Research Journal of Applied Sciences, Engineering and Technology 4(7): 856-862, 2012
ISSN: 2040-7467

Submitted: January 14, 2012 Accepted: February 02, 2012 Published: April 01, 2012

Development of an Efficient Cost based Ranking and Navigation Technique for Biomedical Databases

Janaky Balakrishnan and S. Subasree
School of Computing, SASTRA University, Thanjavur

Abstract: Information overload is the most important problem when searching biomedical databases. This paper gives the solutions for information overload and the navigation cost. Categorization and ranking manage the information overload. Automatic categorization system can generate their own sets of categories through the use of clustering. First, user gives the query and they group the search results into separate categories. Second the categorized results are ranked according to the user interest. The retrieved ranking result will produce the static and dynamic tree structure. Finally, the gen-reduced tree algorithm reduces the navigation cost when compared with the opt-edgecut algorithm. The comparison result produces the overall navigation cost for the user interest.

Key words: Automatic categorization, biomedical data set, clustering, navigation cost, ranking, tree navigation

INTRODUCTION

All Medical database must be able to respond the requests for information from the user (i.e., process queries). Many websites use this database to store the data for medicinal purposes. PubMed mainly using the keyword search interface (Kashyap *et al*., 2009). User can easily retrieve the queries. Queries are ultimately used to reduce the number of files scan operation. The queries are first categorized and then ranking according to the user interests and then finally constructing a tree for the navigation of databases. The tree navigation is similar to a decision tree. The PubMed medical database is a database of abstracts and articles from biomedical journals. PubMed collects biomedical articles data back from 1948. But PubMed was first released in 1996. It is the primary tool for searching and retrieving the biomedical journals. Day by day a million of queries are issued by the user (Hristidis and Papakonstantinou, 2002). It becomes more challenging for users to identify the relevant journal and articles. Users are often overwhelmed by the search results. Even though PubMed has number of advantages they are as follows,

- Users get a overview of the whole search result
- They can choose the number of categorized results
- It enable to derive the general keyword relevant to the search even though they are not mentioned in the article

But there are two major challenges to address the user interests, (Chen and Li, 2007).

- How to summarize the user interests from the behavior of all user already in the system
- How to decide the subset of user interests associated with a specific user

As an example, keyword on PubMed for "kidney" returns more than 1000 records. PubMed is distinct, since it offers the dynamic navigation on a static predefined hierarchy. In this work, once the user issues a query keyword, it will retrieve the relevant results from the biomedical dataset then the PubMed returns multiples of fifty categorized results in a XML file format (Kashyap *et al*., 2009. This result will be appropriate for the user. Since it uses dynamic navigation it will return only the relevant record alone.

LITERATURE REVIEW

The following Fig. 1 shows the architecture of BIONAV. It consists of two parts, offline and online components. In the offline part, it retrieve the results from the MESH hierarchy and stored in the BIONAV database like (concept, citation ID) and have to collect the information from Entrez Programming Utilities (eUtils). In the online process the same query will executes and retrieve only ID. It will just EXPAND and SHOWRESULTS regarding to the user interest. The navigation tree of BIONAV architecture will not be very effective.

In the hierarchy method, will be in a predefined static manner without 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

Corresponding Author: Janaky Balakrishnan, School of Computing, SASTRA University, Thanjavur

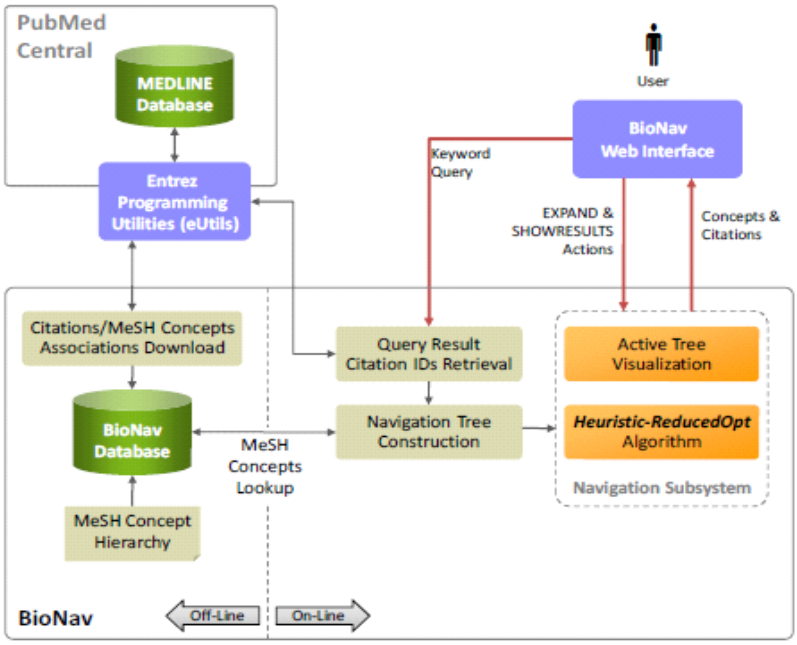

Fig. 1: BIONAV structure

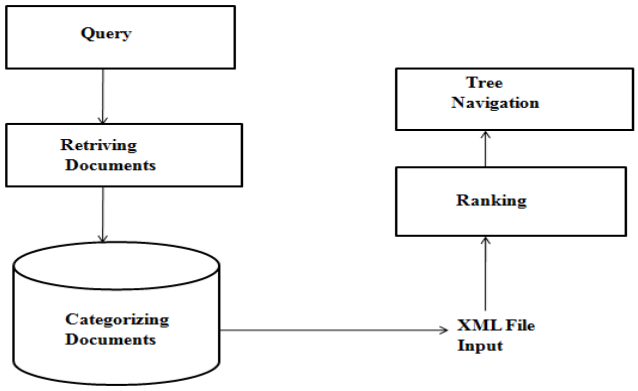

Fig. 2: Architecture of ranking and navigation technique

often have different interests (Kashyap *et al*., 2011). 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 will be a lot of choice for the user interests. Reducing the choice by means of categorizing the results will remove the irrelevant data. So the user can quickly get their relevant data.

Categorization: To solve the information overload problem the existing methodology categorizing the query result given by the user (Chakrabarti *et al.*, 2004) In the one-level categorization reduce the information overload problem, but it compute the cost at the last level of the tree. The drawback of this categorization is:

- It eliminates the wanted and unwanted attribute without considering the partitioning
- Every attribute selected and obtain a good partitioning but every partitioning will happen in the last step

It produces lot of unwanted data till the last level (Pratt and Wasserman, 2000) To overcome this drawback of one level categorization, in this project multi-level categorization algorithm will be used to enhance the categorization.

Ranking: It is easy to implement the ranking method on query analysis. In the existing ranking algorithms are purely based on the similarity between the documents.

(Zobel *et al*., 1992) While finding similarities between the documents it won't consider the user interest while ranking the documents.

In this study, we implemented Web Page and Tag clustering algorithm (Zhao *et al*., 2011) is used for ranking which will focus the not only the similarities between the documents, but also it focuses the user interest into consideration.

Navigation cost: The Navigation cost is calculated in the Opt-Edge Cut algorithm will produce the predefined static manner so it increases the navigation cost (i.e., user time). (Chen and Li, 2007) It won't reduce the size of the tree. So the users have to search the entire tree. Because of searching the entire tree the cost will be increased (Kashyap *et al*., 2011). This is the drawback of Opt-Edge Cut algorithm.

To overcome this static Navigation, we have used dynamic approach algorithm called Gen-Reduced Tree, which will reduced the navigation cost of the user and produce a best result.

PROPOSED TECHNIQUE

The following Fig. 2, architecture shows that the query given by the user will be retrieved from the pub med database. After retrieving the documents it will categorize the documents according to the relevant data. It will execute an XML file with some documents. The XML file will be given as input for ranking the results according to the user interest. Finally a dynamic tree navigation will be constructed for good results.

Multi level categorization: In this multi-level categorization, it will execute and verify at each step. Because of this verification process at each step will produce only relevant data. So, this algorithm will produce 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 as level (L-1) and partition the domain values of attribute. Categorizing the level by level step according to the user interest. At each level (L-1) a node created and added to the tree. Compute the cost of partitioning the attribute and select the attribute 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 better category tree compared to the other models. The following algorithm is for multi-level categorization.

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
    4. 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
    splitpoints from SPLTree
15. (C,A)←Tree with C as root
    With corr. Buckets in ascending
    order of values as children of C
    COST_A←£_{c S}P(C)*Cost 17._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 we can produce the maximum of 100 XML document file according to the user interest. This XML file will be given as the input for the ranking algorithm, because both the process combined together will minimize the information overload problem.

Web page and tag cluster: Ranking is an efficient technique for reducing the information overload and can be powerfully implemented with categorization. We are giving the XML file as input for the ranking algorithm. First, we need to preprocess the download page and tag such as removing all the unwanted data's. The quantity of words will be ranked according to user interest.

When user submits the query the algorithm will preprocess the query. It combines the content of web pages and rank the results.

The Fig. 3, consists of two sets

- tag cluster (TC)
- Web Page Cluster (PC).

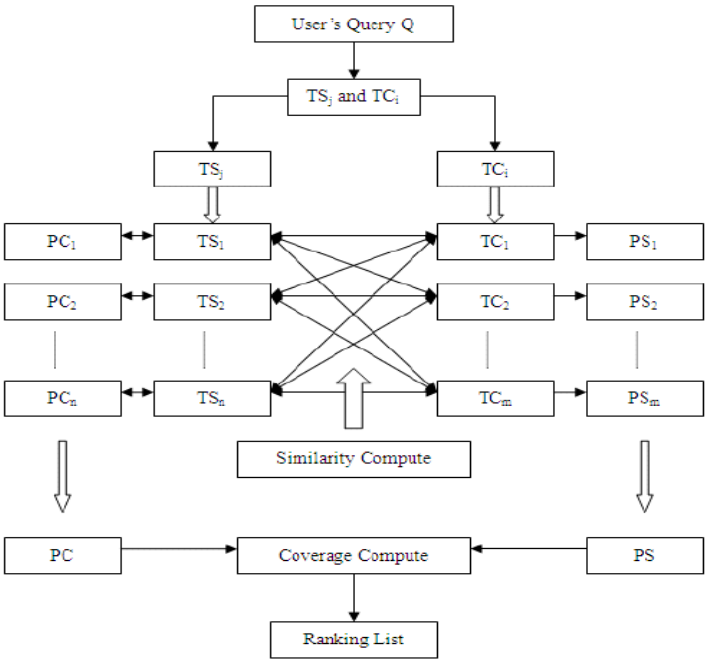

Fig. 3: Ranking architecture

Every tag cluster will map a web Page Set (PS) the element page at least include one tag in this cluster. In the same process each web page cluster (PC) will map a tag set (TS). Finally it find the similarity between all sets and cluster and it will cover all the relevant data and ranked according to user interest.

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)
   L_i.add(Tc_i)
   For j = 1:Lts_j.size
   If(Tc_i contains q)
   L_1.add(Tc_i)
   For j = 1:Lts_j.size
   If(Ts_j contains q)
   L_2.add(Ts_j)
3. For I = 1:Ltc_i.size
   For j = 1:Lts_j.size
   L_{ij} = Sim(L_1.i,L_2.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:Lte_i.size
   For j = 1:Lts_j.size
   LL_{ij} = Cov(Pc_j,Ps_i)
6. Rank the elements in the LL_{ij} 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. Return the ranked list to the user

Using this algorithm we are producing the ranking list which is relevant to the user. The ranking list will be very useful while user searching the list.

Tree navigation: The tree navigation model is used to reduce the cost model. The general navigation model is useful to the user. 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 an set that includes all tree nodes. These tree is closed under the reduced tree

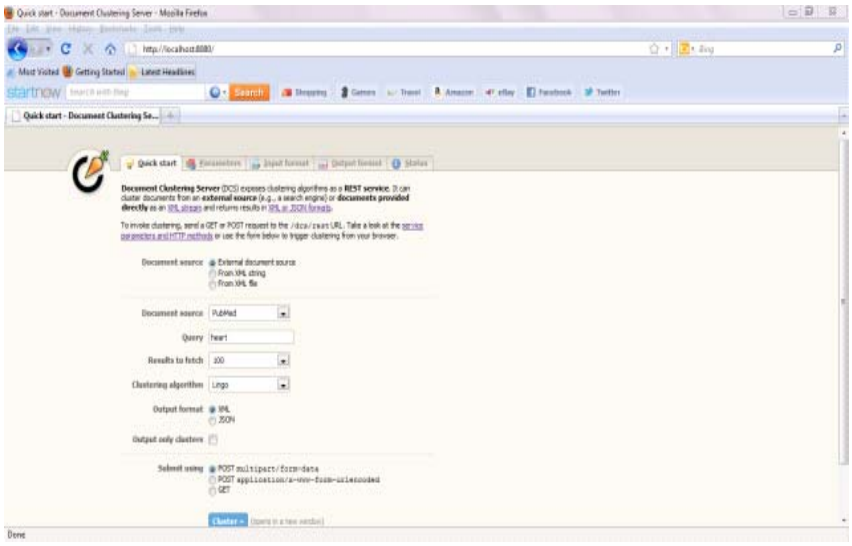

Fig. 4: User giving the query keyword

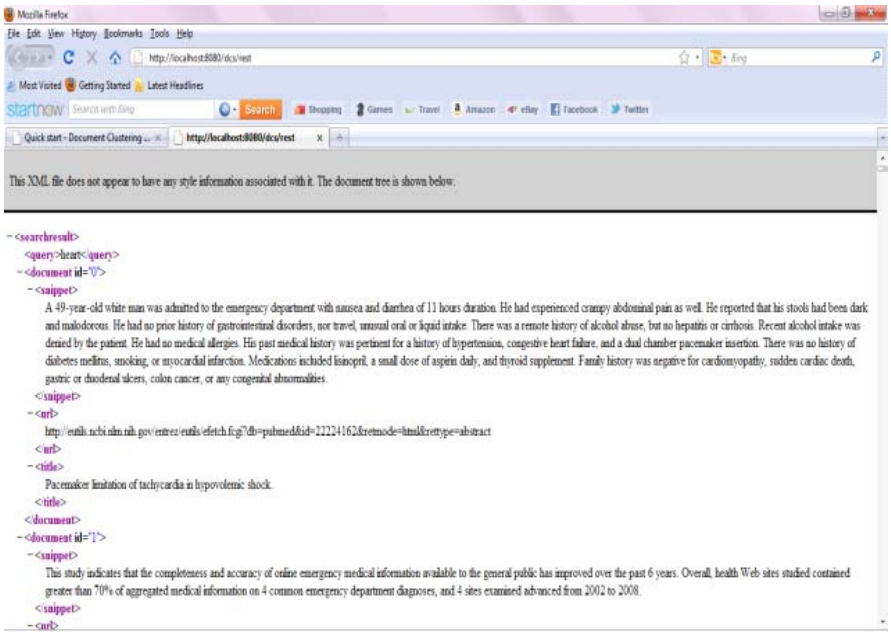

Fig. 5: User giving the query keyword

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 will be in a dynamic manner. So 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)

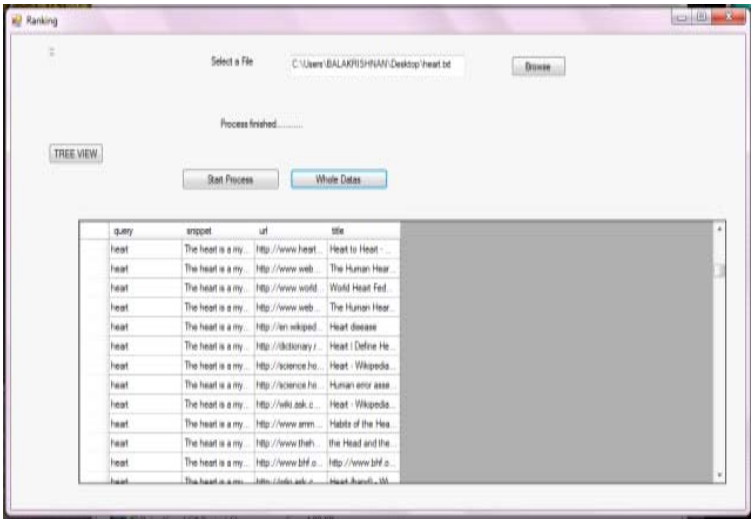

Fig. 6: Ranking the heart disease

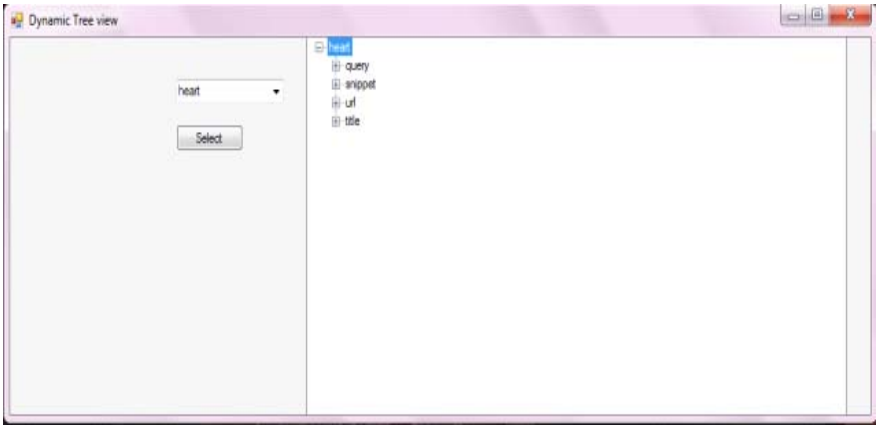

Fig. 7: Dynamic tree view

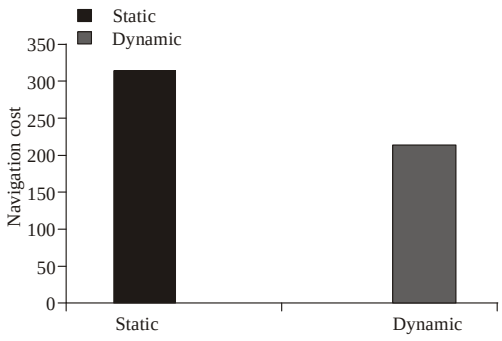

Fig. 8: Static vs dynamic navigation cost

PERFORMANCE ANALYSIS

The categorization, Ranking and Tree Navigations are carried out and implemented using NET with C#.

The Fig. 4, the user will give the keyword query for categorizing the results. It consists of Query name, Source website and number results to be fetched etc. The Fig. 5 after giving the keyword query, it will categorize and give multiples of fifty records relevant to search query. The Fig. 6, will produce the ranking of links based on the categorization. The dynamic tree view of the ranked links can be shown in the Fig. 7.

The experimental result were designed to compare the performance analysis of static and dynamic tree view. The performance analysis of static and dynamic tree navigation can be shown in the Fig. 8. In the static model it will show all the links including the irrelevant one. But, in the dynamic model it will shown only relevant links alone. From the Fig. 8 it is clearly identified that compare to static tree navigation cost the dynamic tree navigation cost will be less.

CONCLUSION AND FUTURE WORK

The information overload problem and the navigation cost is reduced using the categorization and ranking process. The user can get the relevant results by organizing the query. The categorizing results are generated from the PubMed database. The categorized file will be given as input for ranking the file. Finally using the ranking list, we are generating the dynamic tree view for user convenience. In addition to this we are comparing the static and dynamic tree navigation cost. This generation reduced tree algorithm is a dynamic algorithm will produce less navigation cost compare to Optimal edge cut algorithm which is a static algorithm. In the future work, we are planned to introduce Machine Learned Ranking (MLR) algorithm. By using this ranking algorithm we remove all the irrelevant data, so that the tree navigation cost will be further reduced, automatically the execution time is also minimized.

REFERENCES

Chakrabarti, K., S. Chaudhuri and S.W. Hwang, 2004. Automatic categorization of query results. Proc. ACM SIGMOD, pp: 55-766.

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

Hristidis, V. and Y. Papakonstantinou, 2002. DISCOVER: Keyword search in relational databases. Proceeding of International Conference on Very Large Databases (VLDB).

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

Kashyap, A., V. Hristidis, M. Petropoulos and S. Tavoulari, 2011. Effective navigation of query results based on concept hierarchies. IEEE Transact., 23(4): 1041-4347.

Kashyap, A., V. Hristidis, M. Petropoulos and S. Tavoulari, 2009. BioNav: Effective navigation on query results of biomedical databases. Proc. IEEE Conf. Data Eng. (ICDE), pp: 1287-1290.

Pratt, W. and H. Wasserman, 2000. QueryCat: Automatic Categorization of MEDLINE Queries. AMIA, pp: 1067-5027.

Zhao, Z. Zhang, H. Li and X. Xie, 2011. A search result ranking algorithm based on web pages and tags clustering., IEEE, Conf., 978-1-4244-8728-8.