International Journal of Computer Applications (0975 – 8887)
Volume 49– No.19, July 2012

Indexing in Search Engines based on Pipelining Architecture using Single Link HAC

Anuradha Tyagi
S. V. Subharti University
Haridwar Bypass Road
NH-58, Meerut, India

Khaleel Ahmad
S. V. Subharti University
Haridwar Bypass Road
NH-58, Meerut, India

ABSTRACT

Search on the web is a daily activity for many people throughout the world. Applications based on search are everywhere. We are in a data rich situation this becomes an obstacle for Information retrieval system. In this paper, we propose a pipelining architecture of indexing in order to enhance memory utilization and fast query optimization and also a soft clustering algorithm is applied to group documents into clusters hierarchically linked. Each cluster is labeled with is most relevant term known as document identifier. Such that documents within the same cluster are similar. In this way it will create hierarchy of index so that search will travel from lower level to higher level. The comparisons of the document against the user query will direct to specific clusters not the whole collection of clusters.

Keywords

Agglomerative Clustering, Pipelining, Hierarchical, Indexing.

1. INTRODUCTION

The Indexing phase of the search engine is viewed as a web content mining process [1][8][6]. Starting from the large collection of data the indexer first search the document term which we pass as a query to the search engine also track the list of all documents that contains the given term, it also stores the number of occurrences of each term inside every document. This information is stored inside the index that is represented by using Inverted file (IF) for each term t. We must store the list of all documents that contain t and identify each term t by a document ID shown as (Doc ID). We can use fixed size array for this.

Clustering is the grouping a set of objects into the classes of similar object. Document within the same cluster are similar.

Clustering of multidimensional data is required in many fields. One popular method of performing such clustering is hierarchical clustering. This method starts with a distinct points , each of which is considered to be a separate cluster. The two clusters that are closest according to some metric are agglomerated. This is repeated until all of the points belong to the one hierarchically constructed cluster. The hierarchical structure is called a dendogram; this is a tree like structure that shows which cluster is agglomerated at each step. In this way, we reduce the number of comparisons against the user query because the searching process is limited to only certain cluster not whole collection of documents.

In this paper we proposed a architecture of indexing which have the following four phases clustering, loading, process sing and storing. The output of each phase becomes the input of other.

This paper includes the bottom up agglomerative hierarchical [1, 4] clustering. In bottom up clustering take each document as a single cluster at the outset and then successively merge the pair of clusters until all clusters have been merged into single cluster containing all the documents.

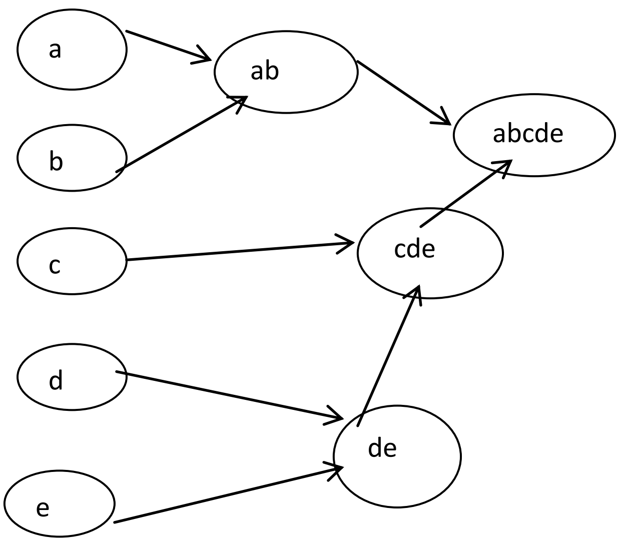


Figure1 Hierarchical Agglomerative Clustering (HAC)

Initially it have five documents a, b, c, d, e as clusters. We have to make one cluster by merging the two most similar clusters into one cluster and repeat the previous step until we achieve the desired result. Clustering is the widely adopted technique aimed at dividing the collection of data into disjoint group of homogeneous elements. Document cluster several indexing technique is adopted to make better index. So that user can get relevant pages in a very efficient way. When the user put the query it checks the indexer for matching the keywords and give the documents to the user. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document in the corpus, which would require considerable time and computing power. For example, while an index of 10,000 documents can be queried within milliseconds, a sequential scan of every word in 10,000 large documents could take hours. The additional computer storage required to store the index.

2. Related Work

A review of previous work on document clustering algorithm is given. In the field of clustering many algorithm are already given, but they are less efficient in clustering together the most similar document.

The K-means [8] [9] clustering method is one of the simplest unsupervised learning algorithms for solving the well-known clustering problem. The goal is to divide the data points in a data set into K -clusters fixed a priori such that some metric relative to the centroids of the clusters (called the fitness function) is minimized. The algorithm consists of two stages:

an initial stage and an iterative stage. The initial stage involves defining K initial centroids, one for each cluster. These centroids must be selected carefully because of differing initial centroids causes differing results. One policy for selecting the initial centroids is to place them as far as possible from each other. The second, iterative stage repeats the assignment of each data point to the nearest centroid and K new centroids are recalculated according to the new assignments. This iteration stops when a certain criterion is met. This is a effective heuristic method. The number of iterations required may be large.

Another work proposed was the Reordering algorithm [10] reordering of a collection of documents D is a bijective function _: {1, . . . ,N} ! {1, . . . ,N} that maps each doc id i into a new integer _ (i). Moreover, with _ (D) = D_(1),D_(2), . . . ,D_(N). It will indicate the *permutation* of D resulting from the application of function. This algorithm is not effective in clustering the most similar documents. The biggest document may not have similarity with any of the documents but still it is taken as the representative of the cluster.

In this technology the number of clusters is unknown. However, two documents are classified to the same cluster if the similarity between them is below a specified threshold. This threshold is defined by the user before the algorithm starts. It is easy to see that if the threshold is small; all the elements will get assigned to different clusters If the threshold is large, the elements may get assigned to just one cluster. Thus the algorithm is sensitive to specification of threshold.

This paper the proposed algorithm has to overcome the shortcomings of existing algorithm. It produces a better clustering of documents in the cluster. This algorithm selects the two clusters with minimum distance between them and merges to form a new cluster. Next it will merge with the cluster having shortest distance between them.

3. Proposed Approach

The Pipelining architecture will create the inverted index for every cluster. As a result a hierarchy of inverted indexes is created. The inverted index store all common word in the cluster at one level and produce the result for the next higher level , so at higher level it have less word in index.

The core of indexer is the index building process that executes on each cluster. The Corpus is the collection of data on the web that is needed to be indexed. During the clustering phase it will take the document from the corpus and starts clustering process by using single link agglomerative clustering. In single link clustering clusters are merged at each stage by Single shortest distance between them . During each iteration after the cluster x and cluster y are merged the distance between the new cluster say n and another cluster say z is given by dnz= min (dxz, dyz) where dnz denotes the distance between the two closest members of cluster n and z. If the cluster n and z were to be merged. Then for any object in the resulting cluster the distance to its nearest neighbor would be at most dnz.

3.1 The Pipelined Indexer Design

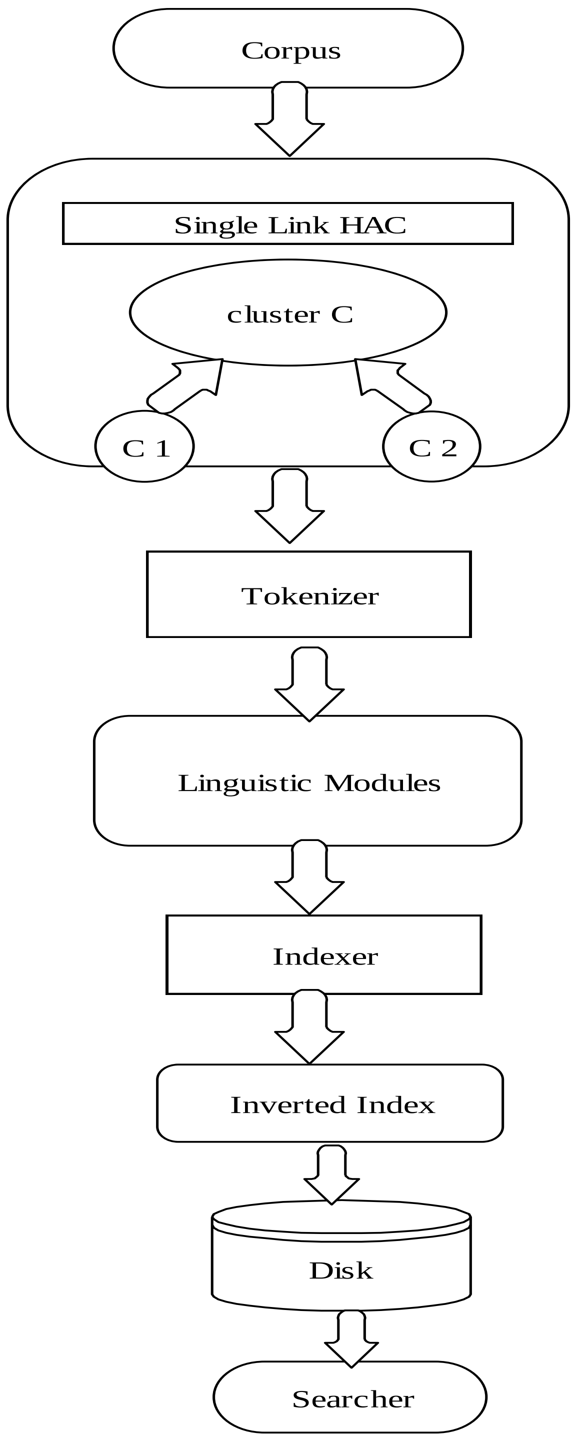


Figure 2 Pipelined Architecture of indexing

This architecture of indexing has four logical process, Clustering, Loading, Processing and Storing. During the clustering phase it will take the document for clustering. During the loading phase some number of clusters is read from the input stream. The processing phase is a combination of following processes. In first process sample document terms is passed through the Tokenizer, will split the document into tokens, it also removes the punctuation marks and spaces. During the second phase the data that is stream of tokens is passed through the linguistic modules in which tokens are modified. The Indexer will generate inverted index.

In inverted index it makes the sequence of (Modified token, Doc ID) pairs. In the third phase, the sorting phase it sort the data by removing the all duplicate entries and it also add the frequency information. This is the core index building step. The inverted index file is created for each cluster inside the hierarchy. As we have the large corpus then we could not store it into the main memory. We can store it on to the disk and then bring the data from the disk to the main memory. The searcher is used the inverted file to find the term based on the end user query. When the end user typed the query is goes to searcher and search the match the keyword from the inverted index file. Firstly it will go to the higher index where the numbers of terms are less. It will be the efficient and time saving technique for finding the documents. All four phases used the disjoint set of system resources. Therefore the indexing performance can be improved by executing all four phases concurrently. These four phases together forms a software pipeline. The indexing phase takes four reusable memory buffers and execute in four runs. That results in minimum overall running time.

3.2 Software Pipeline

The four phases together forms a software pipeline.
These four phases are Loading , Processing , Sorting and Storing. The pipelining technique optimizes the utilization of functional units by overlapping the execution of different executing the different processes. During the first phase the data stream is divided into tokens containing the token name associated with the frequency information. In the successive phases the tokens are processed and sorted according to their position in the document. The resulting array that is posting list is stored into the disk. As shown in the figure 3.

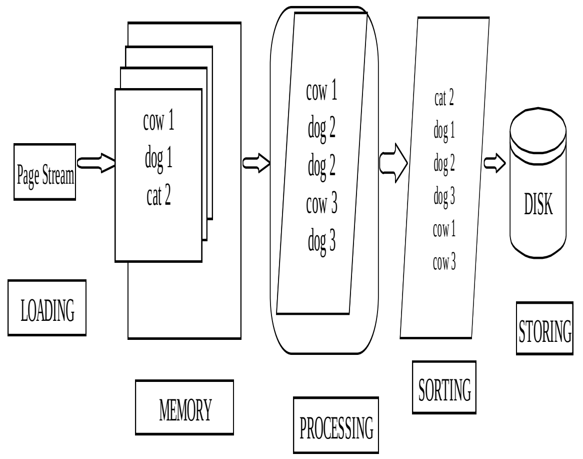


Figure 3 The four phases together forms a software pipeline

4. Clustering Algorithm

4.1 Single Link Clustering Algorithm

In Single Link Clustering Algorithm, the clusters are merged at each stage by the single shortest link between them. As shown in the figure 4. The distance between the new cluster and some other cluster is determined by the distance between the two closest members of two clusters. The single link clustering algorithm is defined as follows

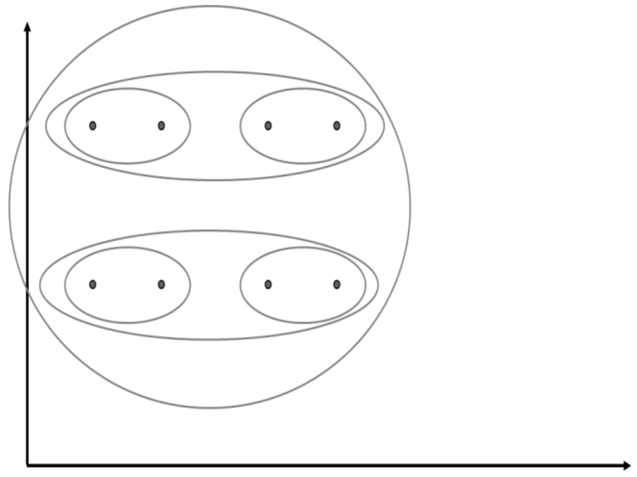


Figure 4 Single link HAC

4.1 The algorithm
The single link clustering algorithm can be defined as follows
Step: 1 Construct the distance matrix from the given pair of matrix
Step: 2 assign each pattern to a cluster
Step:3 Use maximum similarity of pairs:
Sim(Ci, Cj) = max (Sim X, Y)
xε ci, yε cj
Step:3 determine the smallest entry in the distance matrix D, and merge the two clusters say D (Ci , Cj)
Step:4 after merging Ci and Cj, the similarity of the resulting cluster to another cluster , Ck is
Sim ((Ci U Cj), Ck) = max (Sim (Ci Ck) , Sim (Cj , Ck))
Step:5 If only one cluster is left ,stop. Else go to step 3.

4.2Example Illustrating Cluster Formation

An example of showing the single link clustering on Eight patterns is given by the figure 5. A 8*8 matrix is given, as given in the algorithm we have to compute the distance matrix for eight clusters. The different iterations are given as a result of clusters are merged into one as shown

	1	2	3	4	5	6	7	8
1		101	13	60	50	42	22	52
2			133	42	200	210	130	136
3				165	18	30	15	36
4					10	8	25	11
5						**7**	40	47
6							37	43
7								12
8								

	1	2	3	4	5	5,6	7	8
1		101	13	60	50	42	22	52
2			133	42	200	210	130	136
3				165	18	30	15	36
4					10	**8**	25	11
5,6							40	47
7							37	43
8								12

	1	2	3	4,5,6	7	8
1		101	13	60	22	52
2			133	42	130	136
3				165	15	36
4,5,6					25	**11**
7						12
8						

	1	2	3	4,5,6,8	7
1		101	13	60	22
2			133	136	130
3				42	15
4,5,6,8					**12**
7					

	1	2	3	4,5,6,7,8
1		101	**13**	22
2			133	130
3				15
4,5,6,7,8				

	1,3,4,5,6,7,8	2
1,3,4,5,6,7,8		**101**
2		

Figure 5 Similarity matrices

Now according to algorithm it will take two clusters and make one on the basis of similarity. In the last Iteration 1,3,4,5,6,7,8 is the mega cluster and 2 is a alone cluster. It will put to the next level according to algorithm. Now it will create a hierarchy shown as in the figure 6.

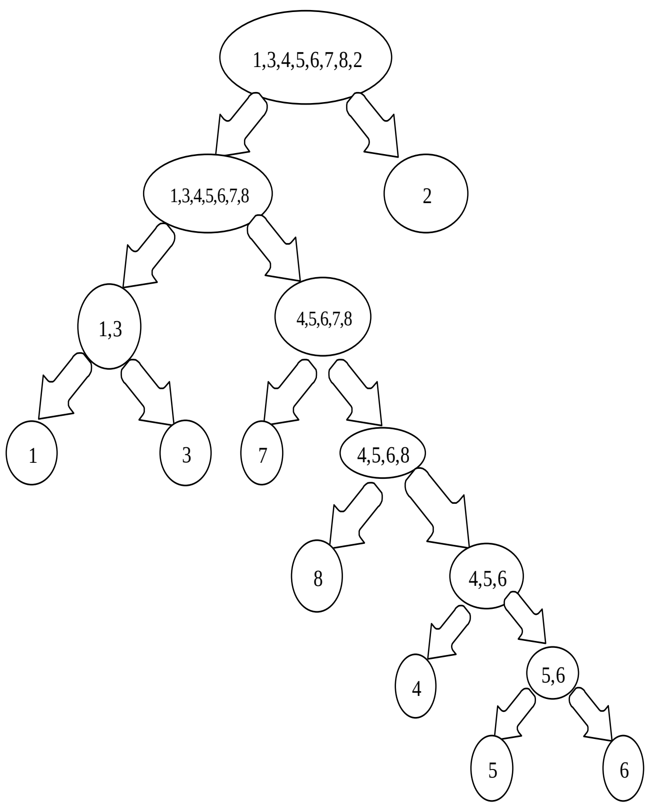


Figure 6 Hierarchies of Seven Documents

5. Performance Analysis

1-After applying the single link Hierarchical agglomerative clustering a hierarchy of documents is generated. Each cluster is has its own inverted index. So if the user generates a query it will corresponds to the lower level of cluster because at lower level the cluster have few terms in it as compare to the higher level. So it takes less time in order to execute the query.

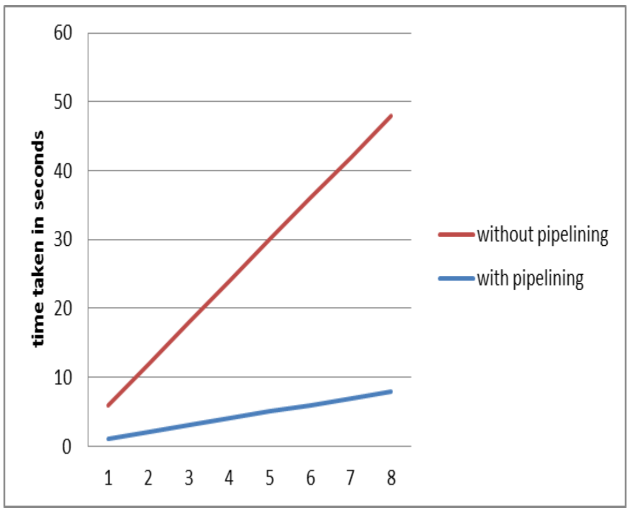
2. In single link HAC, there is no need to refine the value of k for reordering as in the case of K means, which is quite time consuming.

3. K means is a simple heuristic technique which is quite time consuming, and there is no fixed dead end to stop the iterations. It may be faster but it could not guarantee the result.

4. As it based on the pipelining architecture of indexing. it will consume less time because as one thread will do the clustering for one level another thread will create thread inverted index for the other level.

5. It also saves the space because it used to store only those terms which are common at each level so size of the inverted index will also reduce as we go up in the hierarchy.

The performance can be evaluated by the graph shown in the figure 7. The graph is plotted between the execution time and the number of documents..



Number of documents (in thousands)

Figure 7 The graph is evaluating the performance with pipeline and without pipeline

6. CONCLUSION

The proposed pipeline architecture of indexing has four logical phases clustering, loading, processing, storing. It takes less time to index the documents. This architecture is generates a hierarchy of clusters of at the same time it will take the first level of cluster and start processing and create the inverted index. Similarly for the next level it will do inverted index. When the user makes a query it will go to higher index which will have fewer words. By having the less word in index time will be saved for searching the word. For each cluster there is one inverted index. If searcher will match for other keyword which are not in higher index then it will come down to lower index. This will take less time in searching for the documents of clusters. So it is efficient in time and space saving and produce faster search.

7. FUTURE WORK

The future work can be done for making inverted index better or can apply another approach in architecture. We can also rank our results in proper way in order to reduce the time taken in searching the data .

8. REFERENCES

[1] Grossman, Frieder, Goharian. IR Basics of Inverted Index. 2002. Verified Dec 2006.

[2] Anand Rajaraman,Jeffrey D.Ullman, “Mining of massive data sets” Stanford University 2010-11

[3] J. B. MacQueen (1967): "Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability", Berkeley, University of California Press, 1:281-297

[4] Athman Bouguettaya "On Line Clustering", IEEE Transaction on Knowledge and Data Engineering Volume 8, No. 2, April 1996.

[5] Sergey Melnik, Sriram Raghavan, Beverly Yang, and Hector Garcia-Molina. Building a Distributed Full–Text Index for the Web. In World Wide Web, pages 396–406,2001

[6] Parul Gupta, Dr. A.K. Sharma, Hierarchical Clustering based Indexing in Search Engines, communicated to International Journal of Information and Communication Technology.

[7] Sanjiv K. Bhatia. “Adaptive K-Means Clustering” American Association for Artificial Intelligence, 2004..

[8] Oren zamir’’Web document Clustering’’department of Computer science and engineering,University of Washington.2007

[9] Yi Zhang and Tao li ‘’A Framework for evaluating and organizing document clustering through visualization’’ published by ACM, USA.