A New Context Oriented Synonym Based Searching Technique for Digital Collection

M Thangaraj, *Member, IACSIT,* V Gayathri

***Abstract*—Each year sees the introduction of new digital libraries promoted as valuable resources for education and other needs. Yet systematic evaluation of the implementation and efficacy of these digital library systems is often lacking, due to the traditional keyword based search. Not only the keywords, but the synonym of it also plays an important role in the searching era. In order to overcome the issues of the traditional search, this paper suggests an effective technique, NCOSBS (New Context – Oriented Synonym Based Search). Our proposal is illustrated by showing improvements with respect to existing technique, and providing ground for further research and discussion.**

***Index Terms*— B$^+$-Tree; Digital Collection; Hash Table; Synonym–Based Search;**

I. INTRODUCTION

With the tremendous growth of the Web, information "Big Bang" has been taken place on the Internet [2]. Search Engines have become one of the most helpful tools for obtaining useful information from the "Big Bang". Increasing growth of information volume causes an increasing need to develop automatic methods for retrieval of documents and ranking them according to their relevance to the query.

Similarly digital libraries offer diverse information resources in digital format. Traditionally, libraries have been warehouse of knowledge providing information services to the users. Even after the technological intrusion, this basic function remains the corner stone of the library structure. However, the ways and means by which the information is organized, processed, stored, retrieved and filtered have undergone tremendous changes to bring in new technologies and devices.

Digital libraries provide instant access to all information, for all sectors of society, from anywhere in the world. Dr.S R Ranganathan [1], the father of library science has rightly pointed out "Right information to the right user at the right time in the right form". It is observed that the features of digital library seem to reflect the vision of Dr.S.R.Ranganathan.

Any given query may fetch huge number of results. It is obvious that very few results are relevant to the user needs out of the huge set of results even though they contain the keyword. Thus, we need an effective searching technique in digital collection, to produce the best result.

The main problem here is, the relation between the terms in the given query [3] [9] i.e., the meaning of the entire query is missing. Thus it is needed to consider the query as the contexts instead of considering just as keywords. This paper suggests a technique named New Context Oriented Synonyms Based Search (NCOSBS) to tackle the above summarized problem.

The remainder of the paper is organized as follows: Section II is devoted to the issues relevant to searching. In Section III, we describe the architecture for NCOSBS. Section IV shows our performance evaluation result. Finally, in Section V we present conclusion.

II. RELATED WORK

Generally search systems are based on the importance of the papers and / or the existence of the keywords. They do not give much importance for the context.

For checking the existence of the keyword, similarity techniques like Text-based similarity [8], Google based similarity [7] [6] is used. Even though there are many techniques are available, still the end users are struggling to get the desired information.

Because, in a keyword – based search, the main ambiguity is that, a single word may have different meanings, where as different words may also refer to the same thing. Thus we need to search by considering the context of the given query.

In [4], it suggests some structures used for searching like Trie, Composite tree, and bi-partite graph. The main problem with these structures is, time consuming and needs more number of comparisons. To overcome these issues, NCBS[5] uses B+-tree as a searching structure.

Here, contexts are extracted from the documents in the corpus using pattern extraction based techniques. All the documents in the corpus are classified based on the context regardless of the query and are mapped into the NCBS structure (combination of B+-tree and Inverted List).

When a query is given, the relevant context is identified and returned with its synonym as well as the appropriate document of the context. The main drawback of this method is it can search only with the context, not with its synonyms. It will just return the list of synonyms. Thus our improved version NCOSBS will search in both Context and its synonyms.

Manuscript received March 28, 2011.

Dr.M Thangaraj is now Associate Professor, in Department of Computer Science, Madurai Kamaraj University, Madurai, TN, India (e-mail: thangarajmku@yahoo.com).

V Gayathri is with the Department of Computer Science, Madurai Kamaraj University, Madurai, TN, India (e-mail: gayathrivengatmku@yahoo.com).

III. Proposed Work

In this work, we propose a technique named NCOSBS, which effectively retrieves the relevant document from the collection using both context and synonym. It has two major segments such as pre–processing and the query evaluation. The following process is carried out in the pre-processing phase:

Constructing NCOSBS Structure: Patterns are extracted from the digital collection to organize the documents. Documents in the digital collection, which are matched to the extracted patterns, are mapped to the NCOSBS structure.

Note that pattern extraction and construction of NCOSBS structure are pre-executed and not query dependent. The following steps are used in the query evaluation:

a) Processing the query: The query pattern is extracted from the given input text.

b) Identification of Context / Synonyms: The context / synonyms relevant to the query pattern is extracted using the NCOSBS structure.

c) Retrieving Relevant Documents: The document(s) relevant to the context / synonym is retrieved.

The NCOSBS architecture is given in Figure 1. The data set are nothing but the information in the digital collection pertaining to multiple topics. Using the pre-processing phase, the documents in the digital collection are classified based on the context extracted (using pattern extraction technique).

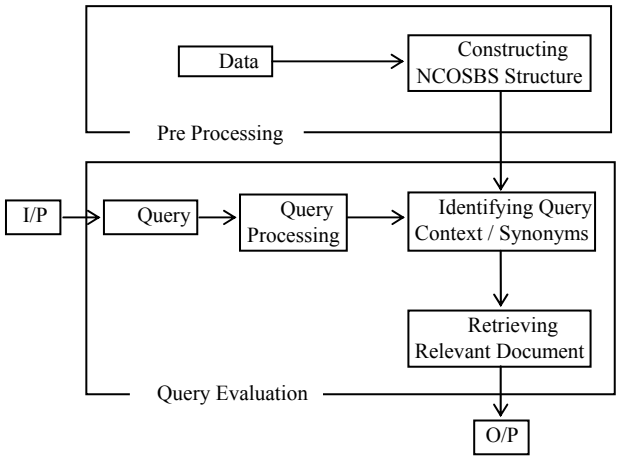

Figure1. NCOSBS Architecture

When a query is specified, the relevant pattern is extracted from the collection of patterns. Pattern matching is performed in the searching structure with the extracted query pattern. The relevant context / synonym is identified and the respective document is returned.

A. Constructing NCOSBS Structure

The digital collections are classified into contexts using pattern extraction based techniques (similar to NCBS). Then the classified contexts are mapped to our NCOSBS structure.

NCOSBS structure is a combination of B+-tree and Hash table. The context B+-tree is constructed based on the classified context with its prefix and suffix terms (<prefix><context><suffix>). The leaf node contains the Context, its synonyms and a pointer to its relevant document.

The structure of NCOSBS is shown in Figure 2. In Hash Table, each bucket is allotted for a set of alphabets, and it is filled by the alphabetically ordered synonyms of all contexts. Synonyms are retrieved from the Leaf of the Context tree and filled in the synonyms structure.

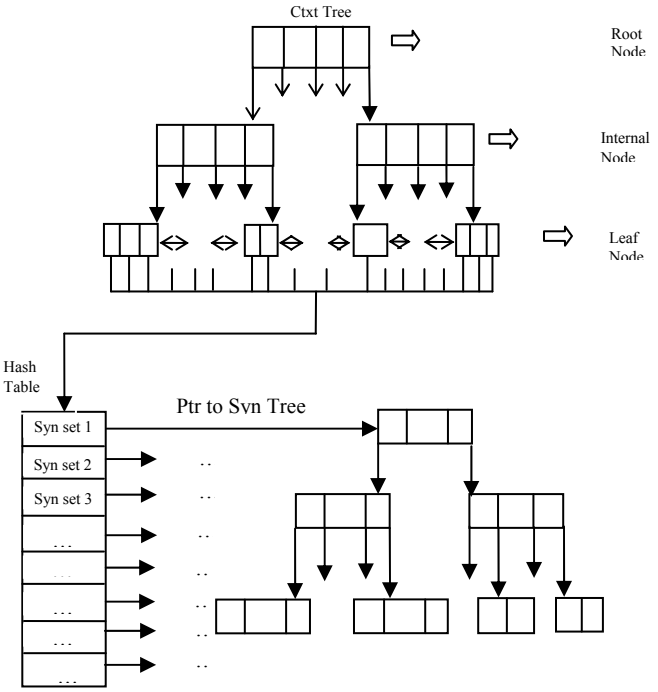

Figure 2. Overview of NCOSBS structure

Thus each bucket contains the set of synonyms and a pointer to the respective synonym B^+-tree. The synonyms tree is also constructed based on the synonym with its prefix and suffix terms (<prefix><context><suffix>). The leaf node contains the following information: synonym details, context to which it belongs to and a pointer to its respective document.

B. Identification of Context / Synonym

NCOSBS structure is searched against the given query context. Initially the query context is searched in the Context tree of NCOSBS (Similar to our earlier model NCBS). If it is found in the Context tree, then its relevant document is returned.

If the given query context is not found in the Context tree, then it is searched by the starting letter with the synonyms available in the hash table. Once the appropriate bucket of the hash table is traced, then further comparisons are done in its respective Synonyms tree.

When the user needs to search for the related synonyms of the given query, it is easier to trace from the NCOSBS structure. Once the respective context (in case of the given query is synonym, find its respective context) is identified, the synonyms list of that context is identified from the Context tree and returned.

When searching is stopped without getting the exact context or its related synonyms document as in the query, then the Context/Synonym, up to which the search mechanism found its match with the query context, is returned as a result. When there is no match occurs, then the Contexts in the root of the Context tree are returned as a suggestion for the user's reference. Instead of getting out with empty result set, the user can get some information to

make improvement in their searching query task. The user can insert the same query into our database by giving some additional information.

```
Algorithm        : Context Identification
Data Structure   : See Fig 2.
Input            : Query q, Node nodepointer
Output           : Context with its document as well as
                   related synonyms
Let i, 0 ≤ i < degree of the Context Tree;
j, 0 ≤ j ≤ 2 (j – each tuple in the query);
x – nodeof Context Tree;
doc-ptr– Pointer to the relevant document of the
Context;
<Pi> - Prefix tuple of the ith node segment of the
       nodepointer;
<Ci> - Context tuple of the ith node segment of the
       nodepointer;
<Si> - Suffix tuple of the ith node segment of the
       nodepointer;
find (Query q, Node nodepointer)
{
        if (nodepointer is NULL)   return (NULL);
        if (nodepointer is a leaf) {
          x= find (q, nodepointer);
          return(x, doc-ptr, Synonyms ); \\x is the
        context
          }
        else {
          for each tuple <qj> in the query context
            for  i = 0 to degree-1{
            if (<qj> = <Ci>)  {
              x = find (qj+1, nodepointer -> childi);
              if (x = NULL)
                return(current Context from Tree
        Leaf);
              return (x);
              }
            else if (<qj> = <Pi>)
              return (find (<qj+1>, nodepointer));
            }
\\ if qj is not available in any of the node segments in
\\ nodepointer, then, search for it in the last subtree
          x= find(<qj>, nodepointer -> childj);
          if(x!=NULL) return (x);
          }
        x=findHash(q);
        if(x!=NULL)
          return (x);
        return headnode;
        }
```

In Context Identification algorithm, each query tuple is compared with the context tuple of the node segment. If match found then, subsequent tuples are searched in its sub tree. If no match found then, it is compared with the prefix tuple. When the query tuple is found at the prefix, then the consequent tuples are compared with the context tuple of the same node segment. If the first tuple of the query is not found, then the process is continued with the next tuple.

When no match found in both prefix and context tuples then, it is searched in the last (i+1) sub tree. In case no query tuple is found in the Context tree, then the same query is searched against the synonyms structure by invoking the Synonym Identification algorithm. If the query elements are not available even in the collection of synonyms then, the Contexts in the root node are returned as a result to the user.

```
Algorithm        : Synonym Identification
Data Structure   : See Fig 2.
Input            : Query q <qp, qc, qs>
Output           : Synonym with its document as well as
                   relevant Context
Let i, 0 ≤ i ≤ 2 (i – each tuple in the query);
findHash (Query q)
{
        for each query tuple<qi> in the query context
        {
          Compute hash value hv for <qi>;
          Identify SynNodePointer of hv;
          x = findSyn (q, SynNodePointer);
          if (x = NULL)
          continue; \\continues with next iteration (i++)
          return (x);
          }
        return (NULL);
          }

Input            : Query q <qp, qc, qs>
                   SynNode SynNodePointer
Output           : Relevant Synonym Node
Let d be the degree of the Synonyms Tree,
i, 0≤ i ≤ d
findSyn (Query q, SynNode SynNodePointer)
{
        if (SynNodePointer is NULL)
          return (NULL);
        if (SynNodePointer is a leaf)  {
          if (SynNodePointer.Synonym = q)
            return (SynNodePointer);
          }
        else {
          if (<qc> <  SynNodePointer.Ci)
            findSyn (q, SynNodePointer.childi);
          }
        return (NULL);
        }
```

In Synonym Identification algorithm, query is taken as an input. The hash value of the query tuple is computed by which the relevant bucket is identified. The query is now searched against the respective Synonyms tree, using the findSyn function. If the query tuple is not found, then the same process is continued for the next tuple.

In findSyn function, context tuple of query is searched against the elements in the synonyms tree. If it is found then the rest of the query is also compared and the respective synonym node is returned.

IV. Performance Analysis

NCOSBS approach has been implemented using C++. The set of experiment explores the effects of data and query parameters on performance. All experiments reported in this section are done on Pentium Duo 1.60 GHz with 1015MB RAM and 150 GB of secondary storage, running Windows Vista. In order to test the model and to find the performance, about 5000 documents have been created and various queries have been implemented.

When compared to existing methods and techniques, this model requires a quite large storage space. Since storage space is now less expensive, the effectiveness of the access structure need to be concentrated.

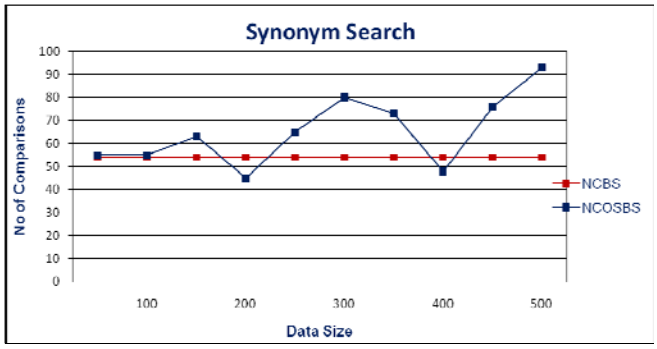

Figure 3. Synonym Search Comparisons

Figure 3 shows the amount of comparisons needed for retrieving the synonym. As NCBS does not have the facility of searching the synonyms, it will search the whole structure and finally return empty result. But our NCOSBS structure will return the exact synonym in less number of comparisons.

V. Conclusion

In this paper, we have proposed a model NCOSBS and analyzed the major issues related to the operations of the digital content. The results of this structure show the effectiveness of the contexts as well as the importance of the searching mechanism. This work can be extended further by incorporating additional features like learning mechanism, automatic query completion.

Acknowledgment

The authors wish to thank the IACSIT for giving an opportunity to publish in this journal and also thank the anonymous reviewers of ICMLC for their significant comments.

References

[1] S Abideen, Srivathsan, "Convergence of Digital Libraries with Knowledgement," Proc. Of ICDL, India, 2004.

[2] http://news.netcraft.com/archieves/web_server_survey.html - Information Explosion.

[3] Fabrizio Lamberti, Andrea Sanna, Claudio Demartini, "A Relation-Based Page Rank Algorithm for Semantic Web Search Engines", IEEE Transactions on Knowledge and Data Engineering, Vol.21, No.1, 2009.

[4] Gae-won You, Seung-won Hwang, "Search structures and algorithms for personalized ranking", Science Direct, Information Sciences, 178, 3925–3942, 2008.

[5] M Thangaraj, V Gayathri, "An Effective Technique For Context – Based Digital Collection Search", Proc. of ICMLC 2011, Vol 5, Singapore, pp 128 – 131.

[6] Ramiz M. Aliguliyev, "A new sentence similarity measure and sentence based extractive technique for automatic text summarization", Expert Systems with Applications 36, 7764–7772, 2009.

[7] Rudi L. Cilibrasi, Paul M.B. Vitanyi, "The Google Similarity Distance", IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No. 3, 2007.

[8] Yen-Liang Chen, Yu-Ting Chiu, "An IPC-based vector space model for patent retrieval", Information Processing and Management, Article in press, 2010.

[9] Yufei Li, Yuan Wang, Xiaotao Huang, "A Relation-Based Search Engine in Semantic Web", IEEE Transactions on Knowledge and Data Engineering, Vol.19, No.2, 2007.

M Thangaraj received his post-graduate degree in Computer Science from Alagappa University, Karaikudi, M.Tech. degree in Computer Science from Pondicherry University and Ph.D. degree in Computer Science from Madurai Kamaraj University, Madurai, TN, South India in 2006.

He is now the ASSOCIATE PROFESSOR of Computer Science Department at M.K.University. He is an active researcher in Web mining, Semantic Web and Information Retrieval and has published more than 50 papers in Journals and Conference Proceedings.

He is a senior member of IACSIT. He has served as program chair and program committee member of many international conferences held in India for about one decade.

V Gayathri received her post-graduate degree in Computer Science in 2008 and M.Phil. degree in Computer Science in 2010 from Madurai Kamraj University, Madurai, TN, India. She is currently a Ph.D. SCHOLAR in Dept. of Computer Science, M. K. University, Madurai, TN, India. Her research interests include Context-based Search, Information Retrieval and Semantic Web.