International Journal of Computing Academic Research (IJCAR)
ISSN 2305-9184 Volume 4, Number 6(2015), pp.270-277
© MEACSE Publications
http://www.meacse.org/ijcar

An Improved Indexing Mechanism Based On Homonym Using Hierarchical Clustering in Search Engine[*]

Varsha Rathi and Neha Bansal
Department of computer science &Engineering BSAITM, Faridabad, Haryana,India

Abstract

Now-a-days, the World Wide Web is the collection of large amount of information which is increasing day by day. For this increasing amount of information, there is a need for efficient and effective index structure. Indexing in search engines has become the major issue for improving the performance of Web search engines, so that the most relevant web documents are retrieved in minimum possible time. For this a new indexing mechanism in search engine is proposed which is based on indexing the homonym terms of the web documents, a homonym term have multiple meaning which gives different context of the web documents. The indexing is performed using hierarchical clustering method which clustered the similar context documents into the same cluster and these clusters are clubbed together to form mega cluster on the basis of homonym term similarity of clusters. It will optimize the search process by forming the different levels of hierarchy. This will give fast and relevant retrieval of web documents to the user.

Keywords: Context, Homonym, Hierarchical Clustering, Indexing, Ontology Repository.

Introduction

With the fast growing internet, the World Wide Web (WWW) has become one of the most important resources for obtaining information. Currently, there are huge amounts of documents existing in the World Wide Web and finding information from WWW according the user interest becomes a critical task. The main aim of the search engine is to provide most relevant documents to the users in minimum possible time. To improve the search results according to the user's query indexing is performed on the web pages. So granting efficient and fast accesses to the index is a major issue for the performances of web search engines. The proposed approach is used in which context and homonym based hierarchical clustering [7] of web documents is performed. The purpose is to cluster or clubbed the web documents based on the context similarity of the documents. The aim of clustering based indexing is to assign the context similar documents within the same cluster. Further the hierarchical clustering is applied in which similar clusters are clubbed to form a mega cluster and similar mega cluster are combined to form super cluster on basis of homonym similarity. This will provide the user the best possible matching results in minimum possible time by directing the search process to a specific path form higher levels of clustering to the lower levels. In proposed approach, the indexing of documents is performed on the basis of context of the documents rather than on the basis of terms. The context of the documents is extracted from the ontology repository. The architecture for the proposed work is represented in Fig. 1. According to the architecture, firstly the web pages are gathered by crawler and are stored in repository of web pages. After this the preprocessing on documents is performed by indexer for extracting the keywords with their frequency count in documents

[*] **The paper was first presented in International Conference on Recent Trends in Computing and Information Technology organized by B.S Anangpuria Institute of Technology and Management, Faridabad, India.**

with their corresponding doc Ids. Now, the keywords whose frequency count is greater than threshold value are extracted and the different contexts of terms are gathered from thesaurus, which are maintained in ontology [1] repository for deducing the context of the documents. Finally, the clustering is performed.

Related Work

In this section, a review of previous work on index organization is given. In this field of index organization and maintenance, many algorithms and techniques have already been proposed but they seem to be less efficient in efficiently accessing the index.

F. Silvestri, R.Perego and Orlando [2] proposed the reordering algorithm which partitions the set of documents into k ordered clusters on the basis of similarity measure where the biggest document is selected as centroid of the first cluster and n/k1 most similar documents are assigned to this cluster. Then the biggest document is selected and the same process repeats. The process keeps on repeating until all the k clusters are formed and each cluster gets completed with n/k documents. This algorithm is not effective in clustering the most similar documents.

Oren Zamir and Oren Etzioni [3] proposed threshold based clustering algorithm. Initially, the number of clusters is unknown, based on the specified threshold value the similarity between the two documents is evaluated and they are classified to the same cluster. 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.

C. Zhou, W. Ding and Na Yang [4], the paper introduces a double indexing mechanism for search engines based on campus Net. The CNSE consists of crawl machine, Chinese automatic segmentation, index and search machine. The proposed mechanism has document index as well as word index. The document index is based on, where the documents do the clustering, and ordered by the position in each document. During the retrieval, the search engine first gets the document id of the word in the word index, and then goes to the position of corresponding word in the document index. Because in the document index, the word in the same document is adjacent, the search engine directly compares the largest word matching assembly with the sentence that users submit. The mechanism proposed, seems to be time consuming as the index exists at two levels.

N. Chauhan and A. K. Sharma [5] proposed, the context driven focused crawler (CDFC) that searches and downloads only highly relevant web pages, thus, reducing the network traffic. A category tree has been used, which provides flexibility to the user for interacting with the system showing the broad categories of the topics on the web. The proposed design significantly reduces the storage space at the search engine side.

P. Gupta and A. K. Sharma [6], worked on context based indexing in search engines using ontology. The index construction is done on the basis of the context using ontology. The context repository, thesaurus and ontology repository are used by the indexer to identify the context of the document.

Proposed Work

This paper proposes an algorithm for indexing the web documents on the basis of homonym terms and context of the documents using hierarchical clustering in search engines. The proposed indexing mechanism will group the same context documents into the same clusters and further these clusters are clubbed into mega clusters on the basis of homonym similarity. The proposed architecture of indexing is shown in Fig. 1.

Description of Various Components. The proposed architecture of indexing in search engine consists of the following functional components.

1) **Crawler.** It is an important component of web search engines, where they are used to collect the corpus of web pages indexed by the search engine.
2) **Repository of Web pages.** This is the collection of web documents that have been collected by the crawler from the WWW. It is a database which stores the web pages that are gathered by the crawler from WWW in order to provide web documents for indexing.
3) **Preprocessing of Documents.** It involves the tokenization phase which splits sentences into individual tokens, typically words. It will simply segregate all the words, numbers, and their characters etc. from given document. It also includes stop word removal phase and stemming process which remove the keywords those occur frequently in the web page but do not contribute to the context of web document. The Stemming phase is used to extract the sub-part i.e. called as stem/root of a given word. For example, the words continue, continuously, continued all can be rooted to the word continue. This will reduce the size of indexing file.
4) **Thesaurus**. It is a dictionary of words available on the World Wide Web from thesaurus.com which contains the words as well as their multiple meanings. Using thesaurus the multiple meaning of the terms and various contexts can be derived.
5) **Ontology Repository.** After the extraction of the keywords from the documents, and extracting the multiple context of the keywords from the thesaurus, this task is further extended by forming their structural framework which would represent the relationship and thus the semantic meaning of the document, and such representations are referred as ‘Ontologies’. Ontology repository is a database which contains the various concepts with their relationships.
6) **Context of the Document.** The context of the document deduce from ontology represent the semantic or theme of the document. At this, level the different documents retrieved for the same term are categorized according to the context. The document context has been extracted using thesaurus and ontology repository.
7) **Hierarchical Clustering based Indexing.** The final index is constructed using hierarchical clustering based indexing on the basis of both context and as well as the homonym term of the documents. In hierarchical clustering based indexing, initially the clustering of documents is done on the basis of context similarity and keyword similarity of the documents, which is calculated using the similarity measure. Further the mega clusters out of the similar clusters are generated on the basis of similarity between the homonym terms of the clusters.
8) **Searcher.** It is that module of the search engine that receives user queries via the user interface and hence after searching the results in the index provides them to the user.
9) **Query Interface.** It is that user interface through which user types the query.

Proposed Algorithm for Indexing. The algorithm depicted in Fig.2 shows the various steps in the construction of the homonym term based index.

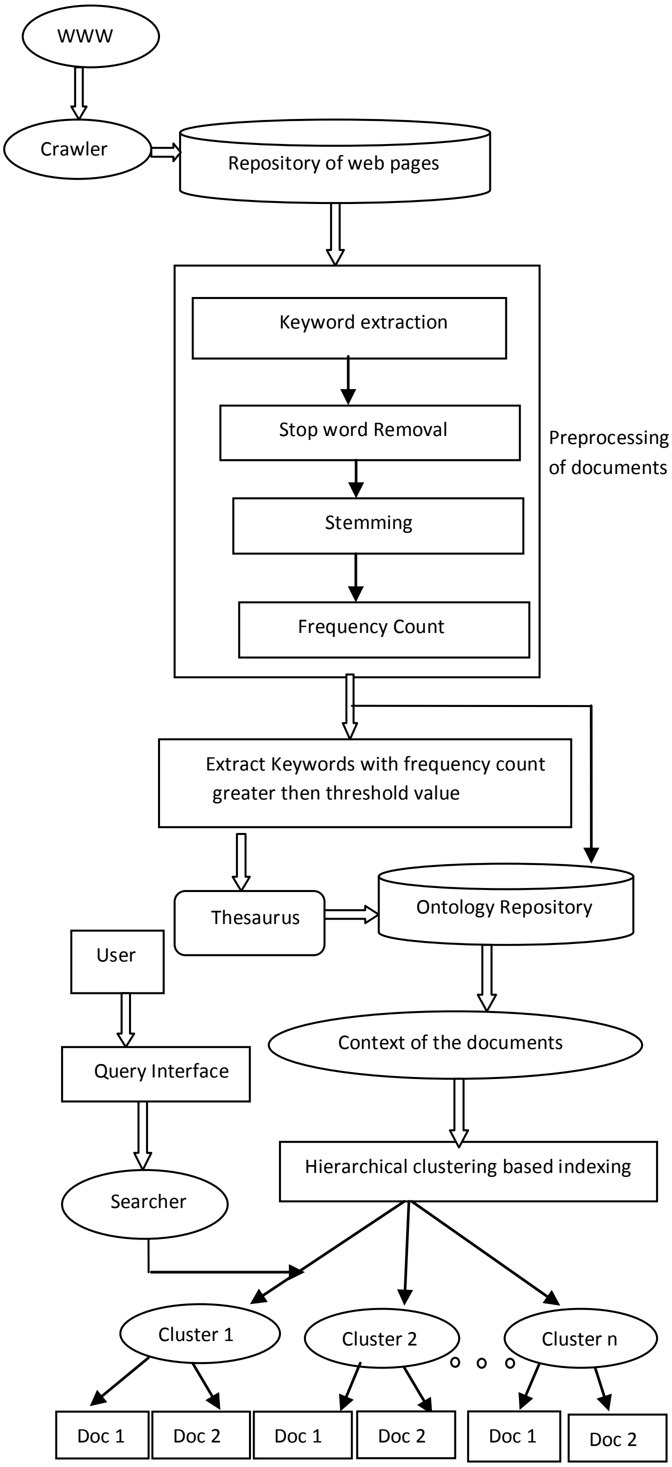

Fig. 1 Proposed Architecture of Indexing

1. The web pages are crawled by the web crawler from WWW and are gathered into the web page repository.
2. In this step the preprocessing of the documents is performed by indexer which includes the fetching of documents from repository with their corresponding doc ids, keyword extraction phase, stop word removal, stemming and frequency count.
3. Once the document preprocessing is complete, the keywords with frequency count greater than threshold value are extracted.
4. Now the keywords that are selected in previous step are searched in the thesaurus for extracting the multiple meanings of the terms (thesaurus can be taken online from thesaurus.com).This step will help in generating multiple context of the keywords.
5. In this step the multiple contexts and the terms of the document are compared with the ontology repository. The context of the document is extracted by matching the keywords of the documents and the multiple contexts with the concepts and the relationship terms in the ontology repository.
6. Now the hierarchical clustering based indexing of documents is performed on the basis of context similarity of the documents and homonym similarity of clusters by calling following algorithms.
   a) Call algorithm Document_Similarity.
   b) Call algorithm Document_Clustering.
   c) Call algorithm Mega_Clustering.
7. The homonym term along with the mega cluster Id are indexed into the posting list by the previous step. The posting list consists of three columns, one containing the homonym term, second containing mega cluster Id and third containing cluster Id. At this step final indexing is performed on the basis of both context as well as the homonym term.
8. When the user fires a query, then the index is being searched for the homonym term.
9. After matching the homonym term the mega cluster Id is fetched this gives the corresponding cluster Id of the clusters containing the relevant documents according to the different context of the terms.
10. Now the user can fetch clusters (giving the relevant documents within them) according to the context which the user desires.

Fig.2 Proposed Algorithm for Construction of Homonym based Index

Algorithm for Computing Similarity Matrix. Let D={D1, D2,……Dn) be the collection of N textual documents being crawled to which consecutive integers document identifiers 1…n are assigned. Each document Di can be represented by a corresponding set Ki such that Ki is a set of all the keywords extracted from Di. Let us denote that set by D* such that D*={K1,K2 ,……….. Kn}. The similarity of any two documents Ki and Kj can be computed using the similarity measure:

Similarity_measure (Ki, Kj) = |Ki ∩ Kj | / |Ki U Kj |

If the context of the document Di and Dj are same then the similarity measure is calculated as

Similarity_measure (Ki, Kj) = |Ki ∩ Kj | / |Ki U Kj | +1

The algorithm that calculates the similarity of each document with every other document using similarity_measure given above is given in Fig. 3.

```
Algorithm Document_Similarity

for i=1 to n
begin
   sim[i][j]=0;
for j=i+1 to n
begin
       if(context.$D_i$= =context.$D_j$)
then
      sim[i][j]=similarity_measure($K_i$,$K_j$) +1;
       sim[j][i]=sim[i][j];
else
       sim[i][j]=similarity_measure($K_i$,$K_j$);
       sim[j][i]=sim[i][j];
end for
end for
```

Fig. 3 Algorithm for Computing Similarity Matrix

Algorithm for Document Clustering. The clustering algorithm which clusters together the context similar documents is given in Fig. 4.

Algorithm for Mega Clustering. The mega clustering algorithm which forms the mega clusters by clustering together the homonym similar clusters is given in Fig. 5.

```
Algorithm  Document_Clustering

i=1
for f=1 to c                          //for number of clusters
begin
cs=0                                  // initially cluster is empty
for e=1 to n/c
  begin
      for j=1 to n
           Select max from sim[i][j]
          if(context.$D_i$= =context.$D_j$)
                    cs = cs U $K_i$
                    D*=D*-$K_i$
             for p=1 to n
                     begin
                            sim[i][p]=0;
                           sim[p][i]=0;
                 end
         i=j
end
end
```

Fig. 4 Algorithm for Clustering

```
Algorithm  Mega_Clustering

i=1
      for f=1 to m                          //number of mega cluster
        begin
           MC=0
           for e=1 to n/m
           begin
             for j=2 to n
                  MC=Ci
                    CS=CS-Ci
               if (homonym.Ci= =homonym.Cj)
                    MC=MC U Cj
                      CS=CS-Cj
             else
                   Cj is not added to the cluster;
                      end
              i=j
end
end
```

Fig. 5 Algorithm for Mega_Clustering

Example

When user types a query with keyword ‘Spider’ which is a homonym term having different context. Search engine will show irrelevant documents which are not according to the user’s interest as the ‘Spider’ word has different context in different fetched documents. For providing relevant documents to the user the proposed approach is used which give relevant document having different context in different clusters.

The following steps are performed by proposed approach:

After performing all the preprocessing steps the keywords are extracted with their frequency count and doc Ids as shown in Table 1.

Table 1 Extracted Keywords from Documents after Preprocessing

KEYWORD	FREQUENCY COUNT	DOC ID
Spider	8	1,2,3,4,5,6,8,9 ,11
Arthropod	2	1,9
8 legs	3	1,4,9
Segment	2	1,4
Body	2	1,4
Web	3	2,6,11
Crawler	3	2,4,6,9,11
Search	2	2,6
Engine	3	2,6
Play	3	3,7
Game	2	3,5
Excellent	1	3
Graphics	1	3,5
Online	2	3
Creepy	1	4,9
Rank	2	4
Species	1	9
Solitaire	1	5
Download	1	6
Internet bot	1	6
Store	1	6
Bat	5	8,7,10,12
Wood	1	7,10
Sports	2	7,10
Hit	1	7
Wings	2	8,12
Flying rat	2	8,12

Table 2 Extracted Context and Homonym Term for documents

Keywords Set	Doc Id	Context	Homonym
{Spider, arthropods, 8legs,segment ,body}	1	Insect	Spider
{Spider, web, crawler, search, www, engine}	2	Computer Program	Spider
{Spider, play, game, excellent, graphics, online}	3	Online game	Spider
{spider, creepy, crawler, 8legs, segment, body, rank, species}	4	Insect	Spider
{Spider, solitaire, game, graphics}	5	Online Game	Spider
{spider, search, www, download, internet bot, web, crawler, store}	6	Computer Software	Spider
{Bat, wood, sports, hit, ball, play}	7	Solid Object	Bat
{Bat, wings, black, flying rat, mammal}	8	Animal	Bat
{Spider, arthropod, creepy, crawler, 8legs, body, species}	9	Insect	Spider
{Bat, Sports, bop, hit, ball}	10	Solid Object	Bat
{Spider, web, crawler, search, engine, www}	11	Computer Program	Spider
{Bat, mammals, black, color, eco, flying rat, wings}	12	Animal	Bat

Table 3 Clustering Of Documents

Context	Homonym	Cluster id	Document id
Insect	Spider	C_1	1,9,4
Computer Program	Spider	C_2	11,2,6
Online game	Spider	C_3	5,3
Solid Object	Bat	C_4	7,10
Animal	Bat	C_5	8,12

Now extract the keyword having frequency count greater than threshold value=3.Only keyword Spider and Bat has frequency count greater than 3. Both keywords will select and their multiple contexts and all other information will be extracted from thesaurus and maintained in the ontology repository in the form frames. Extracting the context of the documents using ontology with the homonym term, following table will generated as shown in Table 2.

The similarity among the documents is computed on the basis of context similarity measure, and similarity matrix is constructed. According to the clustering algorithm documents are clustered to form clusters as shown in Fig.6.

1st cluster will have document 1, then 9 and lastly 4 (all the documents have the same context Insect).

2nd cluster will have document 11, then 2 and lastly 6 (all the documents have the same context Computer program).

3rd cluster will have document 5, then 3 (all the documents have the same context online game).

4th cluster will have document 7, then 10(all the documents have the same context solid object).

5th cluster will have document 8, then 12(all the documents have the same context animal).

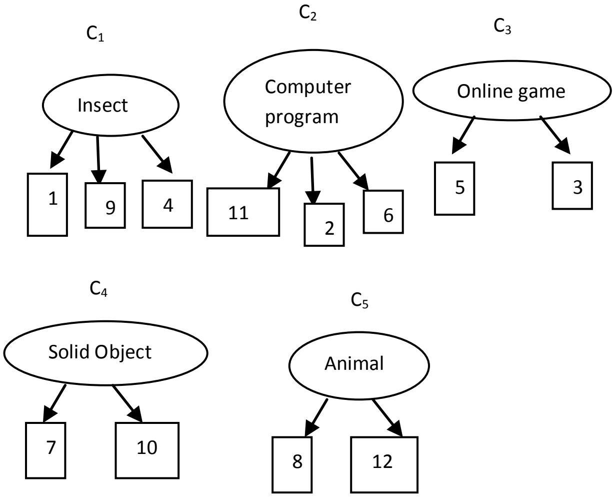

Fig. 6. Cluster Formations of Documents

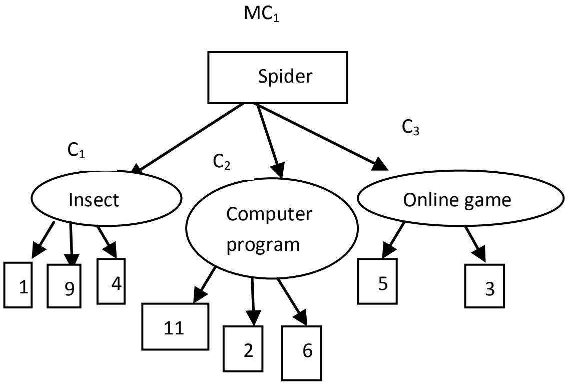

Fig. 7. Mega Cluster Formations of Clusters

Formation of mega clusters by clubbing the clusters on the homonym similarity by using the mega clustering algorithm. The homonym similar clusters are combines into same mega clusters as shown in Fig.7.

1st mega cluster (MC1) will have C1, then C2, and last C3 (having same homonym term Spider).

2nd mega cluster (MC2) will have C4, then C5 (having same homonym term Bat).

Finally the index is constructed on the basis of homonym term and context of the document as shown in Table 4.

Table 4 Final Index on the basis of Homonym

Homonym	**Mega Cluster Id**	**Cluster Id**
Spider	MC_1	C_1 ,C_2 ,C_3
Bat	MC_2	C_4 ,C_5

Conclusion

Homonym term indexing will help the user to extract all the relevant documents which contain every context, and user can choose the document cluster according to the context he/she wants. In this way all the different context documents are fetched as a group. So the problem of getting irrelevant results for the query using the term having multiple context is reduce to some extent.

As the hierarchical clustering based indexing is used it will optimize the search process by forming different level of hierarchy. Also improve the search time as the search get directed to a specific path from high level cluster to the lower level clusters and finally to the individual documents. It also reduces the index size as similar documents are assigned closer ids. Aids into fast retrieval of relevant documents as similar documents get clustered together into the same cluster, the specific query relevant documents can be rapidly picked from that cluster. Overall the proposed work will provide better results.

References

[1]Chandrasekaran, Josephson, Benjamins. "What are Ontologies and why do we need them". IEEE Intelligent Systems, Jan/Feb 1999.

[2]Fabrizio Silvestri, Raffaele Perego and Salvatore Orlando "Assigning Document Identifiers to Enhance Compressibility of Web Search Engines Indexes". Proceedings of SAC, 2004.

[3]Oren Zamir and Oren Etzioni "Web Document Clustering: A feasibility demonstration". Proceedings of SIGIR, 1998.

[4] Changshang Zhou, Wei Ding and Na Yang, "Double Indexing Mechanism of Search Engine based on Campus Net", Proceedings of the 2006 IEEE Asia-Pacific Conference on Services Computing (APSCC'06), 2006.

[5] Naresh Chauhan and A. K. Sharma," Design of an Agent Based Context Driven Focused Crawler",BVICAM'S International Journal of Information Technology, pp 61-66, 2008.

[6] Parul Gupta and A.K.Sharma," Context based Indexing in Search Engines using Ontology", International Journal of Computer Applications, Volume 1 No. 14, pp 49-52, 2010.

[7] Parul Gupta and A.K. Sharma "A Framework for Hierarchical Clustering Based Indexing in Search Engines" BVICAM's International Journal of Information Technology(BIJIT) Bharati Vidyapeeth's Institute of Computer Applications and Management(BVICAM), New Delhi.

[8] Pooja Mudgil, A. K. Sharma, Pooja Gupta,"An Improved Indexing Mechanism to Index Web Documents" 2013 5th International Conference on Computational Intelligence and Communication Networks.

[9] Anchal Jain, Nidhi Tyagi "Context Based Web Indexing for Semantic Web" IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, Volume12, Issue 4 (Jul. - Aug. 2013), PP 89-93 .

[10]Nidhi tyagi, Rahul Rishi, R.P. Agarwal "Context based Web Indexing for Storage of Relevant Web Pages" International Journal of Computer Applications Volume 40– No.3, February 2012.