Declustering Web Content Indices for Parallel Information Retrieval [1]

Yoojin Chung [1], Hyuk-Chul Kwon [2], Sang-Hwa Chung [2], and Kwang Ryel Ryu [2]

[1] Research Institute of Computer, Information & Communication,
Pusan National University, Pusan, 609-735, South Korea
chungyj@pusan.ac.kr
[2] School of Electrical and Computer Engineering,
Pusan National University, Pusan, 609-735, South Korea
{hckwon,shchung,krryu}@hyowon.pusan.ac.kr

Abstract. We consider an information retrieval (IR) system on a low-cost high-performance PC cluster environment. The IR system replicates the Web pages locally, it is indexed by the inverted-index file (IIF), and the vector space model is used as ranking strategy. In the IR system, the inverted-index file (IIF) is partitioned into pieces using the lexical and the greedy declustering methods. The lexical method assigns each of the terms in the IIF lexicographically to each of the processing nodes in turn and the greedy one is based on the probability of co-occurrence of an arbitrary pair of terms in the IIF and distributed to the cluster nodes to be stored on each node's hard disk. For each incoming user's query with multiple terms, terms are sent to the corresponding nodes that contain the relevant pieces of the IIF to be evaluated in parallel. We study how query performance is affected by two declustering methods with various-sized IIF. According to the experiments, the greedy method shows about 3.7% enhancement overall when compared with the lexical method.

1 Introduction

In general, Web search engines replicate the Web pages locally, index them, and do keyword based searching in this local collection. In what follows, we use the words *documents* and *Web pages* interchangeably. In this paper, a PC cluster interconnected by a high-speed network card is suggested as a platform for fast IR service. For efficient query processing, specialized indexing techniques have to be used with large document collections. In this work, documents are indexed using inverted files [2]. Since there are several machines in the PC cluster, it is reasonable to distribute the index among them. In this work, the global inverted-index file (IIF) is partitioned into

[1] This paper was supported in part by the Korea Science and Engineering Foundation under contact NO. 2000-2-30300-002-3.

N. Zhong et al. (Eds.): WI 2001, LNAI 2198, pp. 346–350, 2001.

pieces using the lexical and the greedy declustering methods. The approach used in [3] is similar to ours.

This paper is organized as follows. In Section 2, we present our PC cluster system and we detail the declustering methods. In Section 3, our experiments and results follow. Finally, we conclude with final remarks.

2 Parallel IR System

The overall working mechanism of the parallel IR system model can be explained as follows. We define an entry node as a node that accepts a user query and distributes query terms to processing nodes based on the declustering information described in subsection 2.2. Each processing node consults the partitioned IIF using the list of query terms delivered from the entry node, and collects the necessary document list for each term from the local hard disk. Once all the necessary document lists are collected, they are transmitted to the entry node. The entry node collects the document lists from the participating processing nodes, performs required IR operations and ranks the selected documents according to their scores. Finally the sorted document list is sent back to the user as an IR result.

2.1 Network Architecture

The environment for our parallel IR system is a PC cluster interconnected by a high-speed network card, which is a cost-effective platform for fast IR service. Figure 1 shows the environment where our parallel IR system is implemented, which is an 8-node SCI-based PC cluster system.

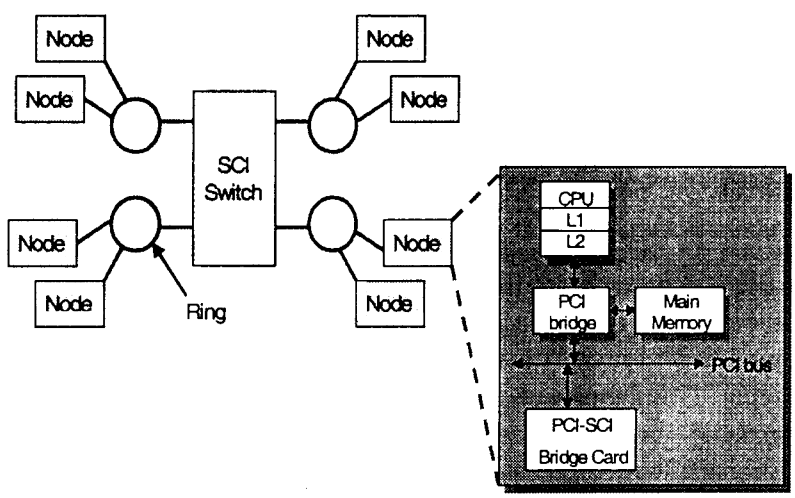

Fig. 1. SCI-based 8 node PC cluster system

2.2 Two Declustering Methods

For the efficient parallelization of the system, it is important to find out the most time consuming part in executing the IR system. Using the sequential IR system developed previously [1], we analyze the system's execution time, and find that the most time consuming part is disk access that takes approximately 45 % of total query execution time.

Thus, it is necessary to parallelize disk access. This can be done by partitioning the IIF into pieces and distributing the pieces to the processing nodes in a PC cluster. Our parallel IR system consists of multiple processing nodes and one of which is an entry node. In our system, documents in the collection are not distributed to multiple processing nodes but all of them are stored in an entry node. But it is desirable to have the IIF appropriately declustered to the local hard disks of the processing nodes because our IR system processes user's query in parallel on a PC cluster.

We can achieve maximum parallelism if the declustering is done in such a way that the disk I/O and the subsequent ranking processes are distributed as evenly as possible to all the processing nodes. We use two declustering methods. The first one is a *lexical declustering* method that just assigns each of the terms (together with its list of document id and weight pairs) in the IIF lexicographically to each of the processing nodes in turn, repeatedly until all the terms are assigned. The second one is a *greedy declustering* method that performs better than the lexical method. Our greedy declustering method tries to put together in the same node those terms that have low probability of simultaneous occurrence in the same query. If the terms in a query all happen to be stored in the same node, the disk I/O cannot be done in parallel and also the ranking processes cannot readily be processed in parallel. For an arbitrary pair of terms in the IIF, how can we predict the probability of their co-occurrence in the same query? We conjecture that this probability has a strong correlation with the probability of their co-occurrence in the same documents. Given a pair of terms t_i and t_j, the probability of their co-occurrence in the same documents can be obtained by the value that is the number of documents in which the two terms t_i and t_j co-occur divided by the number of all the documents in a given document collection. We calculate this probability for each of all the pairs of terms by preprocessing the whole document collection.

In the first step of our greedy declustering algorithm, all the terms in the IIF are sorted in the decreasing order of the number of documents where each term appears. The higher this number the more important the term is in the sense that it is quite likely to be included in many queries. This type of terms also have a longer list of documents in the IIF and thus causes heavier disk I/O. Therefore, it is advantageous to store these terms in different nodes whenever possible for the enhancement of I/O parallelism. Suppose there are n processing nodes. We assign the first n of the sorted terms to each of the n nodes in turn. Each of next n terms is assigned to the node that contains the lowest summation of probability of co-occurrence of the term in the IIF and every term in the node. This process repeats until all the terms in the IIF are assigned. When the size of the document collection is large and thus the co-occurrence

probability data is available only for those terms that are significant, the remaining terms are declustered by the lexical method mentioned previously.

3 Experiments

3.1 Comparison of the Greedy and the Lexical Declustering Methods

The greedy declustering method is compared with the lexical method on a test set consisting of 500 queries each containing 24 terms. The 8-node PC cluster is used for the experiment. To generate the test queries we lexically sampled 500 documents from a document collection. From each document, the most important 24 terms are selected to make a query. The importance of a term in a document is judged by the vector space model. Therefore, a term in a document is considered important if its frequency in that document is high enough but at the same time it does not appear in too many other documents.

Table 1. Comparison of the lexical and the greedy declustering methods(unit: sec)

	Lexical declustering	Greedy declustering	Enhancement Ratio (%)
Average query processing time	1.174	1.130	3.7
Average disk access and local IR operation time	0.794	0.749	5.7

Table 1 shows the experimental results comparing the lexical and the greedy declustering methods using those 500 queries on our 500,000-Korean-document collection. The greedy method shows about 3.7% enhancement overall when compared with the lexical method. Since the two methods show differences only during the disk access and the local IR operations performed at the processing nodes, the time spent for those operations is measured separately. In this measurement, the greedy method shows about 5.7% enhancement, which is more significant than the overall enhancement.

3.2 Effect of the Greedy Declustering Method with Various-Sized IIF

In this subsection, the performance of the parallel IR system is analyzed with the number of documents increased up to 500,000. The 8-node PC cluster and the greedy declustering method are used for the experiment. The size of IIF proportionally increases as the number of documents increases. For example, the size of IIF is 300 Mbytes for 100,000 documents, and 1.5 Gbytes for 500,000 documents.

The experimental result is presented in Figure 2. It takes 0.265 seconds to process a single query with the 100,000 document IIF, while it takes 0.477 seconds with the 200,000 document IIF and 1.130 seconds with the 500,000 document IIF. As the IIF

size increases, the document list for each query term becomes longer, and the time spent for IR operations increases considerably. As a result, the IR operation eventually takes more time than the disk access, and becomes the major bottleneck.

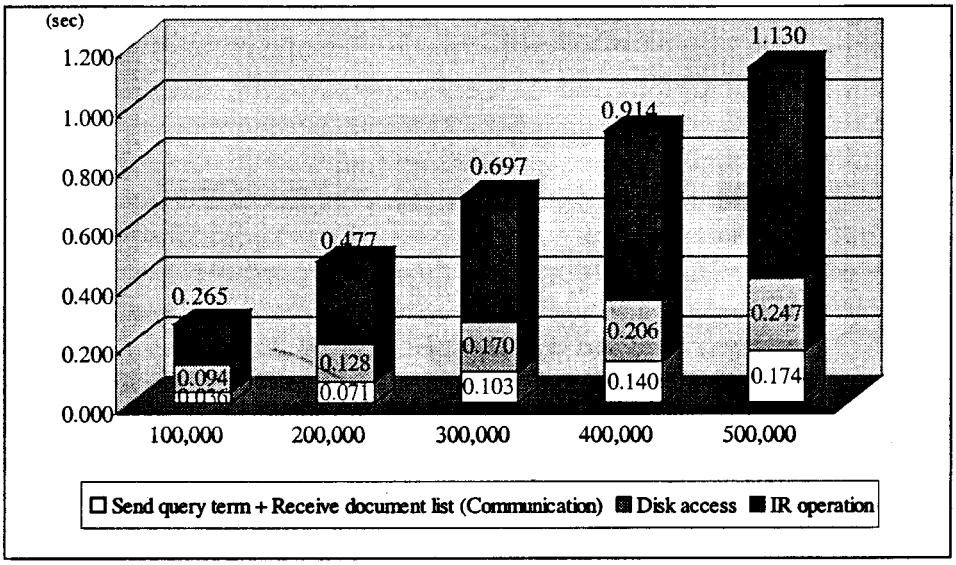

Fig. 2. IIF size vs. query processing time

4 Conclusions

In this paper, we studied the effect of the lexical and the greedy declustering methods for a parallel IR system based on a low-cost high-performance PC cluster system. The data sets used in our experiments consist of newspaper documents. In the near future, we intend to do experiments with various data sets having different characteristics and to evaluate the behavior of our parallel IR system in the presence of very short queries as those found in the Web, which modified by relevance feedback usually have many terms.

References

1. Park, S.H., Kwon, H.C.: An Improved Relevance Feedback for Korean Information Retrieval System. Proceedings of the 16th IASTED International Conference on Applied Informatics, IASTED/ACTA Press, Garmisch-Partenkirchen, Germany (1998) 65-68
2. Frakes, W., Baeza-Yates, R.: Information retrieval – data structures & algorithms. Prentice-Hall (1992)
3. Cormack, G.V., Clarke, C.L.A., Palmer, C.R., Kisman, D.I.E.: Fast Automatic Passage Ranking (MultiText Experiment for TREC-8). The proceedings of the Eighth Text Retrieval Conference (TREC-8), Gaithersburg, Maryland (1999) 735-741