
Compute  $n_{cl}$   
/\* $n_{cl}$  is the number of clusters in the  $D$  matrix of size  $m \times l$ \*/  
 $N_{max} = n_{cl}$
- (c) **while**  $n_{cl} > N_c$  **and**  $l < n$   
 $l = l + 1$   
 $T = T \cup \{t_l\}$   
Compute  $n_{cl}$   
**endwhile**

Notice that according to the definition of the  $D$  matrix, each document should be defined by at least one term. Step b of the algorithm provides this condition. In Step c the number of terms to be used for the description of documents (equivalently the cardinality of set  $T$ ) is increased. This would decrease the individual decoupling coefficients of documents and this in turn means a decrease in the number of clusters  $n_{cl}$ . Therefore, we will keep adding terms to  $T$  until we bring  $N_c$  to the administrator's requirement. At the end of Step b,  $n_{cl}$  would be the maximum number of clusters that can be observed. This is because terms used for the description of documents are the first  $l$  terms with highest TDVs (since  $TDV_l \geq TDV_{l+1}$  for  $1 \leq l \leq n - 1$ ). Thus, if  $N_c > N_{max}$ , there would be no  $T$  for the description of  $m$  documents, i.e., some documents may remain undefined. Accordingly, the condition for the value of  $N_c$  for this algorithm must be  $n_c \leq N_c \leq N_{max}$ . Whenever  $l$  equals  $n$ ,  $N_c$

will equal  $n_c$ , which is the minimum possible  $n_{cl}$  for this algorithm.

A similar algorithm can be easily devised for Case c. In this case, in Step a of the foregoing algorithm the terms should be sorted in ascending order according to their TDVs and the while statement of Step c should be replaced with the following:

**while**  $n_{cl} < N_c$  **and**  $l < n$

At the end of Step b,  $n_{cl}$  will be equal to  $N_{\min}$ , where  $N_{\min}$  is the minimum possible number of clusters. This is because terms used for the description of documents are the first  $l$  terms with lowest TDVs (since  $\text{TDV}_l \leq \text{TDV}_{l+1}$  for  $1 \leq l \leq n - 1$ ). In Step c, as we increase the cardinality of set  $T$ , the individual decoupling coefficients of documents would increase and this also means increase in the number of clusters  $n_{cl}$ . In this algorithm  $n_{cl}$  will reach its maximum when  $l = n$  and it will be equal to  $n_c$ . Accordingly, the condition for the value of  $N_c$  must be  $N_{\min} \leq N_c \leq n_c$ , ( $N_{\max}$  and  $N_{\min}$  are not known beforehand).

It should be noticed that the above algorithms are approximations, since at the time of adding  $t_l$  to  $T$  there is no guarantee that  $t_l$  will be the term with the minimum TDV among  $l$  terms in the case of  $N_c > n_c$ , and similarly, the term with the maximum TDV among  $l$  terms in the case of  $N_c < n_c$ . For  $n$  terms there are  $n!$  possible permutations for the addition sequence. Therefore, it is not possible to consider all possible sequences to reach the best addition sequence. However, the foregoing approximation methods seem reasonable. Their validity is observed in the experiments for the  $D$  matrix defined in Section VI. Furthermore, when we added more and more terms, the TDVs of individual terms converged to their actual TDVs computed in the original  $D$  matrix. The results of the experiments are

plotted in Figure 1. In the experiments,  $N_{\min} = 9.00$  and  $N_{\max} = 41.40$  are observed for  $l$  being 57 and 215, respectively. This is due to the fact that the terms with low TDVs have high term generality. Therefore, by using a small number of such terms it is possible to define the documents of a collection. However, the opposite is true for the terms with high TDVs, and that explains  $l = 215$  for  $N_c > n_c$ .

In the above algorithm, the modified  $D$  matrix is obtained by increasing the size of  $T$ , i.e., by adding terms. Another way of accomplishing this can be by deleting terms from  $T$  of the original  $D$  matrix. The two approaches are practically equivalent.

In the implementation of the above algorithm the cost of computing  $n_{cl}$  is high. This is due to the repetition. However, there is a more efficient way. Consider the formula for  $n_{cl}$  for a  $D$  matrix of size  $m \times l$ :

$$\begin{aligned} n_{cl} &= \sum_{i=1}^m \delta_i = \sum_{i=1}^m \alpha_i \times (d_{i1}^2 \times \beta_1 + d_{i2}^2 \times \beta_2 + \dots \\ &\quad + d_{il}^2 \times \beta_l) \\ &= \sum_{i=1}^m \alpha_{i,l} \times \gamma_{i,l} \end{aligned}$$

where  $\alpha_{i,l} = (1/\sum_{j=1}^l d_{ij})$  and  $\gamma_{i,l} = (d_{i1}^2 \times \beta_1 + d_{i2}^2 \times \beta_2 + \dots + d_{il}^2 \times \beta_l)$ . If the term  $t_{l+1}$  of the sorted list is also used for indexing, only the decoupling coefficient of the documents which contain  $t_{l+1}$  will be affected. If  $\delta_{i,l}$  and  $\delta_{i,l+1}$  indicate the decoupling coefficient of the document  $d_i$  before and after including the term  $t_{l+1}$ , respectively, then  $\delta_{i,l+1}$  can be defined in terms of  $\delta_{i,l}$  as

$$\delta_{i,l+1} = \alpha_{i,l+1} \times \left( \frac{\delta_{i,l}}{\alpha_{i,l}} + d_{i,l+1}^2 \times \beta_{l+1} \right) = \alpha_{i,l+1} \times \gamma_{i,l+1},$$

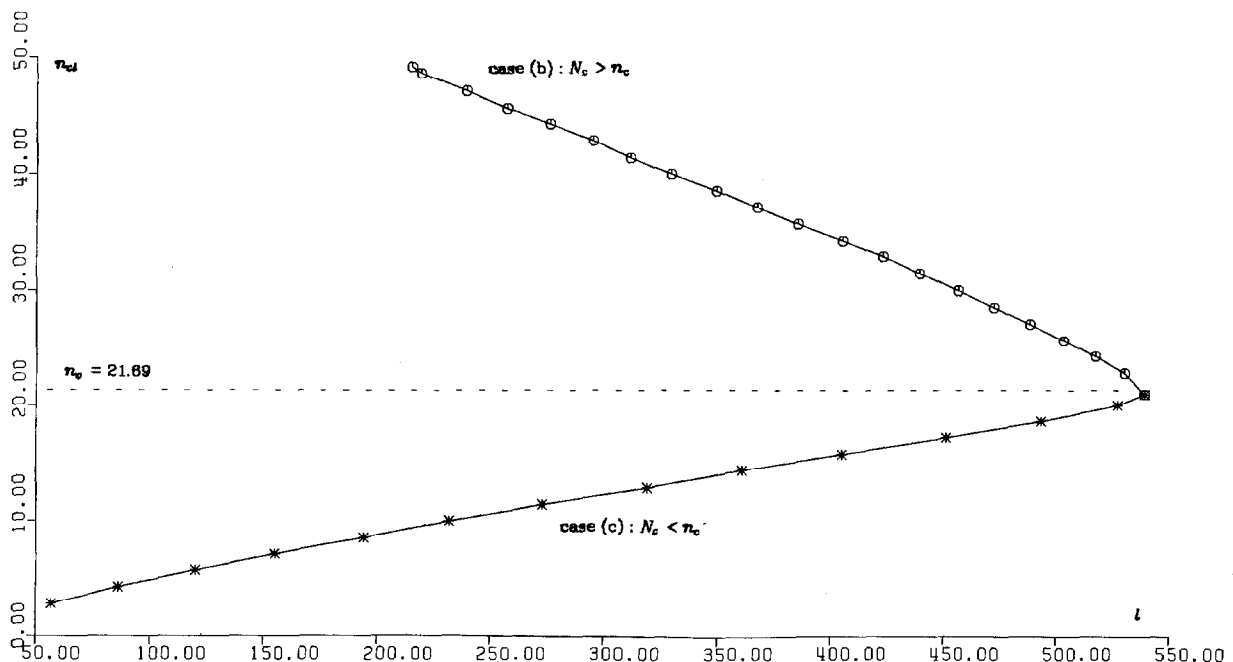


FIG. 1. The results of tunable indexing experiments ( $N_{cl}$  vs  $l$ ).

where  $\alpha_{i,l+1} = (\alpha_{i,l}^{-1} + d_{i,l+1})^{-1}$ .  
Hence

$$n_{c,l+1} = n_{cl} + \sum_{i=1}^{f_{l+1}} \left[ \alpha_{i,l+1} \times \left( \frac{\delta_{i,l}}{\alpha_{i,l}} + d_{i,l+1}^2 \times \beta_{l+1} \right) - \delta_{i,l} \right] = n_{cl} + \sum_{i=1}^{f_{l+1}} (\delta_{i,l+1} - \delta_{i,l}),$$

where  $f_{l+1} = |D_{l+1}|$  and  $D_{l+1} = \{d_i \mid d_i \in D \wedge d_{i,l+1} \neq 0\}$  i.e.,  $f_{l+1}$  is the document frequency of  $t_{l+1}$ . (Notice that  $t_{l+1}$  is the  $l + 1$ st term in the sorted list, i.e., not the  $t_{l+1}$  which is used in the description of the original  $D$  matrix.) In the above formula  $-\delta_{i,l}$  eliminates the effect of  $d_i$  without  $t_{l+1}$  and  $\delta_{i,l+1}$  reintroduces the effect of  $d_i$  on the number of clusters, including the effect of  $t_{l+1}$ .

From the above discussion, it is easy to see that if we maintain the  $\delta_{i,l}$ ,  $\alpha_{i,l}^{-1}$  (row-sum for  $d_i$  in the  $D$  matrix with  $l$  terms), and  $n_{cl}$  values, the computational cost for all  $n_{cl}$  would not be much different from the computational cost of  $n_c$  for the final  $D$  matrix, i.e., the matrix defined by the selected terms. This is because the computation of  $\delta_{i,l}$  also contributes to the computation of the final decoupling coefficients ( $\delta_i$ ) of the documents. As the extreme case, we may assume that the computation of  $n_{cl}$  is done for all terms (i.e., assume that  $n_{cl}$  is also calculated in Step b of the algorithm). The extra computation in this is roughly equal to  $t$  divisions to obtain reciprocal of row sum ( $\alpha_{i,l+1}$ ),  $t$  multiplications for  $\alpha_{i,l+1} \times \delta_{i,l+1}$ ,  $t$  divisions for  $\gamma_{i,l}/\alpha_{i,l}$ , and  $t$  subtractions for  $\delta_{i,l}$ , where  $t$  is the number of nonzero entries in the final  $D$  matrix.

If we assume a binary  $D$  matrix, the computational requirement of  $n_{c,l+1}$  will drop. For this purpose consider the formula given for  $n_{c,l+1}$ . [Notice that  $d_{i,l+1}^2 = d_{i,l+1}$  and  $(\alpha_{i,l+1})^{-1} - \alpha_{i,l}^{-1} = 1$  if  $d_{i,l+1} = 1$  for  $1 \leq i \leq m$ ,  $1 \leq j \leq n$  for a binary  $D$  matrix.]

$$\left[ \alpha_{i,l+1} \times \left( \frac{\delta_{i,l}}{\alpha_{i,l}} + d_{i,l+1} \times \beta_{l+1} \right) - \delta_{i,l} \right] = \left[ \left( \frac{\alpha_{i,l+1} - \alpha_{i,l}}{\alpha_{i,l}} \right) \delta_{i,l} + d_{i,l+1} \times \alpha_{i,l+1} \times \beta_{l+1} \right].$$

Let  $\alpha_{i,l} = 1/x$ , then  $\alpha_{i,l+1} = 1/(x + 1)$ , and  $(\alpha_{i,l+1} - \alpha_{i,l})/\alpha_{i,l} = (1/(x + 1) - 1/x)/(1/x) = -1/(x + 1) = -\alpha_{i,l+1}$ . Accordingly, the increment in the number of clusters becomes  $[\alpha_{i,l+1}(d_{i,l+1} \times \beta_{l+1} - \delta_{i,l})]$ . Hence for a binary  $D$  matrix

$$n_{c,l+1} = n_{cl} + \sum_{i=1}^{f_{l+1}} [\alpha_{i,l+1} \times (d_{i,l+1} \times \beta_{l+1} - \delta_{i,l})].$$

In this case, the extra computation is roughly equal to  $t$  divisions to obtain the reciprocal of row sum ( $\alpha_{i,l+1}$ ),  $t$  multiplications for  $\alpha_{i,l+1} \times (d_{i,l+1} \times \beta_{l+1} - \delta_{i,l})$ , and  $t$  subtractions.

### B. Finding the Optimum Weights for Indexing Terms

A new strategy has been proposed [16] for optimizing user-supplied initial weight values in the  $D$  matrix. In this

strategy, similar to TDV, the discrimination values of documents are computed and this value will be referred to as document significance value, DSV. According to DSV, more significant documents will make the terms distinguishable from each other. That is, introduction of new or relevant articles may help decrease ambiguity of some terms.

As in TDV, deletion of a document may change the number of term clusters. (Obviously, documents exist *a priori* and cannot be added or deleted in collections by users. However, conceptually, we may think of the deletion of a document to compute its DSV.)

In a collection, if we conceptually delete document  $d_i$ , then the new number of term clusters  $n_{cl}$ , which was originally  $n_c$ , is computed as in the TDV concept. That is, the deletion of significant (indifferent) documents would decrease (increase) the number of term clusters by decreasing (increasing) the decoupling coefficient  $\delta'$  of terms. The deletion of indifferent documents would not change the number of clusters. (As in the TDV approach, we will assume that a term is used in at least two documents to be able to use  $n_c$  directly so that the deletion of a document will not decrease the number of terms.)

After the deletion of  $d_i$ , the new number of clusters can be found using the decoupling coefficient of individual terms as follows. Originally (without any document deletion)

$$n_c = n'_c = \sum_{i=1}^n \delta'_i = \sum_{j=1}^n \beta_j \times (d_{1j}^2 \times \alpha_1 + d_{2j}^2 \times \alpha_2 + \cdots + d_{mj}^2 \times \alpha_m).$$

After deleting  $d_i$ ,

$$n_{cl} = \sum_{j=1}^n \delta'_j{}^t = \sum_{j=1}^n \beta_j \times (d_{1j}^2 \times \alpha_1 + d_{2j}^2 \times \alpha_2 + \cdots + d_{l-1,j}^2 \times \alpha_{l-1} + d_{l+1,j}^2 \times \alpha_{l+1} + \cdots + d_{m,j}^2 \times \alpha_m).$$

Similar to the case with term deletion,  $n_{cl}$  can be rewritten as follows:

$$n_{cl} = \sum_{j=1}^m \beta_j^t \times \left( \frac{\delta'_j}{\beta_j} - d_{lj}^2 \times \alpha_l \right),$$

$$n_{cl} = n_c + \sum_{j=1}^{f_l} \left[ \beta_j^t \times \left( \frac{\delta'_j}{\beta_j} - d_{lj}^2 \times \alpha_l \right) - \delta_j' \right],$$

for all  $t_i \in T_i$

where  $f_l = |T_l|$  and  $T_l = \{t_j \mid t_j \in T \wedge d_{lj} \neq 0\}$ , i.e.,  $f_l$  is the number of terms used for the description of  $d_l$ .

After knowing  $\text{TDV}_j$  and  $\text{DSV}_i$ , we modify the weight of  $t_j$  in  $d_i$  (i.e.,  $d_{ij}$ ) by using the following product:

$$\text{DSV}_i \times \text{TDV}_j = \frac{n_c}{n_{cl}} \times \frac{n_c}{n_{cj}} = \frac{n_c^2}{n_{cl} \times n_{cj}}.$$

The above product is referred to as the "weight modification factor" for the weight of  $t_j$  in  $d_i$  and is abbreviated as  $\Delta_{ij}$ .

Accordingly, the  $D$  matrix will be redefined as follows:

$$d_{ij} = \Delta_{ij} \times d_{ij}, \quad \text{for } 1 \leq i \leq m, 1 \leq j \leq n.$$

This product will satisfy the following interpretations:

- (a) The occurrence of a good term discriminator ( $t_j$ ) in a significant document ( $d_i$ ) should have an importance greater than or equal to the occurrence of a poor (good) term discriminator ( $t_i$ ) in a significant (less significant) document ( $d_k$ ). Accordingly, if  $d_i$  is a significant document and  $t_j$  is a good discriminator, and  $d_k$  is a less significant document and  $t_i$  is a poor term discriminator, then the occurrence of  $t_j$  in  $d_i$  should have an importance much greater than the occurrence of  $t_i$  in  $d_k$ . (Notice that  $n_{ci} \times n_{cj} \ll n_{ck} \times n_{cl}$ , and hence  $\Delta_{ij} \gg \Delta_{kl}$ .)
- (b) The occurrence of a good term discriminator ( $t_j$ ) in an insignificant document ( $d_i$ ) should have an importance comparable to that of the occurrence of a poor term discriminator ( $t_i$ ) in a significant document ( $d_k$ ). (Notice that  $n_{ci} \times n_{cj} \approx n_{ck} \times n_{cl}$ , and hence  $\Delta_{ij} \approx \Delta_{kl}$ .) However, the occurrence of a poor term discriminator ( $t_n$ ) in an insignificant document ( $d_m$ ) should have less importance than the occurrence of  $t_j$  ( $t_i$ ) in  $d_i$  ( $d_k$ ). [Notice that  $n_{ci} \times n_{cj} < n_{cm} \times n_{cn}$  ( $\Delta_{ij} > \Delta_{mn}$ ) and  $n_{ck} \times n_{cl} < n_{cm} \times n_{cn}$  ( $\Delta_{kl} > \Delta_{mn}$ ).]

The weight modification factor modifies the weight of a term in a particular document by observing the importance of the term and at the same time by observing the significance of the document with respect to the other documents of the collection [26].

The  $D$  matrix is modified iteratively, that is after an adjustment with  $\Delta_{ij}$  ( $1 \leq i \leq m, 1 \leq j \leq n$ ) the new value for  $\Delta_{ij}$  is computed and terms are readjusted within a loop until the  $D$  matrix reaches a near steady state in terms of a stopping criterion.

The stopping criterion can be one of the following [16]:

- (a) The number of clusters of two consecutive iterations are equal to each other.
- (b) The documents chosen as the cluster seeds in two consecutive iterations are identical, or at least have a commonality greater than a given threshold.
- (c) If we sort the significance value of documents and discrimination value of terms in two consecutive iterations, then the ranks of the documents (terms) in two consecutive iterations must be compatible.

For this purpose a rank consistency measure as discussed in Section VI can be used.

### VIII. Conclusions

In this article we presented the concepts of indexing, CC, and TDV. An efficient way of computing the "exact" TDVs by use of the CC concept is covered in detail. The computational cost of this methodology is much less than an earlier method which uses a similarity coefficient. The computational cost of the CC-based method is also compared to

that of another approach which calculates the approximate TDVs in an efficient way. This comparison yielded favorable results on behalf of the CC-based approach.

In the experiments we have observed that TDVs obtained based on the CC concept are consistent with those obtained by an earlier method based on the cosine similarity coefficient. In the experiments, 167 documents are defined with 539 terms and 4165 term occurrences. The indexing type was binary.

In the article, a new technique for tunable indexing is introduced according to the CC concept. The technique allows the user to specify the number of clusters to be observed within the collection. The terms are selected according to their TDVs in such a way that the document collection representation (the  $D$  matrix) contains the specified number of clusters.

The article also introduces a new heuristic to optimize the representation of a document collection. This heuristic modifies the weight of a term occurrence  $d_{ij}$  according to both the importance of  $d_i$  and  $t_j$ . For this purpose, as for TDV, we have introduced the concept of DSV for each document of a collection.

The foregoing results and those of [18] show that the CC concept which emerged in our clustering research has value in other areas of IR. Currently, more research on the validity of the CC concept within the context of IR query processing is being pursued.

### Appendix

In order to interpret the values given for the rank differences we need to know the maximum possible average rank difference,  $r_{mpa}$ .

**PROPOSITION:** The maximum possible average rank difference for  $n$  terms,  $r_{mpa}$ , is equal to  $n/2$  if  $n$  is even and is equal to  $(n^2 - 1)/2n$  if  $n$  is odd.

**PROOF:** In order to have the maximum rank difference, let us interchange the position of the 1st and the  $n$ th terms. In this way we would have two terms with the rank difference of  $n - 1$  ( $n - 1$  is the maximum possible rank difference for terms). If the places of the 2nd and  $(n - 1)$ st terms are interchanged for these two terms, we would obtain a rank difference of  $n - 3$ . Repeating this in the same way for all  $d_i$  ( $1 \leq i \leq k$ ), i.e., interchange term with rank  $i$  with the one whose rank is  $(n - i + 1)$ , we obtain a rank difference of  $n - 2i + 1$ , where  $k$  is equal to  $n/2$  and  $\lfloor n/2 \rfloor$  for  $n$  being even and odd, respectively. This is because if  $n$  is even we would be able to interchange the terms with ranks  $n/2$  and  $n/2 + 1$ . However, if  $n$  is odd, then the middle term with rank  $\lfloor n/2 \rfloor$  will stay at its original place.

The sum of all rank differences is

$$2(n - 1) + 2(n - 3) + 2(n - 5) + \dots + 2\left(n - \left\lfloor \frac{n}{2} \right\rfloor + 1\right).$$

If  $n$  is even the above summation can be written as follows:  
 $S_c = 2[1 + 3 + 5 + \dots + (n - 1)] = 2 \times (\text{sum of odd numbers up through } n - 1)$ .

If  $n$  is odd we would have

$S_0 = 2[0 + 2 + 4 + \dots + (n - 1)] = 2 \times (\text{sum of even numbers up through } n - 1)$ .

The sum of the even (odd) numbers up through  $2m$  ( $2m - 1$ ) can be written as  $m(m + 1)[m^2]$ . By setting  $2m = m - 1$  [ $2n - 1 = n - 1$ ],  $S_c$  and  $S_0$  can be rewritten as  $n^2/2$  and  $(n^2 - 1)/2$ , respectively.

## Acknowledgments

We would like to express our gratitude for the comments of our referees and especially to Dr. Edward A. Fox for his numerous suggestions that helped to improve this article.

## References

1. Van Rijsbergen, C. J. *Information Retrieval, 2nd ed.* London: Butterworths; 1979.
2. Salton, G. *Dynamic Information and Library Processing.* Englewood Cliffs, NJ: Prentice Hall; 1975.
3. Salton, G.; McGill, M. J. *Introduction to Modern Information Retrieval.* New York: McGraw Hill; 1983.
4. Maron, M. E. "Depth of Indexing." *Journal of the American Society for Information Science.* **19**:224-228; 1978.
5. Borko, H. "Toward a Theory of Indexing." *Information Processing and Management.* **13**:355-365; 1977.
6. Salton, G. "A Theory of Indexing." *Regional Conference Series in Applied Mathematics*, No. 18. Philadelphia, PA: Society for Industrial and Applied Mathematics; 1975.
7. Salton, G.; Wong, A.; Yang, C. S. "A Vector Space Model for Automatic Indexing." *Communications of the Association for Computing Machinery.* **18**(11):613-620; 1975.
8. Yu, C. T., Salton, G. "Precision Weighting—An Effective Automatic Indexing Method." *Journal of the Association for Computing Machinery.* **23**(1):76-88; 1976.
9. Salton, G.; Wong, A.; Yu, C. T. "Automatic Indexing Using Term Discrimination and Term Precision Measurements." *Information Processing and Management.* **12**:43-51; 1976.
10. 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):175-186; 1981.
11. Cooper, W. S.; Maron, M. E. "Foundations of Probabilistic and Utility-Theoretic Indexing." *Journal of the Association for Computing Machinery.* **25**(1):67-80; 1978.
12. Croft, W. B.; Harper, D. H. "Using Probabilistic Strategies with no Relevance Information." *Journal of Documentation.* **35**:285-295; 1979.
13. Can, F.; Ozkarahan, E. A. "A Clustering Scheme." *Proceedings of the ACM SIGIR Conference*, Bethesda, Md.; June 1983: 115-121.
14. Can, F.; Ozkarahan, E. A. "Two Partitioning Type Clustering Algorithms." *Journal of the American Society for Information Science.* **35**(5):268-276; 1984.
15. Can, F.; Ozkarahan, E. A. "Similarity and Stability Analysis of the Two Partitioning Type Clustering Algorithms." *Journal of the American Society for Information Science.* **36**(1):3-14; 1985.
16. Can, F. "A New Clustering Scheme for Information Retrieval Systems Incorporating the Support of a Database Machine." *Ph.D. dissertation.* Ankara: Department of Computer Engineering, Middle East Technical University; January 1985.
17. Ozkarahan, E. *Database Machines and Database Management.* Englewood Cliffs, NJ: Prentice-Hall; 1986.
18. Can, F.; Ozkarahan, E. A. "Concepts of the Cover Coefficient-Based Clustering Methodology." *Proceedings of the ACM SIGIR Conference*, Montreal; June 1985: 204-211.
19. Everitt, B. S. "Unresolved Problems in Cluster Analysis." *Biometrics.* **35**:169-181; 1979.
20. Can, F.; Ozkarahan, E. A.; Kutluay, S. "Validity Analysis of the Cover Coefficient Based Clustering Methodology." Unpublished.
21. Nelson, M. J.; Tague, J. M. "Split Size-Rank Models for the Distribution of Index Terms." *Journal of the American Society for Information Science.* **36**(5):283-296; 1985.
22. Salton, G.; Wong, A. "Generation and Search of Clustered Files." *Association for Computing Machinery Transactions on Database Systems.* **3**(4):321-346; 1978.
23. Willett, P. "An Algorithm for the Calculation of Exact Term Discrimination Values." *Information Processing and Management.* **21**(3):225-232; 1985.
24. Crawford, R. G. "The Computation of Discrimination Values." *Information Processing and Management* **11**:249-253; 1975.
25. Nie, N. H.; Hull, C. H.; Jenkis, J. G.; Steinbrenner, K.; Bent, D. H. *SPSS Statistical Package for the Social Sciences, 2nd ed.* New York: McGraw-Hill; 1975.
26. Croft W. B. "Document Representation in Probabilistic Models of Information Retrieval," *Journal of the American Society for Information Science.* **33**:451-457; 1981.
27. Borko, H. "Automatic Indexing: A Tutorial." *ACM SIGIR Forum.* **16**(2):9-13; 1982.
28. Lancaster, F. W. *Information Retrieval Systems: Characteristics, Testing and Evaluation, 2nd ed.* New York: Wiley; 1975.