

On Relevance and Two Aspects of the Organizational Memory Problem*

Garett O. Dworman Steven O. Kimbrough
Stephen E. Kirk Jim R. Oliver

December 15, 1997

address correspondence to:
Steven O. Kimbrough
University of Pennsylvania
The Wharton School
3620 Locust Walk, Suite 1300
Philadelphia, PA 1910-6366
tel: (215) 898-5133
fax: (215) 898-3664
kimbrough@wharton.upenn.edu

*Thanks to David C. Blair, Michael Gordon, and Scott Moore for comments and discussion on an earlier version of this paper. Thanks to Abeer Y. Hoque for help on the PIRS experiment. File: orgmems5.tex. This material is based upon work supported by, or in part by, the U.S. Army Research Office under contract/grant number DAAH04-1-0391. Special thanks to The Historic New Orleans Collection for support and feedback on the Clarence John Laughlin archives.

Contents

1	Introduction	1
2	Problem 1: Document Retrieval	2
3	Problem 2: Resource Location	6
4	Theory	8
4.1	A Geometric Perspective on the DCB Algorithm	9
4.2	An IR Perspective on the DCB Algorithm	11
4.2.1	The IR model	12
4.2.2	DCB and the IR Model	15
5	Empirical Analysis	16
5.1	Comparison with Alternatives via simulation	17
5.1.1	The Four Algorithms	17
5.1.2	Simulation Results	19
5.2	Experiments with Human Subjects	21
5.2.1	Human Subjects Results	23
5.2.1.1	Group 9	23
5.2.1.2	Group 10	24
5.2.1.3	The combined results	25
5.3	Comparison of DCB to Inquiry Based Upon Human Subject Results	25
5.3.1	Inquiry results	27
5.3.2	Inquiry results excluding ties	28
5.4	Discussion of Empirical Analysis	30
6	Conclusion	31
	References	34

1 Introduction

It is often the case that the clearest view of general principles is through particular examples. Consider, then, the following two rather specific problems.

1. A Congressional oversight committee has a series of questions about certain operating policies of the U.S. Coast Guard. Wishing to have its questions answered, the committee sends a letter to the Coast Guard, specifying its questions. In answering the letter, the Coast Guard must find pertinent information. This information is, for the most part, contained in documents, rather than structured databases. Among the most important and useful documents, if they exist, are the responses the Coast Guard has previously made to similar questions from Congress. Unfortunately, the scale of the document database is fairly large. The Coast Guard answers about 1000 Congressional letters of inquiry each year and several years of prior answers will be worth attending to. Further, the number of potentially relevant reports, memoranda, and so forth, is in the tens of thousands. How can the needed documents effectively be found?
2. Congressional questions need to be answered accurately, reliably, quickly, and in a manner consistent with existing policies. Consequently, it is important that the officer assigned to drafting the replies be as knowledgeable as possible about the subject(s) at hand. Given a particular letter of inquiry, which of the 35,000 employees of the Coast Guard have useful knowledge for the questions at hand?

These two particular problems are each special cases of very general, commonly-encountered problems, which we call (respectively) *the document retrieval problem*, and *the resource location problem*.¹ Further, these two problems are but two facets of what is commonly called *the organizational (or corporate) memory problem*, which may be characterized as follows:

¹There is a long-standing, good-sized research community dedicated to the document retrieval problem. That community tends to use the term *information retrieval* where we use *document retrieval*. We prefer the term *document retrieval* as a way of alluding to the distinction between the retrieval of objects/resources and the retrieval of documents. See [9] for a useful review of the, quite extensive, relevant literature.

The organizational memory problem. Information pertinent to the task at hand has passed through the organization. Was this information captured? If so, how can it be effectively retrieved and brought to bear on the present task?

These problems—document retrieval, resource location, and, in general, organizational memory—are enormously complex and challenging. There is a substantial, long-standing literature on organizational memory (see [23] for a review and an interesting discussion) and the relevant issues surely are broader than can be addressed by computational solutions alone. Our aspirations in this paper can hardly be to analyze the subject fully, let alone to proffer a definitive solution. We do, however, believe that computational solutions can be helpful and we wish to present and develop some general and broadly useful ideas. For the sake of clarity and concreteness, however, we will for the most part couch our discussion in terms of the two specific Coast Guard problems, described above.

We begin with a discussion of problem 1, above, the document retrieval problem as it applies to the Coast Guard answering Congressional questions. We shall address generalizations in the sequel, for we wish to present some very general results regarding computational support for addressing problems of organizational memory.

2 Problem 1: Document Retrieval

We have to expect imperfection in any system for finding documents that are useful for answering a random question from Congress. One reason is that it is difficult to predict what Congress will ask about. A second reason is that often the vocabulary of the Congressional question will not match very well with the vocabulary in the Coast Guard document collection [2, 5]. Add to this the fact that the document collections in question are much too large to examine manually, document by document, and problem 1 becomes quite formidable.

Our problem is to find—in the face of considerable noise and large collections—documents that are relevant to a particular question. Two highly desirable features of any approach to this problem are:

1. *Ranked retrieval.* The method of retrieval should rank the documents

by order of (purported) relevancy to the question. Without ranking, it is simply infeasible to inspect inevitably large, and undifferentiated, retrieval sets. The value of ranked retrieval has been well-established in the Information Retrieval literature (c.f., [3]).

2. *Associative retrieval.* The method of retrieval should take account of not only whether a document has the search term (keyword), but also of the overall lexical constitution of the document. Given a particular question, and a plausible keyword for capturing the topic of the question, it is entirely possible to have two highly relevant documents, one of which contains the keyword and one of which does not contain the keyword. This may, and commonly does, happen for many different reasons. What is much less likely is that the other words in these two documents should be very dissimilar. In addition, two documents may very easily contain the same plausible keyword, yet be relevant to very different degrees. When this happens, it is likely that the non-keyword vocabularies of these documents are quite different. Searching, for example, on the term DNA will yield documents about the Digital Equipment Corporation's network architecture, as well as documents from the microbiology literature. Searching on the term ATM will yield documents about banking, as well as documents about packet switching. It is the non-keyword terms in these documents that will determine whether or not they are relevant. (Were you looking for information on computer communications or for information on protein synthesis? Were you looking for information about automatic teller machines or about asynchronous transfer mode?) The cases are extreme, but the phenomenon is real and common [7]. It is exacerbated on the World Wide Web. Web search engines are notorious for poor performance on multiple term searches.²

Our aim now is to describe a particular computational approach for dealing with the document retrieval problem. In order to understand the approach, and how it answers the requirements for ranked retrieval and associative retrieval, let us consider, and reason through, the following example.

Suppose, for the sake of the discussion, that all of the Coast Guard's

²William Chang of Infoseek has identified a fine example: *home run record*. Others are easy to find.

Congressional questions, and their answers, for the past several years have been collected and indexed. The indexing is done by determining, for every document and every keyword in an appropriately thorough list, whether a document contains a keyword. Having done this, we in effect have a matrix, called K , whose entries are all 1s and 0s, whose rows correspond to keywords, and whose columns correspond to documents [22, 20]. A particular K might look like this.

$$K = \begin{pmatrix} 1 & \dots & 0 & 1 \\ \vdots & k_{i,j} & \vdots & \vdots \\ 1 & \dots & 1 & 0 \end{pmatrix} \text{ (where: rows = keyterms and columns = documents)} \quad (1)$$

Element $k_{i,j}$ is 1 if keyword i occurs (at least once) in document j ; otherwise it is 0. This is a quite general way to think about document indexing. We are simply recording whether the documents in question (the columns) do or do not contain the index terms in question (the rows).

Taking a simple example, suppose we have six keywords and six documents, with the following K matrix.

$$K = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \quad (2)$$

(The fact that K is square in this example is not significant. Our remarks apply no matter what shape K has.)

Suppose further that given the question at hand we are interested in information pertaining to the first keyword. Which of the documents are likely to be of interest to us? Note that documents (columns) i, iii, v, and vi contain keyword i (row 1). Presumably, these documents are relevant to the question. Note further, however, that document ii is almost identical to document i. Document ii lacks keyword i, but is otherwise identical in K to document i.

By the reasoning for the importance of associative retrieval, it is a reasonable guess that document ii is relevant, even though it does not contain

the keyword we have chosen. How relevant? An intuitive case can be made that document ii is presumably *more* relevant than document iv. Not only does document iv lack the keyword search term (row 1), but it also lacks keywords iii, iv, and vi, which commonly occur in the documents that have keyword i. With similar reasoning, we can cast some doubt on the relevancy of document vi. Even though it *has* the keyword, it would seem to resemble document iv more than the documents with keyword i.

There is a simple way to operationalize these intuitions and make them mathematically concrete. We call this the DCB method (or algorithm), after David C. Blair (University of Michigan), who suggested a close variant of this method to us (both in conversation and in an unpublished working paper [4]) for use in document retrieval.

The DCB algorithm begins with the K matrix and a search term, or keyword. (Multiple search terms and boolean combinations of them are possible, but discussion of this is beyond the scope of this paper.) Three steps follow. First, the K matrix is multiplied by its transpose to produce the L matrix. In terms of our present example, we have:

$$L \stackrel{\text{def}}{=} K \cdot K^T = \begin{pmatrix} 4 & 3 & 1 & 2 & 1 & 2 \\ 3 & 5 & 2 & 3 & 1 & 2 \\ 1 & 2 & 2 & 2 & 0 & 2 \\ 2 & 3 & 2 & 3 & 0 & 2 \\ 1 & 1 & 0 & 0 & 2 & 1 \\ 2 & 2 & 2 & 2 & 1 & 3 \end{pmatrix} \quad (3)$$

The interpretation of L is straightforward. L is always a square, symmetric matrix with number of rows (and columns) equal to the number of keywords, i.e., the number of rows in K . The diagonal elements of L indicate the number of documents containing the corresponding keyword. In our example, $l_{2,2} = 5$, indicating that 5 documents contain keyword ii, which may be verified by examining K . The off-diagonal elements, the $l_{i,j}$ s, indicate the number of documents containing *both* keyword i and keyword j . In our example, 3 documents contain both keyword i and keyword ii.

Interpreting an entire row of L is revealing. Looking at row 1, we see that 4 (of 6) documents contain keyword i, 3 contain both keywords i and keyword ii, 2 contain both keywords i and iv, and so on. Evidently, keyword 2 tends to be associated with keyword i, while keywords iii and v tend not to be. Keywords iv and vi are intermediate. Given this, a natural way of

scoring a document’s relevance (in this case, to keyword i) is to give it 4 points if it has keyword i, 3 points for keyword ii, 1 point for keyword iii, and so on. These calculations are performed, in effect, by multiplying L by K to produce the M matrix. In terms of our present example, we have, as step 2 in the DCB algorithm:

$$M \stackrel{\text{def}}{=} K \cdot K^T \cdot K = \begin{pmatrix} 12 & 8 & 9 & 4 & 7 & 7 \\ 15 & 12 & 11 & 6 & 6 & 8 \\ 9 & 8 & 5 & 2 & 3 & 3 \\ 12 & 10 & 8 & 3 & 4 & 5 \\ 3 & 2 & 2 & 3 & 4 & 2 \\ 11 & 9 & 6 & 3 & 6 & 4 \end{pmatrix} \quad (4)$$

Looking at M , we see that the DCB algorithm has produced a relevancy ranking of all the documents, and in doing so uses a form of associative retrieval. Step 3 of the DCB algorithm produces the DCB sort: looking at the row of M corresponding to the keyword search term, treat the entries as scoring the relevance of their corresponding documents, with higher numbers indicating greater relevance. In our present example—with our focus on word i—we see that document i is judged most relevant, followed (in order) by documents: iii, ii, v-vi, and iv.

Happily, this ranking is consistent with, and more specific than, the intuitive reasoning presented above. Judging just how good the DCB algorithm is is a topic we defer until later. We now consider the second facet of our problems for organizational memory.

3 Problem 2: Resource Location

Recall problem 2, the resource location problem example: Given a particular letter of inquiry, which of the Coast Guard employees have useful knowledge for the question at hand?

Our main point in this section is that problem 2, properly understood, is very close in structure to problem 1 and can also usefully be addressed with the DCB algorithm. The point is best seen with an example. We will work with an extremely simple example, but without loss of generality.

Assume the Coast Guard has kept some simple records pertaining to meetings. The record of a meeting includes information on who attended the

meeting and what issues were discussed. (In fact, this information is among that now captured in regulation meeting reports.) The meeting records may be represented in a matrix of 1s and 0s, as before. For the present example, we have the following K matrix.

$$K = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \text{ where } \begin{pmatrix} \text{object } i & = & \text{Person } a \\ \text{object } ii & = & \text{Person } b \\ \text{object } iii & = & \text{Issue } a \\ \text{object } iv & = & \text{Issue } b \\ \text{object } v & = & \text{Meeting } a \\ \text{object } vi & = & \text{Meeting } b \end{pmatrix} \quad (5)$$

K here is similar to K for documents, above, but there are some differences. K here is for information objects in general. An entry, $k_{i,j}$ has the interpretation: object i is associated with object j . Consequently, K for information objects is square; has a row (and a column) for each object indexed in the system; and has 1s on the diagonal (everything is associated with itself). Further, we assume that K is symmetric: if object i is associated with object j , then so is j with i . This last assumption could be relaxed, but doing so makes little sense for the present example.

Finally, the indexed objects in K need not be homogeneous. In our present example, objects i and ii are people, objects iii and iv are issues, and objects v and vi are meetings. Thus, according to K , person a (object i) attended meeting a (object v), but not meeting b (object vi), while person b (object ii) attended meeting b (object vi) but not meeting a (object v). Notice that people and issues are *not* directly linked in this example. The point of the example is to find the most relevant people by finding the people most closely connected with the issue at hand. The problem comes to this: given an issue (arising from the Congressional query), which people are most closely connected to it? The point of the example is that the DCB algorithm can reasonably rank people by issues, even without having any direct and explicit connection of people with issues. Notice that, as in the case of document retrieval,

1. Ranked retrieval, and
2. Associative retrieval

are very desirable features of any approach to this problem. Ranked retrieval is needed because potentially very many people have relevant knowledge of the issue at hand. Associative indexing and retrieval is needed because we want to exploit implicit information. We assume in the present example that no person is directly linked to any issue. Even were this not the case, it should be useful to capture indirect information about a person’s knowledge. Here, we apply DCB to implement the heuristic that people who attended meetings at which the issue at hand was discussed plausibly have good knowledge of those issues.

Given K , the first step in the DCB algorithm is to calculate L . For the present example, this yields expression (6).

$$L \stackrel{\text{def}}{=} K \cdot K^T = \begin{pmatrix} 2 & 0 & 1 & 0 & 2 & 0 \\ 0 & 2 & 0 & 1 & 0 & 2 \\ 1 & 0 & 2 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 & 0 & 2 \\ 2 & 0 & 2 & 0 & 3 & 0 \\ 0 & 2 & 0 & 2 & 0 & 3 \end{pmatrix} \quad (6)$$

Next in the DCB algorithm, we calculate M , yielding expression (7).

$$M \stackrel{\text{def}}{=} K \cdot K^T \cdot K = \begin{pmatrix} 4 & 0 & 3 & 0 & 5 & 0 \\ 0 & 4 & 0 & 3 & 0 & 5 \\ 3 & 0 & 4 & 0 & 5 & 0 \\ 0 & 3 & 0 & 4 & 0 & 5 \\ 5 & 0 & 5 & 0 & 7 & 0 \\ 0 & 5 & 0 & 5 & 0 & 7 \end{pmatrix} \quad (7)$$

Finally, we interpret M . Suppose we are interested in finding people who know about issue a (object iii). As expected, M tells us that person a (object i), who attended the meeting at which issue a was discussed, is more relevant (score 3) than person b (score 0), who did not attend the meeting at which issue a was discussed.

4 Theory

So far we have discussed the DCB algorithm from a procedural and fairly intuitive point of view. Our aim has been to give the reader a sense of what it

does and how it works. Our aim now, in this section, is to provide a deeper, more mathematical perspective on the DCB algorithm. This done, we shall see that DCB is but one of a large family of potentially interesting algorithms for ranked retrieval with associative indexing.

Next, we put DCB into the context of the Information Retrieval (IR) literature. The IR field has investigated many techniques for document retrieval. We find it fruitful to see how DCB fits into a general IR model.

4.1 A Geometric Perspective on the DCB Algorithm

Recall that K is an m -by- n and 0 – 1 matrix. It represents an indexing of n documents (or other objects) with m index terms. An entry $k_{i,j}$ in K is 0 or 1 depending on whether object j is associated with indexing term i . Let $\vec{k}(j)$ be the j^{th} column of K , i.e., the vector of indices for object (or document) j . We may think of K as representing each indexed object as situated at a point in an m -dimensional index space. More precisely, the DCB algorithm uses a representation in which each indexed object is placed at some vertex of an underlying m -dimensional 0 – 1 hypercube. Which vertex? For object j it is $\vec{k}(j)$.

This much is implicit just in the K representation. Let us now see how DCB exploits this information. Recall that $L \stackrel{\text{def}}{=} K \cdot K^T$ in DCB. Let $\vec{l}(i)$ be the i^{th} row of L . Given this, then the final rank score for object (e.g., document) j on indexing term (e.g., word or phrase) i is:

$$\vec{l}(i) \cdot \vec{k}(j) \tag{8}$$

Note that object h has a higher rank than j (is judged by DCB to be more relevant to the query for indexing term i) if and only if

$$\vec{l}(i) \cdot \vec{k}(h) > \vec{l}(i) \cdot \vec{k}(j) \tag{9}$$

We can get a clearer sense of what this means if we normalize $\vec{l}(i)$:

$$\vec{l}(i)^N \stackrel{\text{def}}{=} \frac{\vec{l}(i)}{L_{i,i}} \tag{10}$$

That is, $\vec{l}(i)^N$ is obtained from $\vec{l}(i)$ by dividing each of its elements by $L_{i,i}$. Recall the interpretation of L and its rows, $\vec{l}(i)$: the j^{th} element

($j \in \{1, 2, \dots, m\}$) of $\vec{l}(i)$ is the number of objects (e.g., documents) that contain at least one instance of the j^{th} indexing term (e.g., word), given that the objects also contain at least one instance of the i^{th} indexing term. Clearly, no element of $\vec{l}(i)$ can have a larger value than its i^{th} element, i.e., $L_{i,i}$. Thus, the elements of $\vec{l}(i)^N$ range from 0 to 1. In other words, $\vec{l}(i)^N$ fits inside the m -dimensional 0 – 1 hypercube implied by the K representation.

Given this, $\vec{l}(i)^N$ represents, in index space, a kind of average object (from the n objects collected) that is associated with index term i . That is, $\vec{l}(i)_j^N$, the j^{th} element of $\vec{l}(i)^N$ is the percentage of objects with index term i that also have index term j . We may calculate a rank score with the normalized vector by analogy with expression (8), above:

$$\vec{l}(i)^N \cdot \vec{k}(j) \tag{11}$$

This expression, (11), is easily interpreted. From the definition of the scalar, or dot, product of two vectors,

$$\vec{l}(i)^N \cdot \vec{k}(j) = |\vec{l}(i)^N| |\vec{k}(j)| \cos \theta_{i,j} \tag{12}$$

where $\theta_{i,j}$ is the angle between $\vec{l}(i)^N$ and $\vec{k}(j)$. Also, an object h has a higher rank than j (is judged by this normalized version of DCB to be more relevant to query i) if and only if

$$\vec{l}(i)^N \cdot \vec{k}(h) > \vec{l}(i)^N \cdot \vec{k}(j) \tag{13}$$

or equivalently if and only if

$$|\vec{l}(i)^N| |\vec{k}(h)| \cos \theta_{i,h} > |\vec{l}(i)^N| |\vec{k}(j)| \cos \theta_{i,j} \tag{14}$$

The analog to expression (14) for the original, unnormalized version of the DCB algorithm is:

$$|\vec{l}(i)| |\vec{k}(h)| \cos \theta_{i,h} > |\vec{l}(i)| |\vec{k}(j)| \cos \theta_{i,j} \tag{15}$$

and both algebraically reduce to

$$|\vec{k}(h)| \cos \theta_{i,h} > |\vec{k}(j)| \cos \theta_{i,j} \tag{16}$$

so we see that the normalization, which is helpful for interpreting the DCB algorithm, does not affect the final rankings and only introduces additional computational costs in an implementation.

Let us reflect on expression (16), which is the key to understanding the DCB algorithm, at least in theory. What expression (16) says is that an object's rank score depends on two things: A , the length of its indexing vector from the K matrix and B , the cosine of the angle between the object's indexing vector and the average profile vector produced from L , $\vec{l}(i)$ (or equivalently, $\vec{l}(i)^N$). The rank score is just the product, $A \cdot B$. We note that under the representation implicit in K :

1. $0 \leq |\vec{k}(h)| \leq m$ ($m =$ the number of indexing terms in K)
2. $0 \leq \cos \theta_{i,j} \leq 1$ (the m -dimensional 0 – 1 hypercube implicit in L lies in the positive hyperquadrant)

So, a document gets ranking points to the degree that it: A , has more indexing terms present, and B , lies coincident on, or close to, the average document with the indexing term in question, as measured by $\vec{l}(i)$.

The DCB algorithm, thus, measures a sort of distance in indexing space and it ranks objects according to how close they lie to a sort of average document, used as a benchmark. Pretty obviously, this is a reasonable sort of thing to do. At least it's worth trying. And at least as obviously, there are many other ways to measure distance and to establish a benchmark object. We do not think there is any a priori way to determine which member of this infinite family of ranking algorithms is best.³ Nor is it obvious (at least to us) that there is a best member of this family. Perhaps different members work differentially well on different problems. Empirical investigation is what is needed and we soon turn to a discussion of it. Before we do, however, we look at a different theoretical perspective on the DCB algorithm.

4.2 An IR Perspective on the DCB Algorithm

We have discussed the DCB algorithm from a fundamental, geometric perspective in §4.1. Our purpose in this section is to place the DCB algorithm

³Instead of giving A and B equal weight, their relative importance could be parameterized, e.g., with $A^p \cdot B$. Much else is possible and even sensible. The family of related approaches is indeed infinite.

within the larger context of IR algorithms. The Information Retrieval (IR) literature is replete with such algorithms and techniques for solving—or at least addressing—the document retrieval problem (c.f., [13, 20, 21, 22]). This is a very active area of research and it is beyond the scope of this paper to give a general review of this literature. Instead, we shall discuss a recently-published framework for IR algorithms and show how DCB fits into it.

4.2.1 The IR model

We begin with a synopsis of Kantor’s summary of IR techniques ([15]) and then we follow up by putting DCB into Kantor’s terminology and model. Kantor bases his model of IR techniques upon the following relation:

$$ind(d, c) = \begin{cases} 1 & \text{if concept } c \text{ describes document } d \\ 0 & \text{otherwise} \end{cases} \quad (17)$$

The relation ind defines a table which records which concepts label which documents. Kantor then defines the vector $d(c)$ as the column in this table referencing document d — the list of concepts which describe document d . Similarly, we can define a vector $q(c)$ which lists the concepts of a search query.

If we assume that terms in the corpus are used to identify concepts, then we can substitute the variable t , for term, for c : $ind(d, t)$, $d(t)$, and $q(t)$. What constitutes a term depends on the IR system. A term might be:

- Any word in the corpus
- The above minus stop words
- The above stemmed
- Multi-word phrases (stemmed or not)
- Pre-determined terms from some exogenous dictionary.
- or some other preprocessing or tokenization schemes.

Using the above notation, we can define basic keyterm matching as calculating the match for all documents in the collection where the level of

match between a document and a query is defined by expression (18) below. Document retrieval then presents to the end user the x documents with the highest match (where x is a system parameter).

$$match(d, q) = \sum_{termst} d(t) \cdot q(t) \quad (18)$$

To generalize this model, Kantor adds weighting and concept-to-concept relationships. Weighting allows the model to emphasize some concepts as more important to the match than others. This is modeled mathematically as a matrix, $W(c, c')$, “whose nonzero elements lie only on the diagonal” [15, page 59]. Introduction of concept-to-concept relationships allows the consideration in the match of non-query concepts in the documents that nevertheless are relevant to the query. This can be modeled mathematically as a matrix, $T(c, c')$, “whose entries indicate the degree to which the concepts are related to each other” [15, page 59].

Much IR research has explored different variations of the weighting matrix W . The most common are the *tf* (term frequency) and *idf* (inverse document frequency) measures. Term frequency takes into account the number of times that each term is used in document d , under the assumption that the more often a term is used in a document, the more relevant the concept represented by the term is to the document. Inverse document frequency looks at the entire corpus to see often each term is used (how many documents reference the term). The assumption here is that less frequently referenced terms have more discriminatory power. Other weighting measures have been devised. Empirical study seems to have favored combinations of variations on the *tf* and *idf* measures — often generically referenced as *tf.idf* measures (cf. [15, page 64]).

Kantor discusses different techniques for devising a T matrix. One family of techniques use exogenous, corpus-independent thesauri such as WordNet (c.f. [24]). Unfortunately,

Results to date [with such corpus-independent thesauri] have not been as successful as hoped, and it has been suggested that the specificity of relationships required cannot be found in a universal network of term-term relations. . . . This is akin to the widespread finding in AI that the solution to a specific problem requires development of domain-specific intelligence. [15, page 66]

The alternative to corpus-independent, exogenous sources of the T matrix is to derive it from the corpus itself. Again, Kantor relies on $ind(d, c)$ (although substituting t for c):

$$T(t, t') = \sum_{d \in \text{corpus}} Ind(d, t) \cdot Ind(d, t') \quad (19)$$

Adding the weighting and concept-to-concept extensions to the model in expression (18) and allowing the substitution of terms, t , for concepts, c , we arrive at the full IR model:

$$match(d, q) = \sum_{\text{terms } t, t', t''} d(t) \cdot W(t, t') \cdot T(t', t'') \cdot q(t'') \quad (20)$$

Kantor describes the IR process as one that applies the generalized model in equation (20) at retrieval time. That is, when a user enters a query, q , weighting and thesauri schemes are applied to transform the query into another, q' , that will optimize retrieval — e.g., will return the most relevant documents to a query:

When training data are available (i.e., documents of known relevance), the problem of designing an IR system may be restated as: what choice of WT works best for a specific query q at discriminating the documents whose relevance is known? This may be thought of as finding the best q' for each specific q or as the more general problem of handling all q at once. [15, pages 59–60]

This produces the following retrieval-time processing model:

$$match(d, q) = \sum_{\text{term } t} d(t) \cdot q'(t) \quad (21)$$

where $q \rightarrow q' = WTq$ is found at retrieval time.

4.2.2 DCB and the IR Model

The DCB algorithm can be directly mapped into Kantor's notation and model as follows:

$$K = Ind(d, t) \quad (22)$$

$$L = K \cdot K^T = T(t, t') = \sum_{d \in \text{corpus}} Ind(d, t) \cdot Ind(d, t') \quad (23)$$

$$M = L \cdot K = T(t, t') \cdot Ind(d, t) = \sum_{d \in \text{corpus}} Ind(d, t) \cdot Ind(d, t') \cdot Ind(d, t) \quad (24)$$

From equations (22) through (24) we can see that the basic principles behind the DCB algorithm has a solid foundation in IR techniques. However, there are some crucial distinctions between DCB and other IR techniques.

DCB preprocesses a collection to transform the document vector, for some document j , from $\vec{k}(j)$ (the j th column of K) to $\vec{m}(j)$ (the j th column of M). In Kantor's terminology, DCB uses the following preretrieval-time processing model:

$$match(d, q) = \sum_{\text{term } t} d'(t) \cdot q(t) \quad (25)$$

where $d \rightarrow d' = d \cdot W \cdot T = M_d$ is found before retrieval time,

and $W = I$ (the identity matrix),

$$\text{and } T = L = K \cdot K^T$$

This is in direct contrast to Kantor's retrieval-time process model which transforms $q \rightarrow q'$ at retrieval time (equation 21) instead of transforming $d \rightarrow d'$ before retrieval time (equation 25).

To summarize, the DCB algorithm performs a preprocessing transformation of the d vector using term-term relationships derived from a distance metric of terms in document space. In the IR literature, we can find a few other related techniques. For example: Latent Semantic Indexing (c.f., [10],

[11]) also preprocesses the d vector but uses a technique called singular value decomposition which reduces the concept space into which d is mapped to produce d' ; The Inquiry system's Infinder (c.f., [1], [8]) preprocesses the d vector using a term-term relationships matrix derived from a term-term proximity database. Therefore, our conclusion from §4.1 that empirical investigations are necessary is reinforced. We now turn to such empirical investigations of DCB's merit.

5 Empirical Analysis

The procedural (§§2–3), geometric (§4.1), and IR (§4.2) interpretations of the DCB algorithm raise the question: Is this the right thing to do? We submit that the procedural explanation justifies the DCB algorithm as reasonable, but that the geometric and IR interpretations immediately suggest other possible algorithms. There are other ways to measure closeness between vectors; there are other ways to match documents and queries. These should be explored.⁴ In this we agree with Kantor when he states,

... the proof of a theoretical formulation can be found only in the performance of the algorithms that are realizing that formulation and not in the formulation's self-evidence or rhetorical strength.
[15, page 54]

Landauer, we note, takes an even stronger position:

For the most part, useful theory is impossible, because the behavior of human-computer systems is chaotic or worse, highly complex, dependent on many unpredictable variables, or just too hard to understand. Where it is possible, the use of theory will be constrained and modest, because theories may be imprecise, will cover only limited aspects of behavior, will be applicable only to some parts of some systems, and will not necessarily generalize; as a result, they will yield little advantage over empirical methods.
[19, page 659]

⁴And have been to a large degree [21, 22].

Overstatements can serve good purposes. Certainly for organizational memory the proof is in the pudding. But empirical investigation requires an arduous and lengthy efforts. What follows is—for DCB—we think a promising beginning.

5.1 Comparison with Alternatives via simulation

We have developed and implemented two algorithms that belong to the DCB family. In this section we describe how we have used simulation to compare these two alternatives with DCB. Also, we discuss a fourth approach, LSI (latent semantic indexing), that has appeared in the information retrieval literature.

The simulation methodology that we use to compare the four retrieval algorithms has the following structure. Random, binary association matrices (K matrices) are generated. For every possible (single-term) query, the performances of the four algorithms are compared against one another. In particular, a given query using a given algorithm results in a single ranked list that can be compared against the ranked lists resulting from the other algorithms. The comparison metric we use is the Spearman rank correlation coefficient.⁵

The simulation approach in general has both strengths and weaknesses. On the plus side, it is relatively simple to implement and it focuses on the similarities and differences among the algorithms. On the minus side, it does not address the user’s view, for example, by comparing human rankings to the machine rankings. Also, although the randomly-generated matrices are intended to be unbiasedly representative of actual information collections, there is no guarantee of their similarity to actual information bases.

5.1.1 The Four Algorithms

We have already discussed the DCB algorithm at length, so nothing more needs to be said here about it. The second algorithm is called latent semantic indexing (LSI). LSI is based on singular value decomposition, and is explained

⁵We chose this because it is sensible, popular, well understood, and highly available. It is, however, not the only measure one might wish to use to compare the algorithms, and others should be considered.

fully in the literature (e.g., [10]). The basic idea is to rewrite the K matrix as the product of three matrices:

$$K = T_0 \cdot S_0 \cdot D_0 \quad (26)$$

where:

1. K , as above is m -by- n
2. T_0 is m -by- r and has orthogonal, unit-length columns
3. S_0 is r -by- r , is diagonal, and contains the singular values on the diagonal
4. D_0 is r -by- n and has orthogonal, unit-length columns
5. r is the rank of K

Then, K is approximated by removing the smaller singular values from S_0 . Appropriate rows and columns are eliminated to yield an approximation:

$$\hat{K} = T \cdot S \cdot D \quad (27)$$

where:

1. \hat{K} is m -by- n
2. T is m -by- k and has orthogonal, unit-length columns
3. S is k -by- k , is diagonal, and contains the singular values on the diagonal
4. D is k -by- n and has orthogonal, unit-length columns
5. r is the rank of K
6. k is the selected number of dimensions in the reduced model

Then, the resulting \hat{K} matrix is used essentially as is the M matrix in DCB (look down the row and rank according to entry value) to rank objects according to single index terms.⁶

⁶For details, see the original paper, [10].

Our third algorithm is called SOK1. This algorithm creates an alternative, L^a , to the L matrix of DCB. L^a contains essentially the reciprocals of the Euclidean distances between the vectors of the K matrix, rather than their inner products. SOK1 first computes the Euclidean distance between each pair of intersecting row and column vectors in the K matrix, then adds one to that value and takes the multiplicative inverse. Adding one avoids division by zero. Each element, $l_{i,j}^a$, of this alternative L matrix is a measure of the similarity of the vectors corresponding to the row i and column j of the K matrix. Given this alternative L matrix, the M matrix is then computed using the same inner product computations with the K matrix as in the DCB algorithm.

Our fourth algorithm, SOK2, presents another way to incorporate Euclidean distance for ranked retrieval. SOK2 calculates the L matrix as in the DCB algorithm. Next, an alternative M matrix, M^a , is computed by calculating the Euclidean distance between the vectors (rows) of L and the columns of K , rather than, as in DCB, the scalar (inner) products.

5.1.2 Simulation Results

We varied three parameters in the simulations: the number of rows and columns in the K matrix, the density of ones in the K matrix, and the number of singular values retained in the LSI calculations. Table 1 shows the ten experiments we performed and the associated parameter values.

Exp. No.	Rows	Columns	SVs kept	1s density
1	20	20	2	0.2
2	20	20	5	0.2
3	50	100	5	0.1
4	50	100	5	0.2
5	50	100	10	0.1
6	50	100	10	0.2
7	50	100	5	0.3
8	50	100	5	0.4
9	100	200	10	0.1
10	100	200	10	0.2

Table 1: Simulation Experiment Parameters

The results of the ten experiments, recorded in [12], indicate substantial performance differences among the algorithms. Many of these differences are accentuated under particular circumstances. The worst correlation between two algorithms was a rank correlation coefficient of 0.28 in experiments 3 and 10, in both cases between SOK1 and LSI. The highest correlation between LSI and another algorithm was 0.73 in experiment 8. Typical correlations between LSI and the other three algorithms were in the neighborhood of 0.4. The correlations among the SOKx algorithms and DCB tended to be higher, which is not surprising given that the SOKx approaches are perturbations on the DCB algorithm, rather than an entirely different approach as is LSI. Although these positive correlations are encouraging, given what is known about DCB and LSI, there is a clear need for further study—especially in more realistic retrieval environments—in order to understand better the performance of these algorithms. Our results, however, provide some initial insights.

Besides the overall comparative performance of the four algorithms, a few specific findings are worth comment. The size of the problem, that is the number of indexing terms (rows) and objects (columns), does not seem to have a strong effect. The larger matrices did have lower variances in the resulting rank correlation coefficients. This is not particularly surprising, but it does have implications for determining the number of simulation runs. The number of singular values kept did not have a major effect (c.f., experiments 3 vs. 5 and 4 vs 6). This concords with [10], in which there was a modestly large range over which this system parameter could vary without a substantial effect. The 1s density, however, does at times have a fairly substantial effect.

In sum, these experiments tell us that DCB, in correlating highly with SOK1 and SOK2, which take a similar vector space approach but use Euclidean distance metrics to measure similarity, is plausibly robust. Perhaps the reduced correlations with LSI are due to the fact that LSI discards information. Of course, these conclusions are tentative, even if encouraging. Much more extensive study will be necessary to substantiate them. Of greater urgency, we believe, is testing these algorithms against human judgments of relevance. We shall now describe just such an experiment with the DCB algorithm.

5.2 Experiments with Human Subjects

We have developed several prototype systems that calculate DCB and apply it in various ways to the two aspects of the organizational memory problem. One of these systems, called PIRS (Picture Indexing and Retrieval System), makes use of text associated with pictures in order to index pictures for later retrieval. PIRS uses the DCB algorithm on the associated texts in order to perform ranked retrieval (with associative indexing and retrieval) for users' queries. Operation of the PIRS software is described in [17]. The aim of our human subjects experiment was to test the quality of the rankings done by PIRS/DCB. We chose this application area—picture retrieval based on associated text—for two reasons:

1. Since the subjects see the photos but the ranking is done on the basis of associated text which the subjects do not see, this is a more stressful test of DCB than merely testing the subjects' rankings of texts against DCB's ranking of texts.
2. Subjects can review and judge photos very quickly, much more quickly than they can be expected to read and understand a few pages of ordinary text.

A total of 390 photographs from the Clarence Laughlin archives at The Historic New Orleans Collection were scanned and then put in digital form into PIRS. For each of these photographs, The Historic New Orleans Collection has one or more database records containing text, written by Clarence Laughlin and associated with the picture in question. The Laughlin database was extracted from a database on a Hewlett-Packard minicomputer at The Historic New Orleans Collection, put into a Paradox database, mailed on diskettes to the University of Pennsylvania, and converted to Macintosh files. The resulting textual files for each picture were then processed to extract key words, using a Prolog program. The Prolog program, relying on a publicly available electronic dictionary, removed stop words, adverbs, and verbs from the text associated with the picture. The program retained the nouns and the noun phrases (adjectives followed by nouns) for indexing. Proper names, which were unknown to the dictionary, were retained and treated as nouns. The extracted terms (nouns, noun phrases, proper names) were then used to index each stored photograph, resulting in a K matrix, representing the

presence of each of the various indexing terms in each of the documents, as described above. These indices were then processed in accordance with the DCB algorithm and the results were used to provide the relevance rankings used in the experiment. In particular, the L matrix was computed and stored in the Oracle database. When queries were undertaken, the relevant parts of the M matrix were computed in real time.

In order to arrive at the questions and photographs to be used in the experiment, we performed a number of searches/rankings with PIRS. Rankings for many terms were not fruitful and turned up only a few photographs not widely separated in their rank scores. Other terms produced a richer return. We explored in this fashion until about 10 such queries were found. In particular, although we found additional useful terms, we used queries on the following seven search terms:

- fantastic architecture
- Gothic
- visual poems
- poems of desolation
- mystery
- marine forms
- satire

We then selected the actual pictures to be used for the experiment. In doing so, we sought both a more narrow and a wider ranking distance. In particular, the narrow ranking distance was 5: we choose pictures whose ranks were: 1, 6, and 11. The wider ranking distance was 10: we choose pictures 2, 12, and 22. Insofar as possible, we adhered to this scheme. However, some pictures would not print for us, so we choose pictures nearby in PIRS/DCB rank for the query in question. The information on the actual pictures and their ranks as coded in the given queries is given fully in [12].

In presenting the questions to the subjects, every effort was made to randomize fully. Each subject received three questions, randomly chosen. The labeling of the pictures for each question was determined randomly.

The experiment was given twice to groups of Wharton undergraduates, group 9 and group 10 (referring to the times the experiment was administered).

5.2.1 Human Subjects Results

5.2.1.1 Group 9 In group 9, 32 subjects answered 3 questions for a total of 96 questions answered. All questionnaires were filled out validly. Each of the 96 questions involved ranking by relevance three pictures. Each 3-way ranking can be done in 6 possible ways, only one of which is fully correct in the sense that it duplicates the PIRS/DCB ranking. Further, each 3-way ranking contains 3 implicit pairs of rank judgments: X vs. Y, X vs. Z, and Y vs. Z. Under the null hypothesis, subjects should be “correct” (i.e., in agreement with the PIRS/DCB pairwise rankings) about half the time. Thus, under the null hypothesis, in a given 3-way ranking a subject should expect to get 1.5 of the 3 pairwise rankings correct just by chance.

In group 9, then, there were 32 (subjects) * 3 (questions per subject) * 3 (pairwise rankings per question) = 288 pairwise rankings performed. The number expected to be correct by the null hypothesis is half of that, or 144. A total of 206, or 72%, were in fact correct. This represents an improvement over chance of $72\% - 50\% = 22\%$. The effect is clearly significant statistically. The probability of flipping a fair coin 288 times and getting 206 or more heads is quite low (0.000000000000089396).

Looking at the data in a slightly different way, we note that in a 3-way ranking, there is 1 way to get 3 of the pairwise rankings correct, 2 ways to get 2 of the 3 pairwise rankings correct, 2 ways to get 1 of the 3 pairwise rankings correct, and 1 way to get none of the 3 pairwise rankings correct. Thus, the number of 3s (all 3 pairwise rankings correct) should, under the null hypothesis, equal the number of 0s, and the number 2s should likewise equal the number of 1s.

Here are the actual counts from group 9:

1. 3s: 38
2. 2s: 40
3. 1s: 12

4. 0s: 6

We can think of the 3s and 2s as “good results.” Then, from this data, the probability that a subject agrees with the PIRS rankings on all 3 or on 2 of the 3 pairwise rankings is $(38+40)/(38+40+12+6) = 81\%$, an improvement of 31% over that predicted by the null hypothesis. Again, the effect is clearly significant statistically, since the probability of getting 78 or more heads upon flipping a fair coin 96 times is quite small. In fact, this probability is $2.22347 \cdot 10^{-10}$.

5.2.1.2 Group 10 The analysis, and the results, are essentially the same as for group 9. In group 10, then, there were 51 (subjects) * 3 (questions per subject) * 3 (pairwise rankings per question) = 459 pairwise rankings performed. The number expected to be correct by the null hypothesis is half of that, or 229.5. We found that 346, or 75%, were in fact correct. This represents an improvement over chance of $75\%-50\% = 25\%$. The effect is clearly significant statistically. The probability of flipping a fair coin 459 times and getting 346 or more heads is $7.6985 \cdot 10^{-29}$.

Looking at the data in a slightly different way, we note that in a 3-way ranking, there is 1 way to get 3 of the pairwise rankings correct, 2 ways to get 2 of the 3 pairwise rankings correct, 2 ways to get 1 of the 3 pairwise rankings correct, and 1 way to get none of the 3 pairwise rankings correct. Thus, the number of 3s (all 3 pairwise rankings correct) should, under the null hypothesis, equal the number of 0s, and the number 2s should likewise equal the number of 1s.

Here are the actual counts from group 10:

1. 3s: 70
2. 2s: 59
3. 1s: 18
4. 0s: 6

Again, we can think of the 3s and 2s as “good results.” Then, from this data, the probability that a subject agrees with the PIRS rankings on all 3 or on 2 of the 3 pairwise rankings is $(70+59)/(70+59+18+6) = 84\%$, an

improvement of 34% over that predicted by the null hypothesis. Again, the effect is clearly significant statistically, since the probability of getting 129 or more heads upon flipping a fair coin 153 times is $6.96531 \cdot 10^{-19}$.

5.2.1.3 The combined results The data from the two groups may safely be pooled, yielding the following net counts for 3s, 2s, 1s, and 0s:

1. 3s: $38+69=107$
2. 2s: $40+60=100$
3. 1s: $12+19=31$
4. 0s: $6+5=11$

From this pooled data, the probability that a subject agrees with the PIRS rankings on all 3 or on 2 of the 3 pairwise rankings is $207/249 = 0.83$, and the effect is highly significant statistically. The probability of getting 207 or more heads upon flipping a fair coin 249 times is $1.09781 \cdot 10^{-27}$.

5.3 Comparison of DCB to Inquiry Based Upon Human Subject Results

It is instructive to see how well PIRS/DCB performs when compared to an established IR system. We therefore, compared our PIRS/DCB rankings to those of the Inquiry system. Inquiry has established a good reputation as one of the foremost experimental IR systems. It has consistently performed well in the TREC IR comparisons (c.f., the TREC conference proceedings — e.g., [14]) and it is perhaps the least expensive first-line IR system available to academics.

We took the texts for the same set of photos used in the above experiment and fed them into the Inquiry IR system. Inquiry performed its own preprocessing (e.g., tokenization and stop lists) and ranking. Some of the photos Inquiry did not rank; they were judged (by Inquiry) as non-relevant. We considered ranked photos to be more relevant than unranked photos and, therefore, assigned a rank of ∞ (infinite) to the unranked photos. If two of the three photos were unranked then they were considered to have equal relevance (or to be equally non-relevant) and were treated as ties. We also had

ties on ranked photos; Inquiry occasionally judged two photos to be equal in rank. In comparing Inquiry rankings with subject and PIRS rankings, we analyze ties in two ways. In §5.3.1, we say that if PIRS or a subject ranks photo X over photo Y and Inquiry ranks them as tied, then we count Inquiry as disagreeing with the PIRS or subject ranking. Later, in §5.3.2, we treat Inquiry very charitably and redo the analysis excluding ties altogether.

For example if Inquiry judged photos X, Y and Z as follows:

- X = rank 1
- Y = rank ∞
- Z = rank ∞

and a subject ranked the same photos as:

- X = rank 1
- Y = rank 2
- Z = rank 3

Then the agreement between Inquiry and the subject is 2: They agree that $X \Rightarrow Y$ and $X \Rightarrow Z$.

If another subject ranked the same photos as:

- Y = rank 1
- X = rank 2
- Z = rank 3

Then the agreement between Inquiry and the subject is 1: They agree that $X \Rightarrow Z$.

If another subject ranked the same photos as:

- Y = rank 1
- Z = rank 2
- X = rank 3

Then the agreement is 0.

5.3.1 Inquiry results

Between Inquiry and PIRS there were 9 (photograph sets) * 3 (photos per set) = 27 pairwise rankings to compare. The number expected to be correct by the null hypothesis is half of that, or 13.5. From the tabulations of the data (available in [12]) we find that 19, or 67%, were in fact correct. This represents an improvement over chance of 67%-50% = 17%. The probability of flipping a fair coin 27 times and getting 19 or more heads is 0.026. Therefore, there is a statistically significant agreement between PIRS and Inquiry on these data.

However, 33% disagreement is enough to make us wonder which system's rankings are better when they disagree. The only recourse we have to establish such a judgment is to use the subjects' rankings as the objective answer by which we can score and compare the two systems.

Comparing Inquiry to group 10, 284, or 62%, of the 459 pairwise rankings agree with subjects.⁷ This is statistically significant. The probability of flipping a fair coin 459 times and getting 284 or more heads is $2.05479 \cdot 10^{-07}$.

Inquiry's net counts for 3s, 2s, 1s, and 0s:

1. 3s: 20
2. 2s: 100
3. 1s: 24
4. 0s: 9

Again, we can think of the 3s and 2s as good results. Then, from this data, the probability that a subject agrees with the Inquiry rankings on all 3 or on 2 of the 3 pairwise rankings is $(20+100)/(20+100+24+9) = 78\%$, an improvement of 28% over that predicted by the null hypothesis. Again, the effect is clearly significant statistically, since the probability of getting 120 or more heads upon flipping a fair coin 153 times is $4.13815 \cdot 10^{-13}$.

⁷In making up the group 10 questionnaire packets, certain information from group 9 was lost. In particular, the code orders (e.g., YXZ) and the actual answer orders (e.g., XYZ) were lost due to time and effort limitations. The number of correct answers was recorded carefully (available in [12]), but, unfortunately, the loss of the code orders only allowed us to compare group 10 to Inquiry.

Therefore, Inquiry also performs well with respect to the subjects. However, comparing the Inquiry/Subject results with the PIRS/Subject results, we can see that PIRS has much greater agreement with the subjects: PIRS agreed with the group 10 subjects 75% of the time while Inquiry only agreed 67% of the time. The distinction is more dramatic when looking at the net counts:

	Inquiry	PIRS
3s:	20	70
2s:	100	59
1s:	24	18
0s:	9	6

To see if the difference in performance between Inquiry and PIRS is significant we count how many times PIRS outperformed Inquiry versus the number of times Inquiry outperformed PIRS. We can judge one system to outperform the other for a particular envelope when the first has more agreements with the subjects than the second. The tallies are:

- # envelopes where PIRS = Inquiry: 68
- # envelopes where PIRS > Inquiry: 72
- # envelopes where PIRS < Inquiry: 13

Therefore, PIRS outperformed Inquiry 72 of the 85 times that they did not tie. The null hypothesis suggests that only half the time, or 36, would one system outperform the other when they do not tie. Again, the effect is clearly statistically significant, since the probability of getting 72 or more heads upon flipping a fair coin 85 times is $2.31429 \cdot 10^{-11}$.

5.3.2 Inquiry results excluding ties

Subjects were not given the choice of ranking photos as tied, nor were they allowed to simply not rank photos and thereby declare them as absolutely non-relevant. Therefore, it may be unfair to Inquiry to penalize it for the tied and unranked photos. We repeat here the same analysis as above, but exclude all the data points for which Inquiry did not rank a photo or ranked two photos with equal ranks.

There were 73 envelopes for which Inquiry gave complete rankings. Of the $3 * 73 = 219$ pairwise rankings, Inquiry agreed with subjects on 152 or

69.41% (statistically significant at $4.52858 \cdot 10^{-09}$). This is an improvement of $69\% - 62\% = 7\%$ over the full data set.

Inquery's net counts for 3s, 2s, 1s, and 0s on the non-tied subset of data are:

1. 3s: 20
2. 2s: 41
3. 1s: 10
4. 0s: 2

The probability that a subject agrees with the Inquery rankings on all 3 or on 2 of the 3 pairwise rankings is $(20+41)/(20+41+10+2) = 84\%$ (statistically significant at $2.4011 \cdot 10^{-09}$), an improvement of $84\% - 78\% = 6\%$ over the complete data set.

Therefore, we do see that Inquery was penalized by our earlier treatment of ties. However, when we compare Inquery to PIRS/DCB on this smaller set of data points we still find a statistically significant difference in performance:

- # envelopes where PIRS = Inquery: 38
- # envelopes where PIRS > Inquery: 24
- # envelopes where PIRS < Inquery: 11

Therefore, PIRS outperformed Inquery 24 of the 35 times that they did not tie. The null hypothesis suggests that only half the time, or 17.5, would one system outperform the other when they do not tie. Again, the effect is statistically significant although less so than with the entire data set: the probability of getting 24 or more heads upon flipping a fair coin 35 times is .020.

Therefore, we still see a statistically significant improvement of the PIRS / DCB agreement with subjects over that of Inquery's. It is our contention that it is a deficiency of Inquery that it would fail to rank photos no matter how non-relevant: relevance is not an absolute; it is a relative quality. To say an object has relevance 16 to an index item is meaningless unless compared with another object's relevance to that same index item. No matter how minute

the relevance of two objects to an index item, it may still be of interest to know which one is more non-relevant (or less relevant). Nevertheless, even when accepting this deficiency as a feature we see that PIRS/DCB statistically outperformed Inquiry.

5.4 Discussion of Empirical Analysis

These results are highly encouraging. Although the data merit further analysis (e.g., to compare narrow rankings 1-6-11 with wide rankings 2-12-22), the size and significance of the basic effect are quite impressive. It surely does seem that, for the cases in which PIRS/DCB produced a non-trivial ranking of the pictures, the method studied here produces results that accord well with subjects' independent judgments. Agreement, of course, is far from perfect, but it is hard to avoid the conclusion that, at least for the Laughlin collection and annotations, the computer system is likely to be helpful to those who might be interested in the collection's pictures.

Thus, the PIRS/DCB system is well worth exploring further for these sorts of problems. This said the following three principal issues arise and/or remain that threaten the validity of these results:

1. We relied on the “convenience text” supplied by Clarence Laughlin, describing his photos. It may well be that this text is, for present purposes, of unusually high quality in describing the content—the photos—at hand. Will other artists be as effective in describing their photos? Will text supplied by others, such as catalogers and curators or even the general public in commenting on seen pictures, be equally useful? Will results be as successful on a collection of Congressional questions and answers?
2. Our picture base had only 390 photographs, with associated text. How well will the ranking algorithm work with a much larger scale document base? Scale is known to be the enemy of standard information retrieval (IR) algorithms [6], but the present algorithm, DCB, is not among the standards. Using DCB, an entire collection may be rank ordered on the basis of one term. This rank ordering is typically quite fine-grained. Our initial simulation investigations indicate that such ranking algorithms will be robust under scale transformations (see §5.1, above).

That is, computational cost aside, if the ranking algorithm works well on small collections, it should work well on large collections. This point requires additional mathematical analysis and, especially, additional empirical investigation.

3. The DCB algorithm is but one of a large family of algorithms that can rank objects based on text. How does the DCB algorithm compare in performance with alternative algorithms? Again, our simulations, §5.1, help. Given high correlations with SOK1 and SOK2, it is not likely that these will do tremendously better. Given the excellent performance of DCB in this experiment, it is unlikely that LSI, with its lower correlations and its discarding of information, will do tremendously better. All of this is, of course, very much open to empirical investigation.

Subsequent investigation will be required to resolve these issues. Our principal aim in this experiment has been to produce promising results that by themselves mandate such subsequent investigations. We believe that aim has been richly achieved.

6 Conclusion

Prima facie, the DCB algorithm has much to recommend it. Very importantly for the organizational memory problem, DCB ranks what it retrieves and uses a form of associative retrieval. That DCB can be used both for document retrieval (for which it was originally conceived) and for other problems of organizational memory (as we have shown above) is also a highly attractive feature. Further, we have shown in our implementations and subsequent testing that DCB is computationally practicable, at least for collections with thousands of objects.⁸ With partitioning of the objects, of course, much larger collections could be treated.

At bottom, however, the proof is in the pudding. How well does the system work? We have made it plain that the results reported here are initial. They are promising, but hardly definitive. Only scores of further experiments, on a wide variety of databases and problems, could produce

⁸Our most recent implementation, KSSIR, routinely handles collections of 15,000 items, running on a 166 MHz Pentium processor under Windows NT.

definitive results. Such is the nature of experimental work. But with all these caveats to hand, it is worth recounting the positive and substantial achievements of the work reported in this paper. Principally, they are as follows.

1. We have rescued the DCB algorithm from an obscure working paper [4], and in doing so have reformulated the algorithm with clarity and mathematical precision. See §§2, 3, and 4.
2. We have shown how two organizational memory problems—document retrieval and resource location—can receive a common conceptual and algorithmic treatment. We illustrated this with the DCB algorithm, but the point applies generally, since—as we have also shown—DCB is but one member of a family of similar algorithms. See §§2, 3.

Our discussion has focused on three application areas: picture retrieval from the Clarence Laughlin archives in The Historic New Orleans Collection, resource location, and finding documents pertaining to Congressional questions for the Coast Guard. We hope it is obvious that these are special cases and that many other applications and application areas are plausible.

3. We have shown how DCB, in our formulation, fits within the general IR model for information retrieval. Also, we have noted that we have been unable to find a description, let alone an investigation, of any closely similar algorithm in this literature. See §4. Thus, the rescuing of DCB presents the literature with something effectively new, different, and plausible.
4. We have implemented and tested DCB in a number of contexts, including resource location for the Coast Guard, described in [12, 16], finding documents pertaining to Congressional questions, and picture retrieval for The Historic New Orleans Collection, see [12, 17]. Throughout, we have found the algorithm to produce results with high face validity.
5. Sensibly, DCB locates documents in word space and measures relevance to a query via a measure of distance in that space. DCB's measure is noneuclidean. Is it correct, is it optimal? We see no theoretical way to decide these questions. Our simulation results, however, are cause for

comfort: DCB correlates well with the euclidean measures of distance we tried. There is certainly need for further investigation, but the indication so far is that DCB is fairly robust.

6. In order to test empirically the performance of the DCB algorithm we have designed and introduced a very efficient experimental approach, described in §5.2. The literature on experimental testing of ranking algorithms in IR has generally relied upon relevance judgments made by human experts on a fairly exhaustive set of documents. This is expensive and inefficient. The approach we have taken relies on a sample of the rankings. Further, the resulting data can be analyzed—as we have shown—with statistics free of distributional assumptions (i.e. they are non-parametric).

Important also is the fact that Internet search engines are now delivering ranked retrieval of queries against very large document databases, and doing so with amazing frequency. The algorithms actually used by these engines are mostly kept as trade secrets. The performance of these algorithms, measured as quality of ranking, is generally agreed to be rather poor. The experimental design we have described here offers the prospect of comparatively easy and inexpensive empirical evaluation of the Internet search engines.

7. We have conducted an experiment to test empirically the adequacy of the DCB algorithm and have found that for the problem we gave it (picture retrieval based on associated text) (a) DCB produced a large positive effect (while under the null hypothesis we expect a 50% agreement with paired rank judgments, DCB produced an 83% agreement), (b) this effect was highly significant statistically.
8. We further compared our experimental data with results produced by a leading Information Retrieval system, Inquiry. As in the case of DCB, we found that (a) Inquiry produced a large positive effect, (b) this effect was highly significant statistically. We also found that DCB performed somewhat better than Inquiry on the problem tested.

In short, these investigations are only the beginning of a deep understanding of how computational approaches may alleviate the organizational

memory problem. Extensive empirical work remains to be done. Nonetheless, the steps taken here have produced very promising results, and they leave our community with much to build upon.

References

- [1] Applied Computing Systems Institute of Massachusetts. *Inquiry Manual*, 1996.
- [2] M. J. Bates. Subject access in online catalogs: A design model. *JASIS*, 37:357–376, 1986.
- [3] Nicholas J. Belkin and W. Bruce Croft. Retrieval techniques. In Martha E. Williams, editor, *Annual Review of Information Science and Technology (ARIST)*, volume 22, pages 109–145. American Society for Information Science, 1987.
- [4] David C. Blair. An extensional semantic analysis of document indexing. University of California at Berkeley working paper, December 1974. 27 pages.
- [5] David C. Blair. *Language and Representation in Information Retrieval*. Elsevier Science Publishers, 1990.
- [6] David C. Blair. The challenge of commercial document retrieval: Major issues, and a framework based on search exhaustivity and document collection size. *Information Processing and Management*, in press.
- [7] David C. Blair and M. E. Maron. An evaluation of retrieval effectiveness for a full-text document retrieval system. *Communications of the ACM*, 28(3 (MAR)):289–299, 1985.
- [8] John Broglio, James P. Callan, and W. Bruce Croft. Inquiry system overview.
- [9] Michael Buckland. What is a document? *Journal of the American Society for Information Science*, 48(9):804–9, 1997.

- [10] Scott Deerwester, Susan T. Dumais, George W. Furnas, Ghomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.
- [11] Susan T. Dumais. Latent semantic indexing and TREC-2. In Donna Harman, editor, *The Second Text REtrieval Conference (TREC-2); 1993 August 31–September 2; Gaithersburg, MD*, pages 105–116, Gaithersburg, MD, 1994. National Institute of Standards and Technology, NIST Special Publication 500-215. CODEN: NSPUE2; NTIS: PB94-178407.
- [12] Garrett O. Dworman, Abeer Y. Hoque, Steven O. Kimbrough, Stephen E. Kirk, and Jim R. Oliver. Report on an experiment on picture retrieval using the dcb algorithm, the pirs software and pictures from the clarence laughlin archives at the historic new orleans collection. technical report, University of Pennsylvania, Department of Operations and Information Management, 3620 Locust Walk, Philadelphia, PA 19104-6366, 1997. PIRS Exp. 7/21/97.
- [13] William B. Frakes and Ricardo Baeza-Yates, editors. *Information Retrieval: Data Structures & Algorithms*. Prentice Hall, Englewood Cliffs, NJ 07632, 1992. ISBN: 0-13-463837-9.
- [14] D. K. Harman, editor. *Overview of the Third Text REtrieval Conference (TREC-3)*. U.S. Department of Commerce, Technology Administration, National Institute of Standards and Technology, Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402, 1995. NIST Special Publication 500-225.
- [15] Paul B. Kantor. Information retrieval techniques. In Martha E. Williams, editor, *Annual Review of Information Science and Technology (ARIST)*, volume 29, pages 53–90. American Society for Information Science, 1994.
- [16] Steven O. Kimbrough and Jim R. Oliver. On relevance and two aspects of the corporate memory problem. In Prabuddha De and Clarson Woo, editors, *Proceedings of the Fourth Annual Workshop on Information Technologies and Systems*, pages 302–311. [na], 1994. File: Oliver-sok-WITS94-r3.

- [17] Stephen E. Kirk and Steven O. Kimbrough. PIRS user's manual. Technical report, available from Steven O. Kimbrough, University of Pennsylvania, 3620 Locust Walk, Philadelphia, PA 19104-6366, 24 February 1995. File: PIRS Manual 12/28/94, 2/24/95.
- [18] Robert Korfhage, Edie Rasmussen, and Peter Willett, editors. *Proceedings of 16th Annual ACM/SIGIR International conference on Research and Development in Information Retrieval*. ACM, 1993. ISBN: 0-89791-605-0.
- [19] Thomas K. Landauer. Let's get real: A position paper on the role of cognitive psychology in the design of humanly useful and usable systems. In Ronald M. Baecker, Jonathan Grudin, William A.S. Buxton, and Saul Greenberg, editors, *Readings in Human-Computer Interaction: Towards the Year 2000*, pages 659–665. Morgan Kauffman Publishers, San Francisco, second edition, 1995.
- [20] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979. See also: <http://www.dcs.glasgow.ac.uk/Keith/>.
- [21] Gerard Salton. *Automatic Text Processing*. Addison-Wesley Publishing Company, Reading, MA, 1988.
- [22] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company, 1983.
- [23] Eric W. Stein and Vladimir Zwass. Actualizing organizational memory with information systems. *Information Systems Research*, 6(2):85–117, 1995.
- [24] Ellen M. Voorhees. Using wordnet to disambiguate word senses for text retrieval. In *See reference [18]*, pages 171–180, 1993.