Optimized query navigation by concept hierarchies using data mining

Sahil Nyati
B.Tech (Information Technology)
VIT University, Vellore

Ankita Dhawan
B.Tech (Information Technology
VIT University, Vellore

Sweta Bhattacharya
Assistant Professor, SITE
VIT University, Vellore

***Abstract*— searching the medical information from biomedical database, such as PUBMED which contains all the information is sometimes tedious and fails to provide optimized results. In this paper we introduce a bio-Nav system which provides search interface to the users using the MeSH concept Hierarchy to avoid information overloading. The paper helps in ranking and categorization of the results and navigation tree consisting of nodes is formed. At each node expansion step, BioNav reveals only a small subset of the concept nodes, selected such, that the expected user navigation cost is minimized. This dynamic searching application provides an efficient heuristic and optimal result that helps in minimizing the navigation cost for the user.**

Keywords- Information overload, Biomedical database, Search interface Introduction

I. INTRODUCTION

There has been an enormous growth in the biomedical related information which has lead to the overloading of information. For a single Query almost 100000 citations are received amongst which user finds difficult to search the most appropriate result. Information overload makes it hard for the user to separate the interesting items from the uninteresting ones, thereby leading to a huge wastage of user’s time and effort. In this paper we propose a keyword search interface which retrieves results using MeSH concept hierarchy. First the query results are categorized and then ranked according to the user preferences. Categorization is based on the keywords, attributes values and id’s in the database. Ranking provides user with the list of results which are ordered by either the content similarity of the result or by set of results. The query results are categorized using the MeSH concept hierarchy methodology. In the database methodology the concept hierarchy method is used as a background knowledge and helps to express the discovered knowledge in high abstraction level in a more concise and interesting form. The discretization and concept hierarchy methods are the preprocessing steps for query results. A dynamic navigation tree is formed in which each node represents a category which is assigned with a descriptive label, examining which user can determine the relevance of the category. Entrez programming utilities are used which are collection of web interfaces to Pubmed for issuing a query and downloading the results with various levels of details. In the popularly used systems while querying the system iteratively the refinement process becomes problematic. In such systems after numerous iterations the user becomes unaware that the query has been over specified and finally the relevant results get excluded.

A. CATEGORIZATION

The information overload problem in the existing methodology is eradicated by categorizing the query result given by the user. In one-level categorization the information overload problem is recued. But it computes the cost at the last level of the tree. The drawback of this categorization is to distinguish the wanted and unwanted attribute without considering the partitioning. Every attribute is selected and obtained in a good partitioning the partitioning is done in the last step. So to eliminate this case, 2-level portioning is used to provide wanted categorization effectively.

B. RANKING

Ranking method is applied on the query results. In the existing system, ranking is done based on the similarities between the documents without considering the user interests. In the proposed system similarities between the documents and the user interests are considered by using the Web Page and Tag cluster Algorithm.

II. PROPOSED ARCHITECTURE

The BioNav System Architecture is shown in Figure 1. It mainly has two parts – online and offline processing of the database. In the offline processing it retrieves the results from the MeSH hierarchy and stores it in BioNav. For each concept in MeSH hierarchy, query on pubmed using a keyword is issued. In the BioNav database citation, id for each query result and the concept is stored in a tuple and this information is collected through Entrez Programming Utilities (eUtils). In the online process, BioNav executes the same keyword query from the user against the MEDLINE database and only citation id is retrieved. Entrez Programming Utilities (eUtils) are collection of web interfaces for issuing a query. This interface just shows the result and their expansion. Navigation tree for BioNav is constructed with each concept associated with each citation in the query results. In the hierarchy method, the search is done in a predefined static manner considering the navigation cost modeling and user interest. In this model it assumes that all users have the same user interests but in real life different users have different interests. The first step analyzes the query history of all users and generates a set of cluster over the data, each corresponding to one type of user interests. In the PubMed database there is a lot of choice for the user interests. Reducing the choice by means of categorizing the results removes the irrelevant data. So the user can quickly get their relevant data.

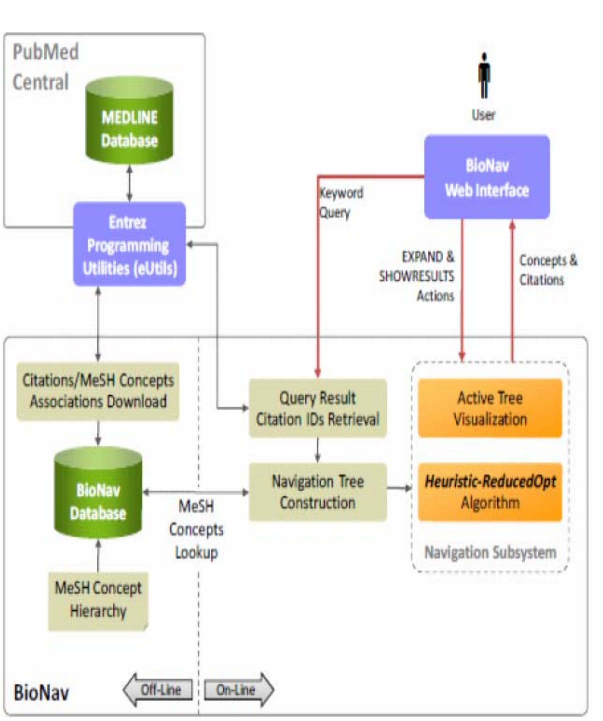
PubMed Central
MEDLINE Database
Entrez Programming Utilities (eUtils)
User
BioNav Web Interface
Keyword Query
EXPAND & SHOWRESULTS Actions
Concepts & Citations
Citations/MeSH Concepts Associations Download
Query Result Citation IDs Retrieval
Active Tree Visualization
BioNav Database
MeSH Concepts Lookup
Navigation Tree Construction
Heuristic-ReducedOpt Algorithm
Navigation Subsystem
MeSH Concept Hierarchy
BioNav
Off-Line
On-Line

Figure1. System Architecture

III. ALGORITHM FRAMEWORK

A. TWO-LEVEL CATEGORIZATION

In this multi-level categorization, it executes and verifies at each step. Due to this verification process it produces only relevant data at each step. As a result, this algorithm produces optimal result for the user query. In this algorithm, for each level we need to determine the categorizing attribute A and for each category will consider a level (L-1) and partition the domain values of attribute. It is categorized level by level according to the user interest. At each level (L-1) a node created and added to the tree. Then the cost of partitioning the attribute is computed and the attribute is selected as α with minimum cost. Finally it completes the node creation at level L. The categorization can be improved by the independence of user interest and the overwhelming of the results. It produces significantly a better category tree compared to the other models. The following algorithm is used for multi-level categorization.

B. ALGORITHM

```
 Begin
Create a root ("ALL") node
(level = 0) and add to T
L = 1; // set current level to 1
While there exists at least one Category at level L-1
With |tset(C)|>M
S<-{C|C is a category at level
(L-1) and |tset(C)|>M}
For each attribute A retained and not used so far
if A is a categorical attribute SCL->list of single categories in
desc order of occ($v_i$)
for each category C in Tree(C,A)<-Tree with C as root and
each non-empty cat
C' SCL in same order as Children of C else// A is numeric
attribute SPL<-list of potential Splitpoints sorted by goodness
score for each category C in S select (m-1) top necessary
Split points from SPLTree 15. (C,A)->Tree with C as root
with corr. Buckets in ascending order of values as children of
C $COST_A$<-$^2$£$_{cSP}$(C)*Cost17.All(Tree(C,A))
Select α = $argmin_A$ COST $_A$ as categorizing attribute for level
for each category C in S
Add partitioning Tree (C,α) obtained using attribute α to T
L = L+1; //finished creating
nodes at level, go to next level
end.
```

Using this algorithm a maximum of 100 XML document file is produced according to the user interest. This XML is given as the input for the ranking algorithm, because both the processes combined together helps in minimizing the information overload problem.

C. WEB PAGE AND TAG CLUSTER

Ranking is an efficient technique used for reducing the information overload and it can be powerfully implemented with categorization. The XML file is provided as input for the ranking algorithm as shown in Figure2. First, the download page needs to be preprocessed and tagged in order to remove irrelevant data. Then the quantity of words is ranked according to user interest. When user submits the query the algorithm preprocesses the query. It combines the content of web pages and ranks the results.

D. ALGORITHM

Input: Query q
Output: the ranked result list
Known: TC_i, TS_j, PS_i, PC_j, T_u: the number tags in page u,
P_v: the page v

```
1. List $Ltc_i$: the tags in $TC_i$,$Lts_j$: the tags in
$TS_j$;
List L1,L2,$L_{ij}$,$LL_{ij}$;
2. For I = 1;$Ltc_i$.size
If($Tc_i$ contains q)
Li.add($Tc_i$)
For j = 1:$Lts_j$.size
If($Tc_i$ contains q)
L1.add($Tc_i$)
For j = 1:$Lts_j$.size
If ($Ts_j$ contains q)
L2.add($Ts_j$)
```

```
3. For I = 1:$Ltc_i$.size
For j = 1:$Lts_j$.size
$L_{ij}$ = Sim(L1.i,L2.j)
4. Rank the elements in the $L_{ij}$ in descending order
5. Find out the largest K couples of TC-TS
and the corresponding PC snd PS,
respectively, and compute the coverage rate
For i = 1:Ltei.size
For j = 1:Ltsj.size
LLij = Cov(Pcj,Psi)
6. Rank the elements in the LLij in
Descending order
7. Find out the largest K couples of PC-PS, and ordered by the number of tags
8. If(q omnly belongs to $p_i$)
$P_i$ to be the first place
Else if ($T_u$ = $T_v$)
The more words the page has, the more previous it will be.
9. Returns the ranked list to the user.
```

This algorithm helps in producing the ranking list which is relevant to the user and useful while user searching the list.

IV. NAVIGATION COST MODEL

The tree navigation model is used to make the system much more cost efficient. The general navigation model is also useful to the user as it works from the top-down navigation starting from the root. In this tree each node is joined with another node containing all component sub tree rooted at n. A navigation tree is converted to an active tree by annotating the root node with a set that includes all tree nodes. This tree is closed under the reduced tree operation. This tree is similar to an embedded tree and the resulting tree are capable of reducing the tree both height and widthwise. The tree has a dynamic structure. Therefore the user can easily retrieve the result with a low navigation cost.

ALGORITHM

```
1 Collect all nodes of I(n) in list L
2 Create list L' to store the nodes of the reduced tree
3 Add to L' a concept node in L with the same label as C and all its ancestors
4 While (sizeof(L')<=maxN) repeat
5 Select a node c' uniformly at random from L
6 Add c' and all its ancestors to L', excluding duplicates
7 Create a tree I'(n) from the nodes in L', preserving the parent-child relationship
8 Return I'(n)
```

V. RESULTS AND DISCUSSION

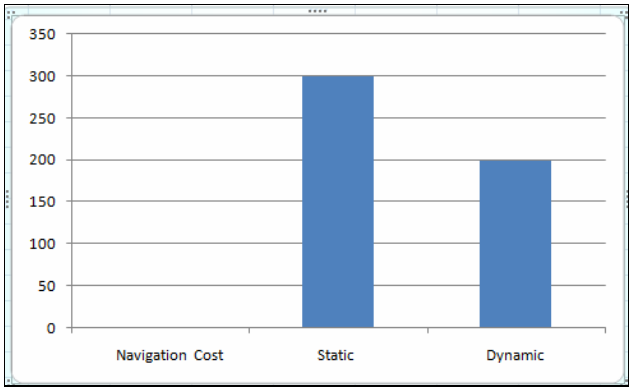

Figure 2. Static vs Dynamic Navigation cost

The performance of dynamic and static tree is compared using the experimental results .Both the performances are depicted in the Figure 2. The static tree shows all the results including the irrelevant one. But in dynamic tree only relevant links are shown which makes its navigation cost very less.

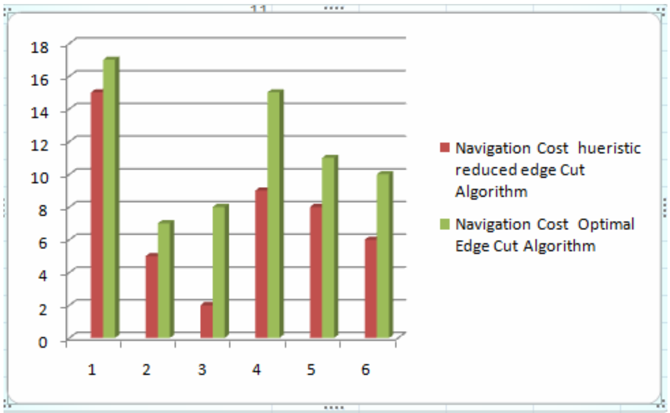

Figure 3. Overall Navigation Cost Comparison

The Navigation cost is calculated in case of Opt-Edge Cut algorithm which produces the predefined static structure and increases the navigation cost. It does not reduce the size of the tree and hence the users have to search the entire tree thus increasing the cost. This is the drawback of Opt-Edge Cut algorithm. To overcome this static Navigation, a dynamic approach algorithm called Gen-Reduced Tree is used which reduces the navigation cost of the user and produce a best result.

VI. CONCLUSION

Multi Level Categorization and ranking are used to minimize the information overload and the navigation cost faced by the users. MeSH concept hierarchy organizes the query results in the form of a navigation tree according to the user preferences using the PubMed database. The categorized file is given as input for ranking the file and then using the ranked list a dynamic navigation tree is generated. Each node expansion on the navigation tree, reveals a small set of nodes, selected from among its descendents, and the nodes are selected such that the information overload observed by the user is minimized. Static and Dynamic Navigation tree cost is also compared. It is also proved that the problem of selecting the set of nodes is NP-complete and it is an efficient heuristic approach. This reduced tree algorithm is a dynamic algorithm which results in less navigation cost compared to Optimal edge cut algorithm which is a static algorithm.

VII. FUTURE WORK

In the future work, Machine Learned Ranking (MLR) algorithm could be used. The main feature of this algorithm is to remove all the irrelevant data and further minimization of navigation cost so that the execution time could be reduced further.

ACKNOWLEDGMENT

The authors would like to thank Prof. Sweta Bhattacharya and Prof. Siva Rama Krishnan Somayaji for their support, guidance and encouragement. We would also like to thank School of information Technology Engineering, VIT University (Vellore) for all the support.

REFERENCES

[1]. J.S. Agrawal, S. Chaudhuri, G. Das, and A.Gionis, “Automated Ranking of Database Query Results,” Proc. First Biennial Conf. Innovative Data Systems Research, 2003

[2]. M. Kaki, “Findex: Search Results Categories Help When Document Ranking Fails,” Proc. ACM SIGCHI Conf. Human Factors in Computing Systems, pp.131-140, 2005

[3]. Chen, Z. and T. Li, “Addressing diverse user preferences in sql query-result navigation. Proc.”, J.ACM, SIGMOD, pp: 641-652, 2007.

[4].Justin Zobel, Alistair Moffat, Ron Sacks Davis, An Efficient indexing technique For full-text Database Systems, Proc of 18th Int. Conf (VLDB)., pp: 352-362, 1992

[5]. C. Plake, T. Schiemann, M. Pankalla, J. Hakenberg, and U. Leser,“Ali Baba: PubMed as a Graph,” Bioinformatics, vol.22, no. 19, pp. 2444.

[6]. H. Shatkay and R. Feldman, “Mining the Biomedical Literature in the Genomic Era: An Overview,” J. Computational Biology, vol. 10, no. 6, pp. 821- 855, 2003.

[7]. T. Zhang, R. Ramakrishnan, and M. Livny, “BIRCH: An Efficient Data Clustering Method for very Large Databases”, Proc. ACM SIGMOD, pp. 103 - 114, 1996.

[8]. R. Hoffman and A. Valencia, “A Gene Network for Navigating the Literature,” Nature Genetics, vol. 36, no.7, pp. 664, 2004.

[9]. Entrez Programming Utilities, http://www.ncbi.nlm.nih.gov/, accessed 2013.