A NETWORK ORGANIZATION USED FOR DOCUMENT RETRIEVAL

W.B. Croft, R. Wolf, R. Thompson

Computer and Information Science Department
University of Massachusetts, Amherst, MA 01003

COINS Tech Report 83-05

Abstract

A network organization for implementing a document retrieval system is proposed. This organization has significant advantages in terms of the range of searches that can be used when compared to either inverted or clustered file organizations. Algorithms for generating and maintaining the network are described together with experiments designed to test their efficiency and effectiveness.

1. Introduction

The main types of file organization that have been proposed for the implementation of document retrieval systems are inverted files and clustered files [SALT82]. Although other organizations have been mentioned which combine aspects of both of these [CROF80], they are generally regarded as strict alternatives with no middle ground. The recent success of experimental and theoretical work with probabilistic models [RIJS79, SPAR80, CROF82] and the continuing popularity of Boolean searches have led to a preference for the inverted file which, combined with a serial file of the documents, can easily implement these strategies. However there is evidence that searches based on clustered files can be an effective alternative strategy for systems using the probabilistic model [CROF79,80], and that efficiency benefits can

be obtained by physically grouping documents that are likely to be retrieved by the same queries.

In this paper, we investigate a network organization which combines the advantages of inverted and clustered files. This organization will be used as part of an adaptive document retrieval system that will be able to choose from a variety of search strategies based on factors such as a particular user's needs and the history of the search so far [CROF81]. A network organization also provides a means of implementing browsing strategies that would be impractical in the inverted or clustered file environments.

The next section of the paper describes the network organization and compares it to inverted and clustered files. The crucial questions about a network representation concern the efficiency of its generation, maintenance and use for searching. These issues are discussed in section 3. The main topics mentioned there are efficient methods of generating nearest neighbours for documents and terms in the network and methods for physically grouping these network nodes. Section 4 discusses updating strategies.

The final part of the paper presents the results of preliminary experiments with the network organization. The goal of these experiments is to demonstrate that a network can be generated efficiently and that the physical grouping of documents and terms increases the search efficiency.

2. The Network

The proposed organization consists of a network of documents and terms (Figure 1). The nodes in the network represent terms or documents and the links represent associations between them. A weight on each link is used to indicate the strength of the association. The links and the weights can be classified as follows;

Link Type	Purpose of weight
Document-Term	Indicates how important a term is in the representation of a document.
Term-Term	Indicates how closely two terms are related.
Document-Document	Indicates the similarity of the content of the documents.

Other types of links, such as document-author and document-citation, could be included in this network in order to extend the range of search strategies available. However, in this paper, we shall concentrate on the representation of document content using index terms and assume that other bibliographic information is available but not used.

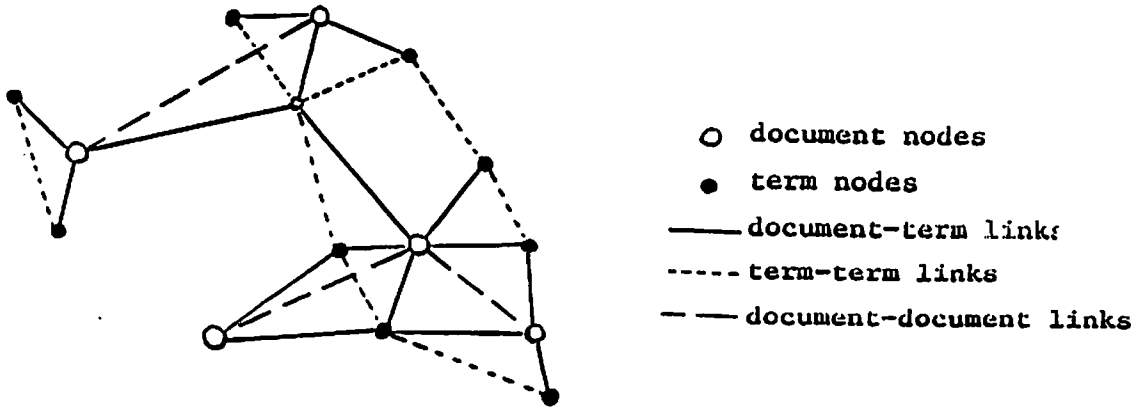

Figure 1.

A network organization.

The document-term links are the links that will be used most often by the search strategies. Since the links are bi-directional, they can be thought of as having two forms; document-term and term-document. The document-term links identify the terms that represent the content of a document. This information is equivalent to that contained in a conventional serial file of documents. The term-document links identify the documents that are described by a particular term. This is equivalent to an inverted file of the documents. Taken together, these parts of the network can be used to implement many of the probabilistic strategies mentioned in the literature. The particular form of the document-term link weight used here is the within-document frequency of the term. It has recently been shown that this weight can be used very effectively in probabilistic strategies [CROF82].

The term-term links contain information about the similarity

of terms which can either be derived manually (using a thesaurus) or statistically (using a minimum spanning tree [RIJS79]). In either case, this information can be used effectively by search strategies.

Document-document links are used to provide the information for a cluster-based search strategy [CROF80]. Many previous experiments have been done with a single-link cluster hierarchy of documents that can also be represented by a minimum spanning tree (MST). The generation of the MST (for both documents and terms) is essentially an O(n**2) process although it can be made reasonably efficient [HARP80]. The main drawback with the MST is that it is expensive to maintain when new documents are included. A more efficient approach results from the observation that experimental evidence indicates that only the strongest document-document similarities are useful for retrieval [SPAR77,CROF80]. Many documents are only weakly connected to each other so their document-document links are not significant and need not be represented. In terms of the single-link hierarchy, this means that only the lowest level (smallest) clusters are required. Therefore, rather than generating the MST in order to construct document-document links, each document need only be connected to its nearest neighbours (defined in terms of a similarity measure). A similar result holds for the construction of term-term links. In both cases, the network is restricted to contain links for the single nearest neighbour only (or the set of equal nearest neighbours).

The document-document (or term-term) links formed by the nearest neighbour process can be thought of as star clusters. A

star is a cluster in which every member of the cluster is a member by virtue of its relation to a distinguished central member of the cluster [SPAR77]. Each document in the network representation serves as the central member of a star cluster formed by connecting that document node to its nearest neighbour document nodes (Figure 2).

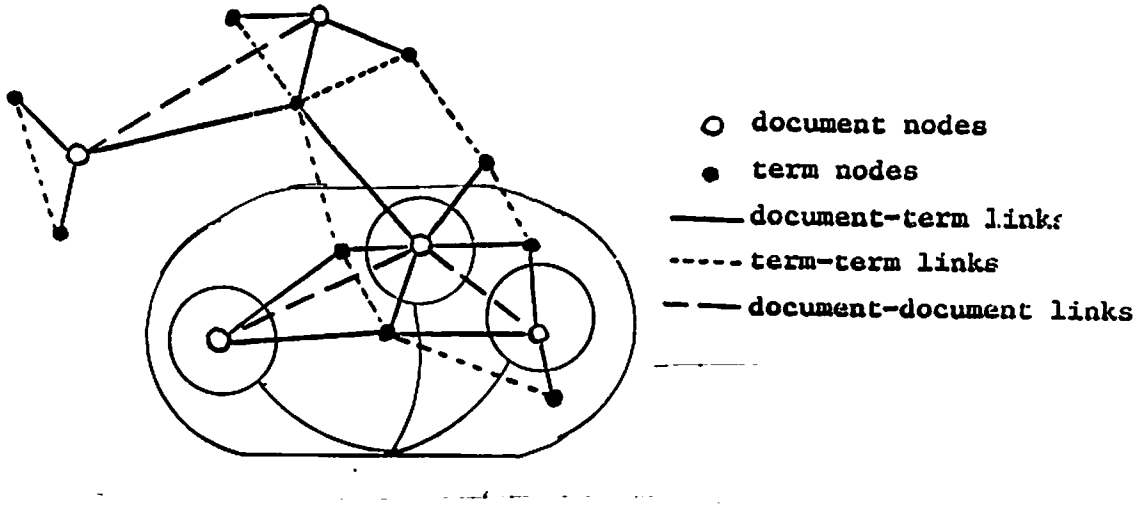

Figure 2.

A star cluster within the network representation

The star clusters formed in the network organization, then, are small overlapping clusters of highly related documents.

In general, the formation of star clusters is highly order dependent in that a different clustering is constructed depending upon which nodes are regarded as central nodes. Since we generate a separate star cluster using each node as a central node, the generation of star clusters is order independent in the network organization.

Cluster searching uses the document-document links from the star clusters. In general, this type ofstrategy requires cluster

representatives for comparison to the queries. However, for maximum flexibility and storage savings, the cluster representatives are not stored in the network as separate entities, but are generated dynamically in the following manner. First, the documents that contain query terms are located using the document-term links (term-document form) of the network. The document-document links are then followed from these documents to form cluster representatives.

As well as the probabilistic and cluster-based searches, the network organization allows the user to follow any links in the network while searching for relevant documents. A special retrieval strategy, called browsing, could be based on this ability. A browsing strategy will be particularly useful when other retrieval strategies have failed. In some related work, Oddy proposed a simple network organization which was used to implement a system entirely based on a browsing strategy [ODDY77].

The main advantage, therefore, to a network organization when compared to an inverted, serial or clustered file is the wide variety of search strategies that can be implemented within this framework. The efficiency of the network organization is the major subject of this paper. The efficiency measures we are concerned with are the storage overhead and the amount of time required to generate, maintain and search the network. These topics are discussed in the next sections.

One interesting feature of the network organization is that the structure of the network and many of the operations required (retrieval, insertion, deletion) are readily represented in the

relational data model [DATE81]. The implementation of a document retrieval system using a database system has been the subject of much discussion [MACL81,PORT82], but it will not be pursued further here.

3. Generating the network

3.1 Finding nearest neighbours

Various methods for efficiently finding nearest neighbours for documents and terms have been proposed [SMEA81,MURT82]. These methods use the following basic algorithm;
To find the nearest neighbour for document (term, query) D, calculate similarity values between D and documents having at least one term in common. The document-document pairs for the similarity calculations are found by first finding the set of terms that describe D and then using the inverted lists associated with those terms. Documents which have been seen in a previous inverted list are ignored.

Smeaton and Van Rijsbergen modified this procedure to include an upperbound (U1) on the maximum similarity value that will be found if more inverted lists are processed. This permits the similarity calculations to stop when U1 falls below the highest similarity value already calculated. If the inverted lists were organized in order of frequency of use of the associated terms, this method will also avoid using the longest lists.

Murtagh used an upperbound (U2) which would be calculated

for each candidate document on the inverted lists in order to determine whether the actual similarity calculation should be made. The use of U2 avoids many calculations of the number of co-occurring terms between two documents which is the most expensive part of the similarity calculation.

The method used in the experiments reported here is a combination of these two approaches with some modifications which are designed specifically for the task of finding nearest neighbours for all documents in a collection. Upperbound U1 is used, as described previously, to determine if more inverted lists should be processed. Upperbound U2 is used to determine if documents within these term lists should be considered. A minimum threshold T is placed on the similarity value. Upperbounds U1 and U2 are compared to the maximum of T and the greatest similarity value found so far. No similarity value less than T is considered for a nearest neighbour. The use of this threshold tends to produce documents with no nearest neighbours. This is appropriate in the context of removing links which are not significant.

While a nearest neighbour for a particular document is being calculated, a record is kept, for every other document, of the highest similarity value that the document was involved in. For example, if while finding the nearest neighbour for document 100, a similarity value of 0.53 was calculated for the document pair 100,123, this similarity value is compared to the previous highest value seen for document 123 and stored if it is higher. This means that when the nearest neighbour for document N is to be found, the calculations start with the similarity value found

when calculating nearest neighbours for documents 1 through N-1. This procedure also ensures that document pairs containing documents 1 through N-1 need not be considered. The efficiency of this procedure for finding nearest neighbours is studied in more detail in section 5.

3.2 Physical Grouping

Because of the declining cost of disk storage, large collections of documents will continue for some time to be stored on disk, a cellular memory device. Grouping of closely related items in the same cell to reduce inter-cellular references can significantly reduce the time required to respond to user queries. Grouping is similar in concept to the use of document (and term) clustering to organize a document collection for efficient retrieval. Although grouping shares some of the same attributes and techniques of document clustering, the two are not identical. In particular, the groups into which a document collection is partitioned need not correspond to the clusters produced by document clustering.

Unlike document clustering, where a document may belong to more than one cluster, a document may belong to only one group, since it is stored in one location in physical memory. A method of grouping is required that constructs non-overlapping groups. In addition, in document clustering the size of a cluster may vary whereas grouping attempts to produce groups that are of fixed size, the size of a secondary storage cell (disk track or page), since the entire cell will be transfered into primary

memory when a reference to any member of that cell occurs. A method of grouping is required, then, that constructs fixed size groups.

Groups are constructed using strings, a clustering technique similar to stars. Strings are clusters in which strongly connected objects are linked up to the natural cutoff of a loop or maximum length [SPAR77]. A string is a group of objects that are related transitively as nearest neighbours (Figure 3).

Once an object has been placed in a string, it is not eligible for membership in another string. If in the course of constructing a string it leads into another string, the two strings are joined together as one. If the size of a string becomes larger than the maximum size allowed, the string is split into two.

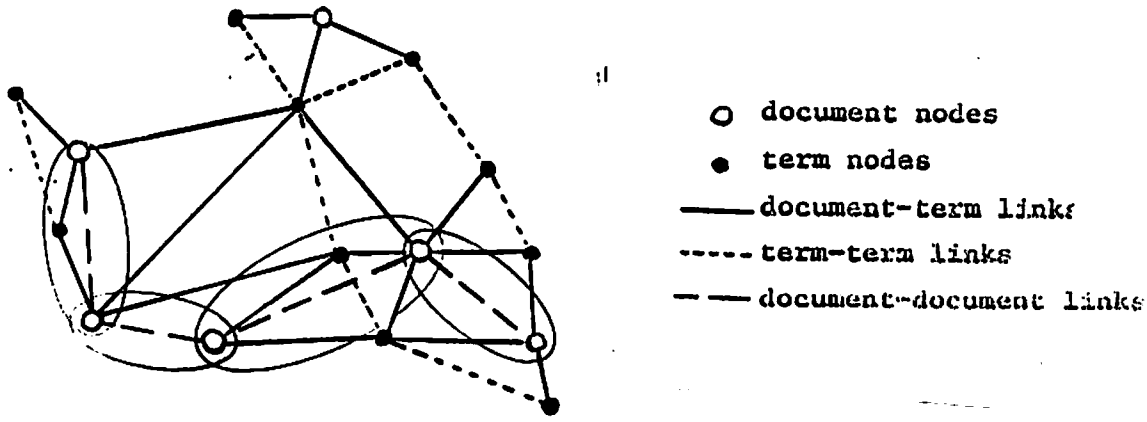

Figure 3.

A string within the network representation

The use of strings for grouping suffers from the problem that the construction of strings is dependent upon the order in which

objects are inserted into the network. Another problem is that it is difficult to generate fixed size groups since many strings will complete before the specified size is reached. To overcome this, small strings will be stored together in cells until the desired size is obtained. Some space will be left in each cell for updating. The properties of strings used for grouping will be studied in section 5.

3.3 Implementation

The experimental system is implemented on a Vax 11/780 in Pascal. As Pascal is not a systems progamming language such as C or BCPL, the virtual memory portion of the system is simulated rather than integrated into the memory management of the host hardware. We used the Vax page size in our simulation as a standard which would represent a typically sized page in a machine.

The representation of the network is accomplished by six arrays storing the document nodes, term nodes, and the four varieties of links. The varieties are Doc-Doc, Doc-Term, Term-Doc, and Term-Term links. A link consists of a node number, which corresponds to an index into one of the node arrays, and a weight. The nodes contain the minimal amount of information to define a term or document, which consists of the pointers to the links , the number of the links, and some other details required for the operation of the grouping algorithm. Having only this information allows us to keep the nodes small so we can place a number (8) of them on a page. The primary disadvantage of

separating the links from the nodes is that it may generate a fault as we follow the links from node to node. The storage overhead required for this organization is very similar to that required by a combined serial/inverted file which also contains document-document and term-term links. The number of these links compared to the size of the collection will be studied in section 5.

Portions of the contents of a node, information such as document-title, author, and term-name, are stored in a separate area. Access to this information will generate a page fault. If this information is needed often, it could be stored with the node. If the network representation will only be used with an experimental document retrieval system, this information can be omitted.

The nodes (documents and terms) are grouped and the lists of links are also grouped, so that lists of links of similar nodes are stored contiguously. Using this technique it is possible to fit many nodes on one page, but few lists of links on one page, since each list of links will take up a good portion of a page. If some of the contents of a node, such as the document-title, author, etc., are stored separately from the rest of the node, they should also be grouped.

Indexing into the network will be performed by a simple table lookup with the document (or term) number being the index. Although hashing was the original implementation, the simple lookup is quicker. The numbering of the documents is not sparse so little space is wasted on empty positions in the table. The primary operations in a retrieval system are inquiry and document

insertion so no holes should develop. It will usually cost one page fault to locate a document or term node, given the document or term number.

4. Updating the network

One of the most important operations on the network is the inclusion of new documents and terms. If the network is to be used in a real system existing in a dynamic environment, updating must be efficient and must not degrade the effectiveness of the structure. Insertion of a document into the network establishes links to the related documents and terms, creates additional term nodes if new terms are introduced, and modifies any term-term or document-document links that are affected.

Insertion is performed as follows (Figure 4). First, space is allocated for a new document node and the document-term forms of the links are constructed. The document-term information is taken from the document representative provided as input to the insertion operation. Next the term-document form of the link is constructed for every term of the document representative. If a new term is used, space is allocated for that term in the network. The nearest neighbour(s) of the new document are then determined to construct the document-document links. Insertion of a new document may require modification of existing document-document links, in which case the document-document links of the nearest neighbours of the new document are reconstructed. Insertion of a new document may also modify existing term-term links, so for each term of the new document,

the term-term links are reconstructed. The reconstruction of links is not propagated further in the network than the nearest neighbour documents or terms, for reasons of efficiency. The nearest neighbours of the new document and terms should be used to guide the placement of the new node in the physical grouping of the network. That is, a new document node should be stored as close as possible to its nearest neighbour.

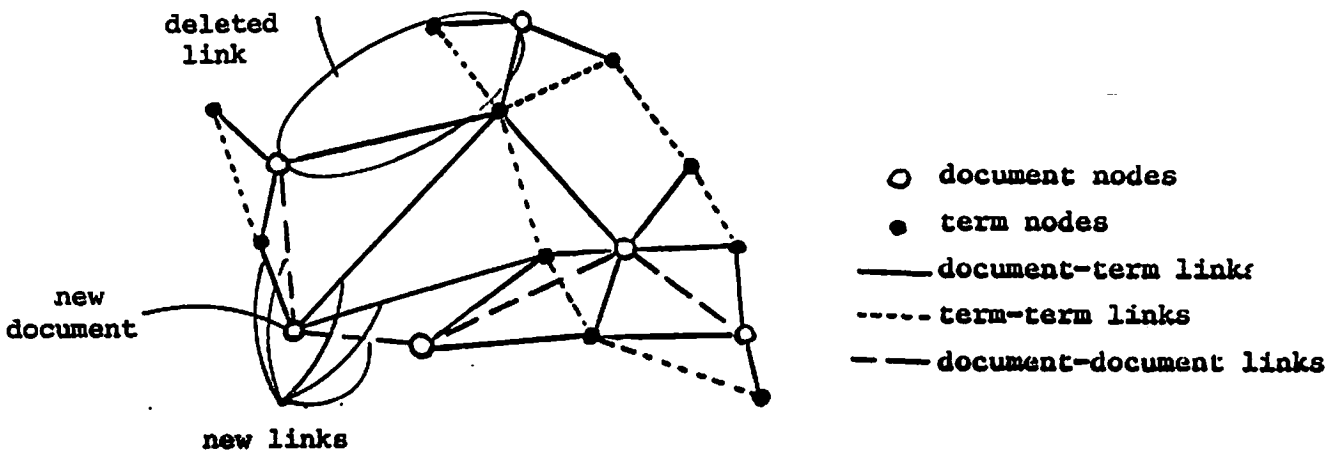

Figure 4.

Insertion of a new document in the network

The insertion algorithm may result in a network configuration that is not optimal, in the sense that it is not the network representation that would be instantiated if the network were constructed all at once, as during initiation. When retrieval operations become inaccurate and inefficient due to insertion of new documents, the network may be reorganized. The network representation should not depend, however, upon costly and untimely reorganizations of the data base to cope with the addition of new documents.

5. Experiments

The experiments reported here deal with two aspects of the network; generating nearest neighbours, and the grouping algorithm. All experiments were carried out with the Cranfield collection of 1400 documents [SPAR77]. To avoid any inherent structure in the test set, the documents were "shuffled" and renumbered.

The first experiment recorded the number of similarity calculations required to form nearest neighbours for the documents and terms and the effectiveness of U1, U2 and the "previously seen" modification in reducing this number.

The experiments to test the effectiveness of the grouping were run in two phases. First, a control was run. The control performed a null grouping which placed the documents, terms, and links into memory in the same order in which they were read. For each of the 225 queries the documents were ranked in the order of the number of matching terms. During this, the page accesses and faults which occured in the retrieval processing were recorded.

In the second phase the documents and terms were grouped according to the previously described algorithm. This second phase also tested the effect of limiting the maximum size of the group. The two possibilites were unlimited and 16. Again the queries were processed and the accesses and faults counted. In the second phase the documents and terms were grouped according to the previously described algorithm and processed against the 225 queries.

5.1 Experimental Results

The nearest neighbour calculations are summarized in Table 1. This table shows the total number of similarity calculations performed for the entire collection along with the average number per document or term. It also gives the percentage of calculations that each technique saves when compared with the total possible number of calculations.

	Total Sim Calcs	Average Sim Calcs	Percent Saved By U1	Percent Saved By U2	Percent Saved By Pre Calc
Docs	292693	209	20.9	12.8	34.1
Terms	66738	25	1.5	34.1	47.8

	Total Possible Similarity Calculations	Avg/Item
Docs	1305079	946
Terms	412063	153

Table 1.

Nearest Neighbor Calculations

By using the bounds, U1 and U2, and by precomputing (or "remembering") nearest neighbour calculations we achieved significant savings. Only 32.2% of the total possible calculations were required for the documents and only 16.6% for the terms. Precomputing the similarities obtained the greatest

reduction, especially in the case of the terms.

The great efficiency in the term calculation is a result of processing the terms in the order of the least frequent to the most frequent. Very few calculations were performed on the most frequently occuring terms in the collection. It must be noted that term pairs which occured together in only one document were not included.

The savings realized by these methods suggest that it is reasonable to perform nearest neighbour calculations on large collections of documents.

Tables 2, 3, and 4 show the results of the grouping experiments. Table 2 gives the average size of the "strings" produced in two cases; the first case is when no limit is placed on the group size, the second is when the size is limited to 16. The limited size strings were designed to be more convenient for storing on pages.

	Documents	Terms
Limited (16)	3.79	1.14
Unlimited	4.98	1.19

Table 2.

Grouping Results

(Items / Group)

Table 3 shows the number of pages on which the top 20 retrieved documents were found. The results are shown as the

total number of pages for all 225 queries, the average number of pages per query and the percentage improvement compared with the ungrouped data. The grouped results are broken down into 4 categories. The categories come from varying the maximum size of a string and from whether a new string is stored on a new page or packed together in a page with other strings.

	Total Pages	Pages/Query	Percent Improvement
Not Grouped	4023	18.3	
Grouped			
Limit 16	3405	15.1	15.4
Limit 16 new pg	3322	14.8	17.4
Unlimited	3328	14.8	17.2
Unlimited new pg	3311	14.8	17.6

Table 3.

Page Results

Table 4 contains the page fault results. The total number of references to pages is constant for all grouping variations. The "Not Grouped" statistics represent the control experiment. The grouped results are divided into the same 4 categories as before.

	Term	Term/Doc	Document	Percent Improvement For Documents Retrieved
References	1738	288820	4500	
Faults				
Not Grouped	1373	5570	3483	
Grouped				
Limited Size				
No New Page	1304	5570	2909	16.6
New Page	1314	5548	2926	16.1
Umlimited Size				
No New Page	1350	5521	2861	17.7
New Page	1346	5567	2905	16.7

Table 4.

Page Fault Results

The results in Tables 2, 3, and 4 are summarized in the following discussion.

The grouping results show that only a small effect on the average group size results from groups decreasing the maximum allowable size to 16. In both cases the size was less than the number of documents which fit on one page in memory. This indicates that, on t e average, a new document can be placed on the same page as its nearest neighbour.

The results in Table 3 show that grouping gave a moderate reduction in the number of pages that contained the top 20 retrieved documents. the variations in the grouping algorithm

produced no significant differences in the results. strings produced as well as whether a new page was allocated for a new string produced no significant difference in the grouped results.

The page fault data obtained during query processing also demonstrates only a moderate gain in efficiency for the document page faults. An approximately 16% improvement was found for the case with a limit of 16 on the group size. A slightly larger improvement, 17.7%, was found for the unlimited group size case. In both cases starting a new page for a new string gave slightly worse results. No significant gain was obtained for the term or term-document page faulting. However, the improvement in document faults is only a small gain in the overall processing. If the number of accesses are counted, then 98% of the work for this strategy occurs before the top 20 documents are retrieved. The moderate 17.7% efficiency gain in the final document retrieval is less than a 1% gain for all the query processing.

6. Conclusion

Conventional document retrieval systems use a representation in the form of a file organization designed to operate efficiently for a particular search strategy. An organization which decouples the representation from the search strategy allows a much wider range of searches to be used. This paper describes the design and implementation of a network representation of documents and terms in a document retrieval system.

The experimental results indicate that the network can be

generated efficiently, but no significant efficiency benefits were obtained by grouping documents and terms. The latter result suggests that a database implementation of the network should be pursued further.

References

[CROF79] Croft, W.B.; Harper, D.J. "Using probabilistic models of document retrieval without relevance information" Journal of Documentation, 35: 285-295; 1979.

[CROF80] Croft, W.B. "A model of cluster searching based on classification" Information Systems, 5: 189-195; 1980.

[CROF81] Croft, W.B. "Incorporating different search models into one document retrieval system" Proceedings of the 4th ACM SIGIR Conference, SIGIR Forum, 16: 40-45; 1981.

[CROF82] Croft, W.B. "Experiments with representation in a document retrieval system" Information Technology, (to appear).

[DATE81] Date, C.J. An introduction to database systems. Addison-Wesley, Reading, MA. (1981).

[HARP80] Harper, D.J. "Relevance feedback in document retrieval systems: An evaluation of probabilistic strategies" Ph.D. Thesis, University of Cambridge (1980).

[MACL81] Macleod, I.A. "A data base management system for document retrieval applications" Information Systems, 6: 131-137; 1981.

[MURT82] Murtagh, F. "A very fast, exact nearest neighbour algorithm for use in information retrieval" Information Technology, 1: 275-284; 1982.

[ODDY77] Oddy, R.N. "Information retrieval through man-machine dialogue" Journal of Documentation, 33: 1-14; 1977.

[PORT82] Porter, M.F. "Implementing a probabilistic information retrieval system" Information Technology, 1: 131-156; 1982.

[RIJS79] Van Rijsbergen, C.J. Information Retrieval. 2nd edition. Butterworths, London (1979).

[SALT82] Salton, G.; McGill, M. Introduction to modern information retrieval. McGraw-Hill, New York (1982).

[SMEA81] Smeaton, A.F.; Van Rijsbergen, C.J. "The nearest neighbour problem in information retrieval" Proceedings of the 4th ACM SIGIR Conference, SIGIR Forum, 16: 83-87; 1981.

[SPAR77] Sparck Jones, K; Bates, R.G. "Research on automatic indexing 1974-1976" British Library Research and Development Report 5464, Computer Laboratory, University of Cambridge (1977).

[SPAR80] Sparck Jones; Webster, C.A. "Research on relevance weighting 1976-1979" British Library Research and Development Report 5553, Computer Laboratory, University of Cambridge (1980).