Efficient Building and Querying of Asian Language Document Databases

Phil Vines Justin Zobel
Department of Computer Science, RMIT University
PO Box 2476V Melbourne 3001, Victoria, Australia

Email: phil@cs.rmit.edu.au
Phone: +61 3 9925 3310
Fax: +61 3 9662 1617

Abstract

Over recent years there has been great interest in Asian-language information retrieval. Much work has been done on the identification of appropriate index terms, usually involving one or more of characters, words, or bigrams, or some combination thereof. Although the relative effectiveness of these approaches has received considerable scrutiny, efficiency considerations have had less attention, but are of at least equal importance in the implementation of commercial systems. In this paper we explore the relative efficiency of several indexing approaches used for Asian-language text, and investigate the tradeoffs between effectiveness and efficiency.

Efficient Building and Querying of Asian Language Document Databases

Abstract

Over recent years there has been great interest in Asian-language information retrieval. Much work has been done on the identification of appropriate index terms, usually involving one or more of characters, words, or bigrams, or some combination thereof. Although the relative effectiveness of these approaches has received considerable scrutiny, efficiency considerations have had less attention, but are of at least equal importance in the implementation of commercial systems. In this paper we explore the relative efficiency of several indexing approaches used for Asian-language text, and investigate the tradeoffs between effectiveness and efficiency.

1 Introduction

Although research on various aspects of Asian-language information retrieval has been underway for some time [6], there has been a noticeable increase in recent years. This is partly due to the increased interest in information retrieval generally, resulting from the growth of the Web and the interest it has focused on search engines and retrieval techniques. In addition the decision to include a Chinese retrieval track in the TREC [7] experiments further fuelled interest in this area [1, 3], and more recently a similar effort has been undertaken for the Japanese language [2].

Most of the work on Asian text retrieval has focused on the effectiveness of different indexing, ranking, and query strategies. In English language information retrieval it is generally accepted that words, once stopped and stemmed, are the most appropriate unit of indexing for most applications. The situation is much less clear in languages such as Chinese, Japanese and Korean (the *CJK* group), which do not have explicit word boundaries. Most research has looked at characters, words, and bigrams (pairs of adjacent characters), or some combinations of these. It appears that words and bigrams, either separately or combined, provide more effective retrieval performance than simple characters [3].

The growth of the Web has lead to a significant increase in the use of search engines that operate as the back end to Web search tools. Use of the Web and hence these search engines is currently increasing at an exponential rate, and the providers of the more popular search engines continue to make significant investments in computing hardware in order to maintain performance. Because of the enormous demand for these services, providers are continually looking for more efficient ways to implement the search engines, and frequently adopt approaches that sacrifice retrieval effectiveness in return for increased efficiency. It is tempting to suppose that this is a passing problem that will be solved once

the next generation of computers arrives. However, the amount of data on and use of the Web is increasing at a rate that significantly outstrips increasing hardware speeds, and, while CPU speeds have grown by an order of magnitude every few years, the increase in the speed of disk access time has been negligible by comparison, limiting the efficiency improvements available through advances in hardware.

Thus efficiency is an important issue for search engines. While words and bigrams have been shown to be effective for Asian-language information retrieval, the resource requirements of these techniques, in particular bigrams, are significantly greater than for characters. In this paper we experimentally measure the resource requirements for indexing and querying two large test collections. These experiments allow us to examine the implications for efficiency of different kinds of indexing. We show that bigrams lead to much bigger vocabularies and indexes, and slower querying, but, in our Japanese collection, significantly greater effectiveness. For Chinese, word indexing is both more efficient and more effective.

2 Measuring Efficiency

Most research to date has concentrated on the effectiveness of different indexing approaches. This is measured in terms of *recall* (that is, how many of the relevant documents where found) and *precision* (what proportion of the documents returned were actually relevant). A common way to characterize this is with *interpolated recall-precision* values, which are the precision at fixed levels of recall. These figures are often averaged to give an overall *recall-precision* (non-interpolated). Another common characterization is to give precision figures for various fixed numbers of documents fetched.

The retrieval system we use is *mg* [4, 8], a freely available retrieval system that provides efficient full-text indexing of documents and supports free form text queries. However the system was not originally designed to handle Asian-language documents, and we have dealt with this by preprocessing the documents before they are indexed by *mg*. Our document collections use a mixture of 16-bit codes for the Chinese and Japanese characters, interspersed with English-language words represented by 8-bit ASCII codes. It is relatively easy to distinguish between the ASCII and other codes, as the Chinese and Japanese codes always have the high order bit set in each byte, where as in plain ASCII encoding this is not the case. (The Extended ASCII coding set often used for various European languages also uses the high order bit, and hence it would not be possible to embed this form of encoding within Chinese or Japanese.) We converted 16-bit CJK codes to ASCII by simply using the hexadecimal representation as an ASCII string, for example the code $bab0_{16}$ was converted to the ASCII string "bab0". If the indexing unit was to be characters we would simply emit these codes, separated by spaces, as input to *mg*, which treats these as indexable tokens and processes them in the normal way. If words are used as the unit of indexing, a two character word would be encoded as "bab0cdef", and so on. While this process is, as a retrieval mechanism, somewhat indirect, it has no impact on our measurements of retrieval speed or database size.

Table 1: *Collection statistics for the TREC Chinese and NTCIR Japanese collections.*

	Size (Mb)	Number of documents	Distinct characters	Distinct words	Distinct bigrams
Chinese coll'n	163.2	164,789	15,520	88,683	1,589,685
Japanese coll'n	312.7	339,483	12,252	459,673	1,379,443

For the purpose of our experiments we have used data and queries from two collections, the TREC collection of Chinese documents and 28 queries [7]. and the NTCIR collection of Japanese documents and 53 queries [2]. Some statistics relating to the size and numbers of distinct terms according to the indexing method used are shown in Table 1; the segmentation technique used to extract words is discussed below.

3 Indexing Approaches

For English text retrieval, queries and documents are usually matched at the word level. For Asian text retrieval, the two most common levels of matching are characters or words. These techniques tend to suffer from being too general or too specific, respectively. Many words are two characters long, and often the individual characters have a number of meanings or combine with other characters to form a range of specific meanings. This means that character-based matching tends to retrieve irrelevant documents, because query terms match documents with characters in common, but are actually part of words with different meanings. Word-based matching, on the other hand, misses relevant documents in which the words are similar in meaning to those in the query, typically with a character in common, but are not identical.

Character Indexing

Character indexing is the simplest and most obvious form of indexing for CJK languages. The number of distinct characters, while not well defined, is about 15,000 for Chinese (counting all character-level tokens), and somewhat less for Japanese and Korean. There is a small number of terms, and the size of the dictionary is quite small.

In a text database, each distinct indexed term is held in a dictionary (or vocabulary). For each term there is an inverted list identifying the documents containing that term. A consequence of character-based indexing is that the number of inverted lists is small, but each list is relatively long. For a large database, fetching and processing a long inverted list is a significant component of query evaluation costs.

Word Indexing

As there are no spaces between characters in CJK languages, word indexing requires a preprocessing step to parse the character stream into words. There has been much research on how best to do this [9]. The most common way to this is via dictionary segmentation, using a dictionary of known words to parse the text. Generally, such methods are reasonably successful, although there is always some degree of ambiguity. For example, if the dictionary contained the word entries *ab*, *abc*, and *cd*, the string *abcd* could be parsed as *ab*|*cd* or *abc*|*d*. Greedy parsing will choose the latter, which in this case is more likely to be wrong. However parsing the entire text in reverse will overcome this problem, and tend to produce better results. Other researchers have used a technique of generating multiple possible parse sequences and using a word-scoring system to choose the most probable correct one.

One of the problems that all dictionary-based techniques suffer from is that they don't recognize words which are not in the dictionary, such as names of people and places. Statistical techniques have been proposed to overcome this problem. While segmentation is never perfect it is probably not a great problem [5]. Provided a that a word segmented incorrectly in a document is also segmented in the same incorrect manner in the query the terms will still match, and in most cases the document will still be retrieved.

Word-level indexing reduces the number of tokens in each document, thus reducing overall index size. For dictionary-based segmentation, the size of the database vocabulary cannot exceed the sum of the size of the dictionary and the size of the character set; however, much Asian-language text includes text in other languages, in particular English, which can further increase vocabulary size. On average inverted lists will be much shorter than with character-based indexing.

Bigram Indexing

Bigram indexing generates terms by using overlapping pairs of characters. Thus the character string *abcd* would give *ab*, *bc*, and *cd* as index terms. Bigrams give better coverage of the text and several researchers have shown that retrieval effectiveness is typically slightly better than words and characters [3]. However they generate much bigger dictionaries than characters or words. In Chinese for example there are roughly 10^4 characters and roughly 10^5 words. Because bigrams can arise out of the random juxtaposition of words they can potentially number in the order of 10^8. To examine dictionary size we processed the first n kilobytes of text from the Japanese collection for a range of values of n, observing the number of distinct bigram terms as shown in Figure 1. As can be seen in the graph, there is no obvious trend toward a limit in the number of terms as the collection size increases. However, individual inverted lists are expected to be short.

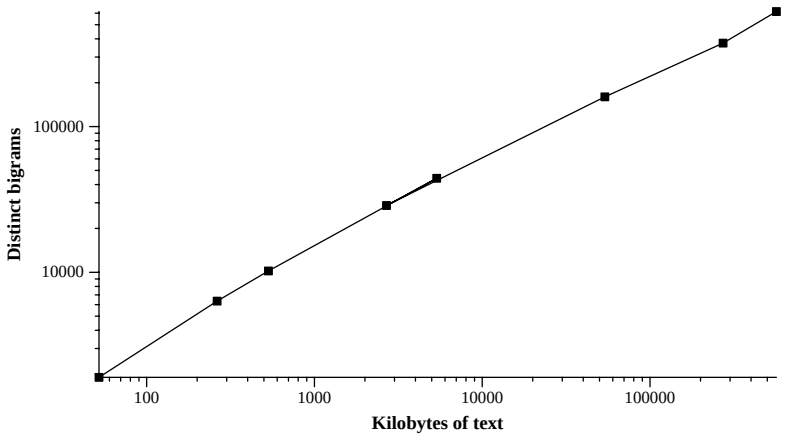

Figure 1: *Rate of increase in vocabulary terms for Japanese data indexed using bigrams.*

4 Resource Requirements

There are two distinct activities in the normal use of a text retrieval or document database system, the building of the database and the querying of it. The building occurs infrequently, and possibly on a more powerful machine than that used to answer the queries. A production system will process many queries and must be able to do this efficiently, and thus it pays to do as much pre-processing as possible during the building stage, to make the query processing as efficient as possible. The experiments in this paper were all performed on a PC with a 333 Mhz Intel PII processor and 256 Mb of RAM, running the Linux operating system.

Collection Building

For the *mg* system, text database construction is a two-pass process. The first pass builds a dictionary of all the index terms encountered, while the second pass constructs the inverted lists. For efficiency reasons it is desirable to hold the dictionary in memory during both passes. However some approaches tend to have particularly large vocabularies and thus may have large memory requirements. Table 2 shows the time taken by the first and second pass of the collection building process, and the total size of all the files needed by the *mg* system for the collection including the text and all indexes. These are all stored in a compressed format, both to save space and to facilitate faster access [8]. It is usual for this total to be less than the original size of the uncompressed text—163 Mb and 312 Mb respectively for the Chinese and Japanese collections.

An interesting observation is that the collection size of the Chinese word-indexed files is less than that of the character-indexed collection. When each document is segmented into words, there are significantly less tokens compared to using characters. In the Chinese collection there were 77 million index terms using character segmentation, and 52 million using word segmentation. For the Japanese collection there were around 101 million terms using character

Table 2: *Database sizes and build times for Chinese and Japanese collections.*

	Number of terms	First pass (sec)	Second pass (sec)	Total time (sec)	Database size (Mb)
Chinese collection					
Characters	15,520	519	876	1395	133
Words	88,683	906	1371	2277	117
Bigrams	1,589,685	963	1727	2690	285
Japanese collection					
Characters	12,252	598	1095	1693	161
Words	647,256	723	1275	1503	254
Bigrams	1,379,443	1620	3030	4650	508

indexing and 94 million using word indexing, a much smaller difference. In addition, there was a surprising number of distinct word terms in the Japanese collection, leading to the larger overall collection size. Most of these terms are English words (with many misspellings).

In both collections the bigram indexing approach generated a much larger number of distinct index terms, took longer to process, and the total database size was significantly larger. This is because there are many terms which occur infrequently, giving short inverted lists that do not yield much compression. sheer number of terms greatly adds to the size of the indexes. The bigram collections where about 170% of the size of the original text, whereas the other approaches varied from 50% to 80% of the original collection size. Perhaps more importantly, the number of new bigram terms appeared to be growing almost linearly in proportion to the collection length, as shown in Figure 1. This suggests that there will be problems in scaling this technique for larger collections.

Collection Querying

Querying time tends to be proportional to the number of terms in in a query. The reason for this is that each term requires that the list of documents containing that term be fetched, and included in a vector product computation. Each term index must be fetched from disk, and in terms of elapsed time this is by far the most time-consuming activity, and the processor will often lie idle waiting for the disk. A typical example is shown in Figure 2, where the number of terms in a query is plotted against the CPU time required to retrieve the top 1000 documents from the Chinese collection, using character indexing. The time is averaged over all 28 queries; queries of n terms were generated by choosing the first n characters from each query. As mentioned above it is advantageous to compress the term indexes, so that each disk access can effectively fetch more data, reducing the total number of accesses and improving caching [8].

The fact that times are proportional to the number of terms in the queries

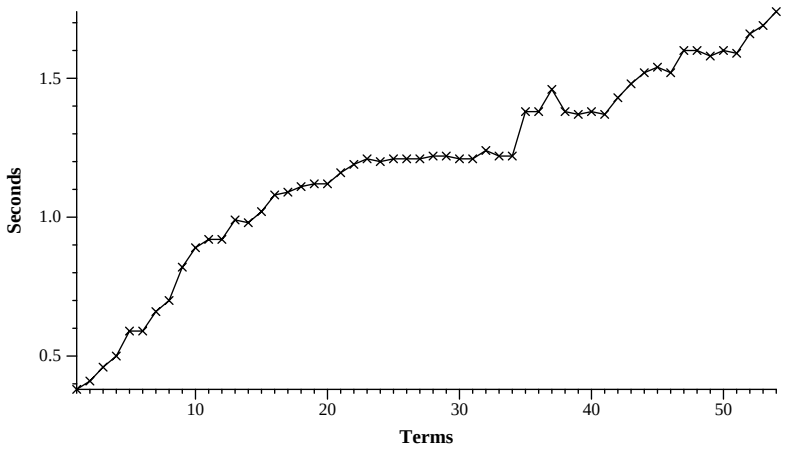

Figure 2: *Number of query terms vs CPU time: Chinese collection, character indexing.*

Table 3: *Query times for different indexing techniques on the Chinese and Japanese collections.*

	Total time (sec)	Av. no of terms	Time per term per query (sec)
Chinese collection			
Characters	135	116	0.021
Words	57	65	0.016
Bigrams	188	116	0.030
Japanese collection			
Characters	420	385	0.021
Words	255	191	0.025
Bigrams	423	385	0.021

is again illustrated by the total query times for each of the indexing methods, as shown in Table 3. Word indexing has only about half the number of terms per query as character or bigram indexing, and consequently runs about twice as fast. We have excluded the processing time taken to segment the query, however this would be small, provided it was running on a machine that could afford to keep the dictionary loaded in memory. The average time to process each term in each query is also shown. Although the test queries are long, Figure 2 show that the processing cost is roughly linear in query length, and the relative efficiency of the indexing methods is as applicable to short queries as to long queries.

Retrieval Effectiveness

The average recall precision for the different indexing methods for each of the collections is shown in Table 4. For the Chinese collection, word and bigram

Table 4: *Average precision for different indexing techniques*

	Chinese	Japanese
Characters	0.449	0.212
Words	0.502	0.234
Bigrams	0.494	0.332

indexing give about the same performance; both are slightly better than character indexing. For the Japanese collection, character and word indexing produce similar results, while bigram indexing is noticeably better. Although we have only shown the average precision it was in this case a good indicator of the relative performance according to each measure. The similarity measures used are the best identified in our work with Chinese text at TREC.

5 Conclusions

We have investigated the relationship between efficiency and effectiveness for retrieval of Asian-language text. The effectiveness of several indexing methods—in particular, character-based, word-based, and bigram-based—has been widely explored in previous work. In this paper we have re-examined these indexing techniques with regard to efficiency.

Word indexing is the most efficient indexing technique for Chinese. It yields modest vocabulary size, small database size, fast query evaluation, and high effectiveness. Retrieval results for the Chinese collection suggest that word indexing performs similarly to bigram indexing, and other researchers have come to similar conclusions [3]. We conclude that, overall, word-based indexing is the best method for Chinese text.

For our Japanese collection—which, unlike the Chinese collection, contains a significant quantity of English text—the results are more mixed. Word indexing provides the fastest query evaluation, with effectiveness better than character indexing but significantly worse than bigrams; while character indexing leads to the smallest storage overheads. Thus each technique is the best on one of our comparison criteria and no overall conclusion can be drawn.

References

[1] A. Chen, J. He, L. Xu, F. Gey, and J. Meggs. Chinese text retrieval without using a dictionary. In *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 42–49, Philadelphia, July 1997.

[2] K. Kageura. NACSIS Corpus Project for IR and terminological research. In *Natural Language Processing Pacific Rim Symposium*, pages 493–496, Phuket, Thailand, December 1997.

[3] K. L. Kwok. Comparing representations in Chinese information retrieval. In *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 34–41, Philadelphia, July 1997.

[4] MG public domain software for indexing and retrieving text, including tools for compressing text, bilevel images, grayscale images, and textual images, 1995. Available from ftp://munnari.oz.au/pub/mg and from http://www.mds.rmit.edu.au/mg/.

[5] V. B. H. Nguyen, P. Vines, and R. Wilkinson. A comparison of morpheme and word based document retrieval for Asian languages. In R. R. Wagner and H. Thoma, editors, *International Conference on Database and Expert System Applications — DEXA 96*, pages 24–33, Zurich, Switzerland, 1996. Springer.

[6] Y. Ogawa, A. Bessho, and M. Hirose. Simple word strings as compound keywords: An indexing and ranking methos for japanese texts. In *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 227–236, Pittsburgh, June 1993.

[7] R. Wilkinson. Chinese document retrieval at TREC-5. In D. K. Harman, editor, *The Fifth Text REtrieval Conference (TREC-5)*. NIST, November 1996.

[8] I.H. Witten, A. Moffat, and T.C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Van Nostrand Reinhold, New York, 1994.

[9] Z. Wu and G. Tseng. Chinese text segmentation for text retrieval: Achievements and problems. *Journal of the American Society for Information Science*, pages 532–542, October 1993.