TEXT FILE INVERSION: AN EVALUATION

R. M. Bird

J. B. Newsbaum

J. L. Trefftzs
Operating Systems, Inc.

ABSTRACT

This paper compares inversion of text files with inversion of more structured records. The unique characteristics of textual data which restrict the utility of inversion are identified and discussed. Inversion is shown to be useful only for small, static data bases, and when full text search is not required.

1. TEXT SEARCH RATIONALE

1.1 Uses of Text Files

A variety of institutions are in the information business, and as a result, accumulate large files of textual data covering a wide range of subjects. Such organizations (such as newspapers, specialized libraries, and intelligence agencies) need to be able to search their text files rapidly in order to provide answers to an unrestricted range of potential questions, and to provide documentary backup to the answers which they provide to their clients.

Modern technology has made it possible to store large textual data bases in machine-readable form: indeed, within some organizations the majority of textual information is presented in electrical form at some point in its life cycle, and is thus amenable to entry into an electronic storage and retrieval system.

Conventional techniques provide insufficient search and retrieval capabilities against such a textual data base. Many users of such data bases cannot predict the kinds of questions they will be asking: pre-need document indexing, no matter how good, is an insufficient response to their information requirements.

As a result, such institutions perceive a need for large-scale unrestricted search systems which allow access to the full, unaltered texts of the documents stored therein.

1.2 Query Capabilities

Query capabilities fall into the four general categories of:

- Term Matching
- Numeric Ranging
- Term Processing
- Zone and Field Restrictions

The minimum capability which must be provided by a text searching system consists of exact term matching, coupled with Boolean logic combinations using the AND, OR, and NOT operators among terms. Exact term matching with Boolean AND, OR, and NOT allows textual data bases to be searched and documents retrieved in a useful way. By term we mean both complete words, such as "aircraft" and "university", and contiguous word phrases (CWPs) such as "wide-bodied aircraft" and "Syracuse University."

The effectiveness of document retrieval when the system has only these capabilities, however, is limited. Potential problems are illustrated in Figure 1a. Retrieval effectiveness consists of two components:

- Recall: the percentage of relevant documents actually retrieved in response to a query; and
- Precision: the percentage of the retrieved documents which are actually relevant to the query.

In general, retrieval of unnecessary documents is tolerable, and can be handled by human review of the retrieved documents, throwing out those which are not relevant, but limits are imposed by the practical requirements of reviewing all the retrieved documents in order to reject those which do not satisfy the stated information need.

The more serious problem is that of insufficient recall. Documents which are missed remain unknown to the user, and are likely never to be retrieved.

Query effectiveness can be improved by adding query capabilities as shown in the Venn diagram of Figure 1a.

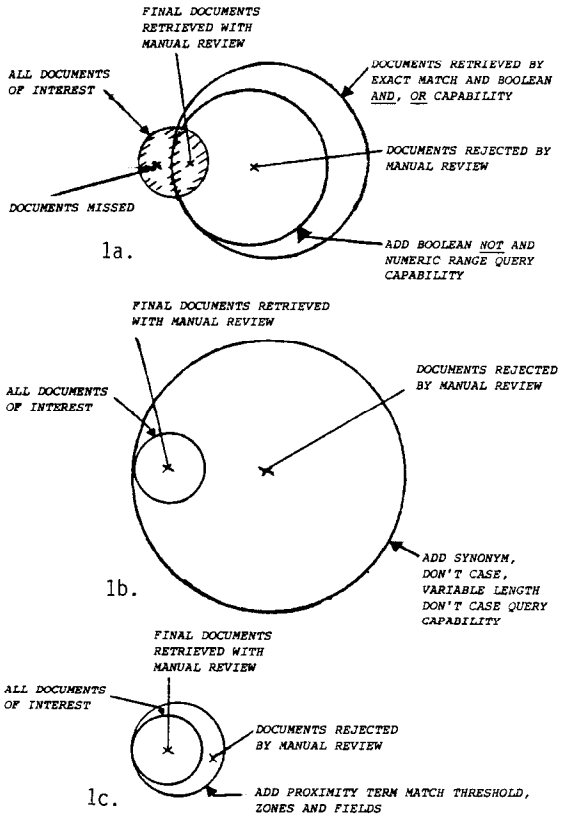

ILLUSTRATION OF QUERY PRECISION AND RECALL

Figure 1.

Precision can be increased at no cost in recall by adding the Boolean NOT and numeric ranging to the query capabilities provided by the text search system.

Spelling variations may result in a wide range of key terms within a textual data base. To access these terms easily, queries may be allowed to contain incompletely-specified terms. Such terms may consist of known characters in conjunction with place holders, representing either a single character (a "fixed-length don't care") or an unknown number of characters (a "variable-length don't care").

Adding pseudo-synonym capabilities through implementation of fixed and variable-length don't care conditions (FLDC and VLDC), as shown in Figure 1b, also serves to increase recall, although at some cost in precision.

Precision increases can be achieved by adding new capabilities peculiar to text applications, as shown in Figure 1c:

- Proximity: inter-term relationships stated in terms of words, sentences, or paragraphs
- Zone and Field Specification: requiring terms to occur within specified areas of the target documents. Such specified areas might be the title, abstract, author, or text areas.

Increases in capability along these lines result in concomitant increases in software complexity, additional bulk memory overhead, and longer processing time for each user query. Nevertheless, these capabilities are necessary whenever a generalized full-text search system is required.

2. INVERTED-FILE-SYSTEMS

An inverted file system is a searching system implementation model that uses files and information structures which explicitly associate index term search keys with sets of records from a data base. A fundamental property of an inverted file system is that it is able to determine these sets without directly retrieving the records in the data base itself. As a minimum, inverted files are composed of records, called postings lists which are lists of references from a particular index term to records in the data base. One implementation of inverted files orders these postings lists by the index key value associated with each of them. This implementation is called a posted term file set.

Maintenance considerations associated with inverted files for large scale data bases, however, generally dictate that the postings lists and the set of search keys be split into separate files. The file containing the index terms and pointers to postings lists is called an index file. Entries in the index file are normally ordered alphabetically, with primary and second level directories to facilitate term lookup.

File inversion has a conventional interpretation when applied to data bases comprised of structured records. A search key is associated with an assignment of a value to a field within a record, and a postings list is associated with that subset of the data base such that each record in the subset has the value for the field determined by the search key.

As an example, consider a personnel file, with records containing fields such as:

- Last Name

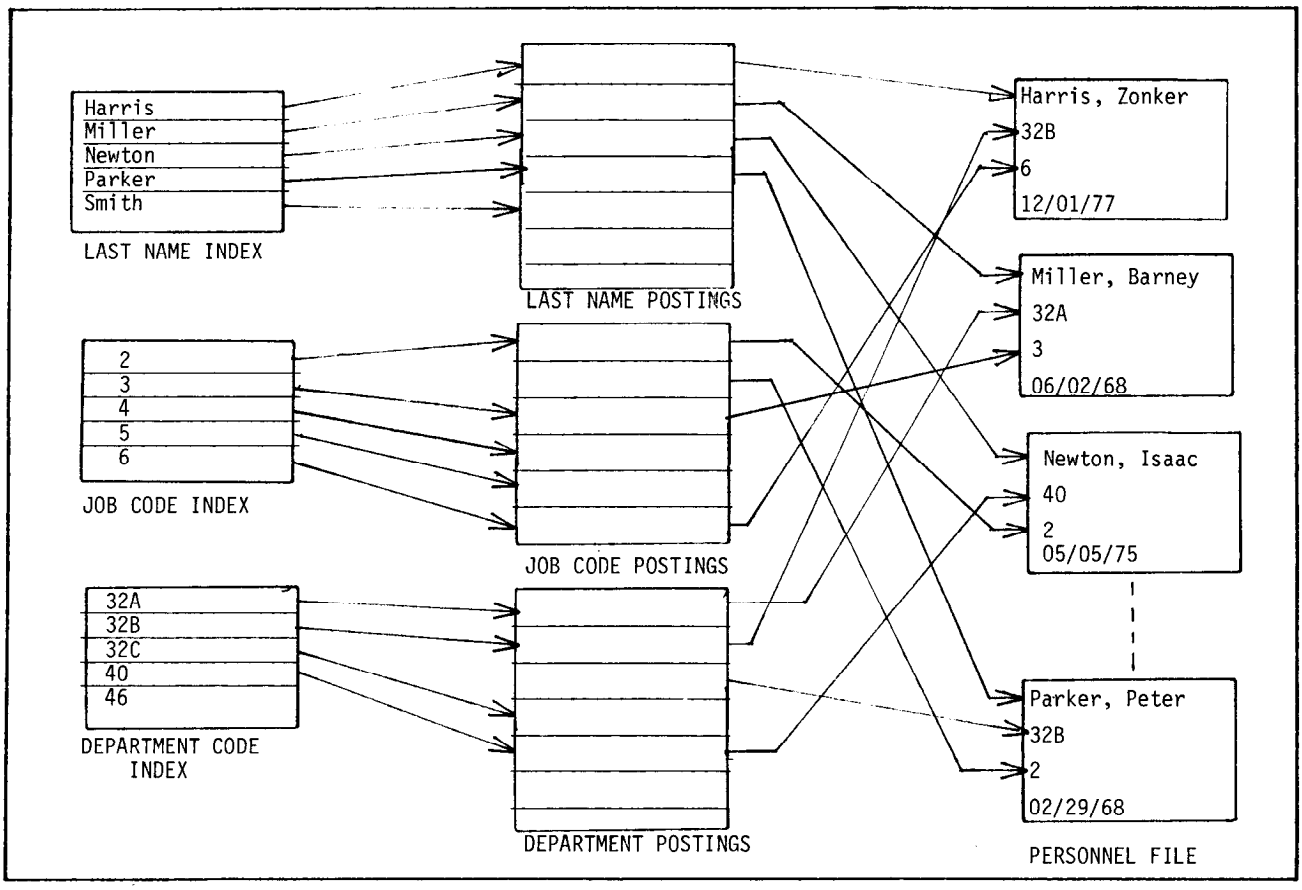

Figure 2. Inverted File Structure

- First Name
- Middle Name
- Department Code
- Salary
- Job Code
- Hire Date

When a file containing such records is inverted, separate index files are generated, each sorted according to its own key. For example, index files might be created ordered on last name, department code, salary, job code, and hire date. A record in one of these index files could contain only two fields:

- The key value (a particular Job Code, for example), and
- A pointer giving the location of the record in the primary file which has this value of the key field.

In general, though, any one key value can be found in more than one record in the primary file, and the location pointer in the index file points to a postings list in a postings file. Each postings list entry consists of a pointer to the primary file. An example of such a file structure is shown in Figure 2.

2.1 Inverted File Search and Retrieval

Retrieval from an inverted file on a single key involves three steps:

- Locating the key value in the index file;
- Locating the postings list associated with that key and that value; and
- Retrieving those records which contain that value of that key.

2.2 Strengths of Inversion

File inversion provides two major capabilities to a retrieval system:

- Rapid retrieval by multiple keys
- Query evaluation without reference to the primary file

These characteristics allow rapid response to queries such as:

> Which employees have JOB CODE = 6 and work in DEPARTMENT = 32B ?

or

How many NAME = 'SMITH'
make SALARY GE 27524 ?

Queries of the second type can be answered without ever accessing the personnel file, simply by counting the number of postings remaining after performing the AND operation between the postings list for SMITH and the range of postings for SALARY greater than or equal to $27,524.

Queries of the first type involve an additional step, that of retrieving records from the primary file, but only those records which are known in advance to meet the query criteria need be retrieved.

2.3 Weaknesses of Inversion

Inversion is not the perfect answer, however. There are a variety of costs associated with constructing and maintaining an inverted file system. Among the obvious costs are:

- Complex file structures
- Increased storage requirements
- Complex file maintenance processes

In addition, there are hidden limitations. In the personnel file example it is impossible to access the file based on FIRST NAME except by a serial scan of the entire primary file because the FIRST NAME field was not an inversion key.

2.3.1 Complex File Structures. A pair of files must be created for each field upon which the primary file is inverted. One of these files, the vocabulary file, contains an ordered list of the key values which have occurred in the field, associating each key value with its list of occurrences through a postings pointer. The second file, the postings file, contains a list of record pointers for each value in the vocabulary file.

In order to preserve the integrity of this file structure, any operations which alter the order of records in the primary file must be reflected in the pointers in the postings file, and any reorganization of the postings file must be suitably reflected in the pointers in the vocabulary file.

2.3.2 Storage Requirements. As more and more fields are inverted, the size of the inverted index files approaches (and may exceed) that of the primary file itself. Inverted file sets provide an excellent example of trading space for speed: the more fields are inverted, the more space is required, and the faster results can be obtained.

Size estimates can be readily obtained: each vocabulary entry must be large enough to contain a postings list pointer (normally a constant size, typically related to the computer's word length) and a key value, while each postings entry need contain only a pointer to the relevant record in the primary data base. Where keys are the same length, no space is wasted in the key field, but when keys are variable-length, additional space is required for either the maximum key length or for an additional key-length field.

2.3.3 Maintenance Processes. Maintenance of an inverted file set includes the following basic operations:

- Addition of new records
- Deletion of old records
- Modification of existing records

For each of these operations, all three files must be updated. The vocabulary file must be examined for each key value in an altered record in order to add new values and delete unused values from the vocabulary list. The postings file must be accessed for each value of the keys in order to add new postings or delete removed postings. The primary file must be updated by addition of a new record, removal of a deleted record, or modification of an existing record.

As more and more fields are inverted, maintenance becomes more and more complex. Each inverted field leads to two more sets of accesses: one to its vocabulary file, and one to its postings file.

3. EXTENSION TO FILES OF UNFORMATTED DATA

A number of information retrieval systems, mainly designed to store and process large collections of bibliographic data, have been built as index based systems using inverted file techniques. Items in these data bases are usually referred to as documents, although the data base record itself may represent subdocuments, such as articles within technical journals or newspapers.

These systems reference documents in the data base in much the same manner as card catalogs reference collections housed in libraries. Indexers extract from the document values or attributes associated with bibliographic citations such as subject, title, publication date, author(s), language(s). The indexers then analyze the content of the document, and tag it with a set of subject terms or keywords. Depending upon the subject matter of the data base, the indexers may be constrained to choose the terms from a predefined controlled vocabulary, and may use broad or narrow subject content criteria to determine which and how many terms are assigned. The set of keywords and values of certain of the citation related fields

becomes the index term vocabulary from which queries can be formulated.

Although the citations to the documents are themselves structured records, they may or may not be retained in the data base beyond their functional usage as transactions records for updating the vocabulary and postings lists. When the citations are retained, they may be stored as separate integral records within the data base, or they may be incorporated into the documents they reference. The net result, in any case, is the extension of inverted file search capabilities to unstructured bibliographic records.

4. INADEQUACIES OF THE CLASSICAL EXTENSION

Extending the inversion technique to files of unformatted textual material by imposing an external structure, such as assigned index terms, has proven to be inadequate to solve the complex problems of unrestricted text search which are required by many users. While it is true that structured files can be quickly accessed using inversion techniques, the structuring of text files has the effect of altering the very information which the users wish to query: the text itself.

No matter how well indexing is performed, the fact remains that only the indexes -- not the original documents -- are available for searching, and these surrogates can provide only the answers for which they were first developed.

Any transformation of the original input texts suffers from at least one of two flaws:

- Manual indexing leads to inconsistencies in term assignment
- Automated index term selection is consistent, but may be inaccurate
- Either approach limits the vocabulary, and thus the scope of the concepts which can be found in the data base to those originally embodied in the indexing assignment process.

These limitations, however, are precisely those that the text files were intended to avoid. The full generality of text search capability is available only when the full text is made searchable, leading naturally to the concept of full text inversion.

5. FULL-TEXT-INVERSION

In the context of a text based system, the records in the data base represent collections of textual material such as documents, and the index terms used as search keys are the words of text themselves. As such, index terms in text-based systems have positional relationships to each other beyond the standard term-document mapping.

Three varieties of text file inversion appear to be of interest:

- Full inversion, with comprehensive postings lists for each term in the vocabulary
- Partial inversion, where the entire vocabulary is preserved, but postings lists contain only minimal information (e.g., document identification, but not location within document)
- Incomplete inversion, where the vocabulary itself is limited by being filtered through a stoplist in order to remove the highest frequency structure words

In a fully inverted text search system, an entry within a postings list normally describes the document identifier, field, paragraph, sentence, and word location of the occurrence of a term within a document.

In a partially inverted system, an entry within a postings list contains the document identifier (and possibly a field qualifier). Thus the only information available is whether a given word occurs within a document (and possibly which fields contain the word).

Incompletely inverted systems can contain either complete postings lists or the limited postings lists of the partial inversion technique. Incomplete inversion limits the types of queries to which the system can respond without additional mechanisms. A query for the contiguous word phrase

gang of four

would lead to a preliminary retrieval of all documents containing both gang and four, but their relationship in the desired contiguous word phrase would require further analysis of the retrieved documents. Such analysis could be performed by the user or by a text streaming routine which would look only at those documents retrieved by the inverted file search.

Full inversion allows proximity relations between terms to be handled by the postings list merge processor as special cases of Boolean and arithmetic operators, with adjunctive masking operations on the postings word.

In a partially inverted system, an entry within a postings list contains the document identifier (and possibly a field qualifier). Thus the only information available is whether a given word occurs within a document (and possibly which fields contain the word). In order to

handle proximity relations in a partially inverted system, adjunctive text streaming routines must be added. In well-behaved queries, the strictly Boolean operators are handled by the postings list merge processor, in order to narrow down the search to a manageable number of entries in the postings lists associated with the partially evaluated query. The associated documents must then be retrieved, and the text streaming routines determine whether the non-Boolean criteria specified within the query are satisfied for each candidate document.

As might have been expected, the incorporation of text-theoretic operations into an inverted file framework introduces several new problems (in addition to the classical ones shared by almost all search key based retrieval systems). Some of the difficulties which can arise are illustrated by the following examples.

5.1 Contiguous Word Phrases

Consider, for example, the phrase:

changing of the guards

Even though this phrase conceptually represents but a single term, the fact that file inversion is based upon word units implies that the above phrase must be translated into an expression of words connected by proximity operators. If "<w+i>" designates the proximity operator followed within i words by, the example phrase is translated as the expression:

changing <w+1> of <w+1>
the <w+1> guards

Although evaluation of this expression does result in precisely the list of documents containing the example phrase, the amount of time and computer memory expended in merging the constituent term lists is considerable. Particularly irksome is the necessity to generate and maintain postings lists for prepositions and articles such as the and of, which appear many times in almost all documents in the data base. An alternative approach is to approximate the phrase by the expression:

changing <w+3> guards

It must be observed, however, that this latter expression is indeed an approximation. Not only would documents containing the desired phrase be retrieved, but also documents containing phrases such as "changing uniforms for guards ..." or "changing combination locks guards out of blockhouse...".

Another approach to CWPs includes them in the vocabulary explicitly, as two-word CWPs, three-word CWPs, etc. This approach consumes additional systems resources in two ways:

- The vocabulary file size is increased, both by the addition of new terms (the pairs or triples), and by the increased term length which results from such pairs or triples.
- The postings file size is increased by the addition of a postings list for each such term which has more than a single occurrence in the data base.

5.2 Partial Word Strings

Query terms which contain fixed or variable don't care characters are patterns which represent sets of terms, the elements of which are precisely those words which match the specified pattern. Since text file inversion methods produce postings lists for words, the query translation process must, as a preliminary step, search the vocabulary file in order to find all words matching the don't care patterns, and replace them with equivalent expressions which "or" the postings lists of matching words. For the sizes associated with vocabulary files in large text-based systems, the cost in additional storage associated with file structures necessary to perform these searches effectively is considerable, and the question of whether file inversion techniques are feasible in systems which require the full generality of word pattern matching as outlined above remains open.

A summary of query terms and capabilities is presented in Table 1.

5.3 Characteristics of Natural Language

5.3.1 Definitions. We begin by defining the following terms:

TYPE: A unique word.

TOKEN: An occurrence of a type in running text.

WORD: A character string consisting of a contiguous string of word characters, bounded on each side by strings of interword characters.

WORD CHARACTER: A character which may occur within a word. While variable, this set typically includes the alphabetic characters (A-Z), and the numerics (0-9), and may be extended to encompass a few of the special characters such as the apostrophe, decimal point (where it can be distinguished from the period), and the hyphen.

INTERWORD CHARACTER: Interword characters are all characters in the machine's character set which are not word characters. For purposes of completeness, document boundaries are treated as interword characters as well.

5.3.2 Characteristics of English. Kucera and Francis [KUC67] analyzed a sample of 1,014,232 words of running English text, taken from a variety of sources, and found that their sample contained 50,406 types.

The Oxford English Dictionary contains about 600,000 entries, but many of these are homographs which would appear identical to a string recognition program. On the other hand, the string recognition program would distinguish between verb forms, plurals, and other morphological endings which are lumped together in the dictionary.

Word frequencies in the Kucera and Francis sample are highly skewed: the 100 most frequent types (the, and, of, etc.) account for 47.4% of the tokens, although they make up only 0.198% of the total vocabulary. Low-frequency words, on the other hand, make up a large portion of the vocabulary. Indeed, in the sample, 44.723% of the types occurred only once. The mean values from this sample were:

- 20.1213 tokens/type
- Standard deviation 448.3906
- Mean Type Length: 8.13 characters
- Mean Token Length: 4.74 characters

5.4 Implications

The Kucera and Francis corpus was relatively small, only a little over one million words of running text. This corresponds to only 500 documents which average 2,000 words each. When their results are extrapolated to larger data bases we find that the skewness of the distribution becomes even worse. Consider, for example, a data base containing one million documents similar to those in the Kucera and Francis corpus. We expect such a data base to have the following characteristics:

- Vocabulary size: 300,000 to 500,000 types (more if not all material is in standard English)
- Hapax legomena: 150,000 to 250,000 types will occur only once
- Total text: 2 billion words
- Average Postings Lengths:
  - 100 High Frequency Words: 10 million postings each
  - 249,900 Medium Frequency Words: 4,000.6 postings each
  - 250,000 Hapax legomena: 1 posting each

These expectations are borne out by empirical analysis of large-scale textual data bases. Indeed, one of the data bases with which we have had experience appears to have an even larger potential vocabulary and higher hapax percentage because it is not restricted to standard English text.

TABLE 1: Query Capabilities

Query Capability Category	Match Specification	Query Statement	Example
Term Matching	Exact Match	SYRACUSE	Will match SYRACUSE only
	Fixed Length Don't Care (FLDC)	SM@TH	Matches SMITH, SMYTH, etc.
	Variable Length Don't Care (VLDC)	?PRESS	Matches PRESS, COMPRESS DEPRESS
Numeric Ranging	Greater Than	N>10	Matches 11, 222, 301, 10134, etc.
	Less Than	N<51	Matches 50, 29 1, etc.
	Between Limits	10<N<51	Matches 50, 25, 11, etc.
Term Processing	Boolean AND	SYRACUSE AND UNIVERSITY	Terms may appear anywhere within the document, unless otherwise specified.
	Boolean OR	UNIVERSITY OR SCHOOL	
	Boolean NOT	UNIVERSITY NOT COLLEGE	Only documents which contain UNIVERSITY and do not also contain COLLEGE
	CWP	SYRACUSE UNIVERSITY	Contiguous Word Phrase
	Word Proximity	SYRACUSE +3w UNIVERSITY	SYRACUSE followed within 3 words by UNIVERSITY
	Sentence Proximity	PLANE +0s TRAIN	PLANE and TRAIN within the same sentence.
	Paragraph Proximity	CONFERENCE +3p PROCEEDINGS	CONFERENCE and PROCEEDINGS within 2 paragraphs of each other.
Zones	Zone Specifier	[LOC] SYRACUSE	SYRACUSE in the location zone.

6. EVALUATION OF FULL TEXT INVERSION

6.1 Existing Inverted Text File Systems

Several inverted file systems have been implemented for specific kinds of textual data bases. Examples are LEXIS and WESTLAW, [LAR77], for legal data retrieval; ELHILL-III, [HUM75], for medical data

retrieval at the National Library of Medicine; and RECON, for aerospace data at NASA [PRY76]. Other general purpose inverted file systems are commercially available, such as ORBIT, available through Systems Development Corporation, [SDC76], DIALOG, available through Lockheed Missiles and Space Company, [LOC76] and RECON, available through Informatics.

As far as is known, none of these systems utilize full text inversion nor do they allow unrestricted query formulation. As a minimum, stoplists are used to eliminate the highest-frequency words from the inversion process.

6.1.1 Performance. The performance of these inverted file systems varies over a wide range, depending on the complexity and capability of the query and the size of the textual data base. The following are representative search times for different query complexities and capabilities where the textual data base is a few billion characters, and the host computer is a large, general-purpose computer.

Simple queries without proximity or zoning can normally be processed in a few seconds; complex queries without proximity or zoning normally can be processed in ten to twenty seconds. These times reflect the exclusion of the most common English words from the inverted indexes. Simple queries with proximity or zoning normally require a few tens of seconds.

6.1.2 Cost. The average cost per query as reported in a survey ,[CUA76], of inverted file users is between $20 and $30. Upper limit costs were reported to be as high as $100.

High storage costs are associated with such systems, too. The inverted index files range in size from .5 to 3 times the size of the associated primary file, depending on the amount of compression associated with term usage and the degree to which term occurrences are described in the postings lists.

6.2 Practical Uses of Text Inversion

Text file inversion is a practical technique under the following conditions:

- The data base is specialized (such as a legal or medical data base).
- Queries are restricted to limited areas of discourse.
- Fully generalized contiguous word phrase, proximity, and partial string matching are not necessary.
- The full text need not be inverted.
- The data base need not be updated in real time.
- The documents are well controlled -- standard English is used with few or no idiosyncratic spellings and variations.
- The data base is relatively small (thousands of documents, a few millions of characters)

A specialized data base automatically limits the universe of discourse to specialized topics, and implies the types of questions to which the data base is expected to respond. By limiting the universe of discourse, we find that a specialized vocabulary can be used, and those terms which are expected to occur in queries can be identified in advance, thus allowing for the construction of a specialized set of indexes.

When the index terms are controlled in this manner, useful results can be obtained without requiring the full generality of CWP, proximity, and partial match processing. Contiguous word phrases corresponding to the specialized terms used in the limited vocabulary can be included in the indexes without requiring that all such phrases be included.

Because the system is expected to respond to only a limited range of queries, and because the form and content of these queries are known in advance, the full text need not be inverted. Typical implementations utilize a stoplist which prevents inversion on the highest-frequency words in order to reduce system overhead.

Inverted file maintenance is a major process. Whenever such maintenance can be deferred to a time convenient to the system implementors and operators (as opposed to the users), it can be made more effective by taking advantage of batch processing economies of scale.

Linguistic standardization also helps to reduce the costs associated with large inverted file systems. All the problems associated with vocabulary control and maintenance are exacerbated whenever the input contains a wide variety of idiolects, jargon, and non-standard usage.

6.3 Circumstances Hostile to Text Inversion

Text file inversion breaks down as more and more of the following conditions must be met:

- The data base is heterogeneous.
- Queries cannot be predicted.
- The input is uncontrolled (intelligence reports, for example).
- The entire data base must be searchable, including all forms of CWPs.

- The data base must be updated in real time.
- Fully generalized character string, contiguous word phrase, and proximity matching is required.
- The data base is large (billions of characters, millions of documents).

These conditions are appearing more and more frequently as technological innovation and computer development is bringing the capability of storing and searching large quantities of textual material within the reach of more and more potential users. While the initial impetus toward developing large textual storage and retrieval systems came from the U.S. intelligence community, similar needs are now being felt by libraries, corporations, and news media organizations as well.

7. CONCLUSIONS

In summary, we find that inversion is a technique which is applicable to textual files only under certain highly restricted conditions, and is by no means a panacea for text search problems. Particularly, we conclude that inversion is not cost-effective when unrestricted querying of large textual files is required.

REFERENCES

[CUA76] Cuadra, Carlos A., Fishburn, Mary, and Wanger, Judith, Impact of On-Line Retrieval Services: A Survey of Users, 1974-75 Systems Development Corporation, 1976

[HUM75] Hummel, Donald J., "A Comparison of an Online Retrieval Service Employing Two Distinct Software Systems," Journal of Chemical Information and Computer Science, February 1975, 15(1) pp 24-27.

[KUC67] Kucera, Henry; and Francis, W. Nelson; Computational Analysis of Present-Day American English, Brown University Press, Providence, Rhode Island, 1967.

[LAR77] Larson, Signe, "Online Systems for Legal Research," ONLINE, Volume 1, Number 3, July 1977, p 10.

[LOC76] Lockheed Information Systems, "A Brief Guide to DIALOG Searching," Lockheed Corporation, September, 1976.

[PRY76] Pryor, Harold E., "An Evaluation of the NASA Scientific and Technical Information System," Special Libraries, 1976, No. 66(11), pp 515-519.

[SDC76] SDC Search Service, "ORBIT User's Manual," System Development Corporation, March, 1976.