

Maximizing User Experience While Navigating Query Results Through Concept Hierarchies

Ravikumarreddy. B,
Department Of CSE,
Bharat Institute Of Engineering And Technology,
Ibrahim Patnam, Hyderabad, India.

Abstract

Web databases such as PubMed return large number of tuples against user queries. However, user may not be interested in all the results. To navigate to the required results user has to spend some time in browsing the results. This is the problem that needs research. To solve this problem the query results can be presented in the form of ontologies. This paper provides required algorithms to reduce the number of tuples and also focuses on presentation of query results that would be effective and results in rich user experience by minimizing the navigation cost and reducing the number of expansions. The experiments are done using PubMed database which is having entries with specific annotations that is used to effectively prepare query results for effective navigation by end users. This paper presents a new search interface and the algorithms that make the results to be most effective navigation so as to reduce the browsing time. The experimental results revealed that the proposed system is useful and can be used in the real time applications.

Index Terms—Effective navigation, PubMed database, ontologies, navigation cost

INTRODUCTION

The amount of data provided over World Wide Web (WWW) is increasing rapidly every year. In the past decade it started growing drastically. Especially biomedical data and the literature pertaining to it that reviews the aspects of biomedical data across the globe have seen tremendous growth in terms of quantity. Biological data sources such as [1], [2], and [3] are growing in terms of lakhs of new citations every year. The queries made by people associated with healthcare domain have to search such databases by providing a search keyword. The results are very huge in number and the users are not able to view all the records

when they actually need a subset of them. This has led to users to refine query with other keywords and get the desired results after many trials. Here it has to be observed that user time is wasted in refining search criteria and also the navigation of query results which are abundant and bulky. The navigation cost is more as user has to spend lot of time in finding the required subset of rows from the bulk of search results. This problem has been researched in [1], [2], [3] and the problem is identified as information overload. Figure 1 shows static navigation of MeSh hierarchy of biomedical data.

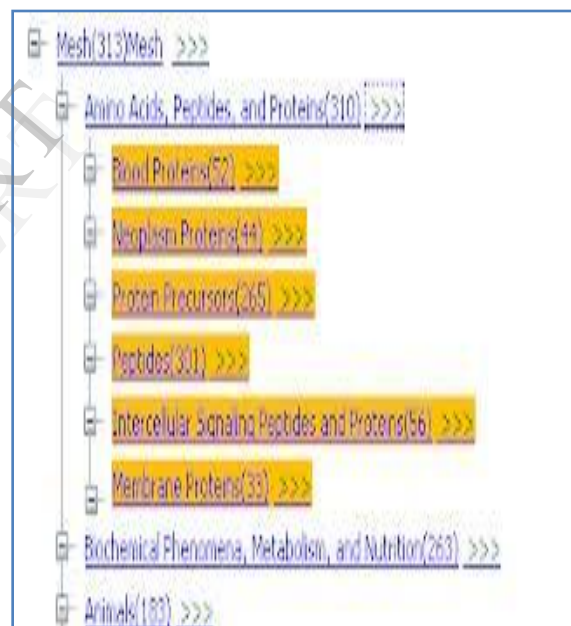


Fig. 1: Static structure of MeSh hierarchy of biomedical data[7]

The solutions are of two types namely categorization and ranking. However, these two can be combined to have more desired results. The proposed system is specially meant for presenting results in such a way that the navigation cost is reduced. For this purpose categorization techniques is used and concept hierarchies are built. The categorization techniques are supported by simple ranking techniques. The proposed solution uses citations as described in [4] and effectively constructs a navigation tree that can reduce cost of navigation and user's experience is much better when compared with existing systems that do not use these techniques. These techniques are being used

by e-Commerce systems to let their users have smooth navigation to the results returned by such systems.

The proposed system uses a cost model that lets it estimate the cost of navigation and make decisions in providing concept hierarchies. The cost of navigation is directly proportionate to the navigation subtree instead of the whole results in the tree. Earlier work on dynamic categorization of query results are in [2], [3], [5] and [6]. They made use of query dependent clusters based on the unsupervised technique. However, they neglect the process of navigation of clusters. In this aspect the proposed system is distinct and provides dynamic navigation on a pre-defined concept hierarchy. Another telling difference between existing systems and the proposed one is that the proposed system uses navigation cost model that minimizes navigation cost no matter what the bulky of search results is. Overall, our contributions are development of a framework for effective navigation of query results; a formal model for cost estimation; algorithm to optimize the results' navigation cost; experimental evidence on the effectiveness.

OVERVIEW OF THE PROPOSED FRAMEWORK

MeSH (Medical Subject Headings) is the hierarchy based on which the framework is built. It is a made up of a concept hierarchy. A concept hierarchy is a labeled tree which has concept nodes and edges with a root node. Each node is identified by a unique ID and associated with a label. As discussed in [8], as per the MeSH concept hierarchy the label of parent is not specific when compared with that of child. Almost all concept hierarchies follow the same conventions. We have built a prototype application whose interface after search operation is as shown in fig. 2.

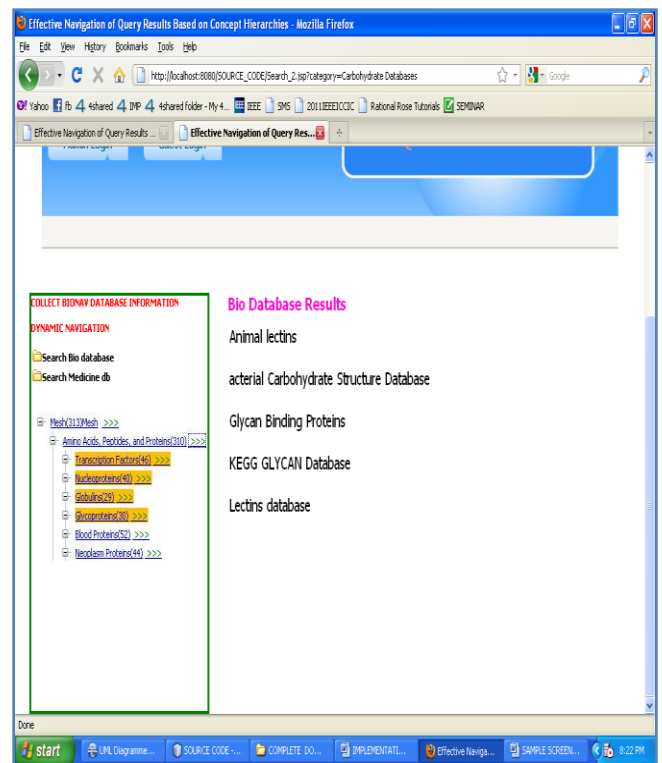


Fig. 2 –Prototype application

When a query made by user, the application returns results in terms of citations list associated with MeSH concepts. The application constructs an initial navigation tree which is nothing but a concept hierarchy. Every node in that hierarchy is known as a concept. The navigational tree presented initially may contain nodes that are empty. As MeSH is very large hierarchy, the proposed application removes empty nodes and thus reduces size of the navigation tree. The removal of empty node is done carefully by preserving node relationships. The resultant navigation tree is nothing but the initial navigation tree where the empty nodes have been removed to reduce the size of the tree.

By removing empty nodes from the navigation tree, the tree size is reduced. However, the structure of the navigation tree is very big and can't be presented to end user as it is. The proposed application automatically expands the tree at required place to ensure that user navigation cost is less and user will directly see the desired results. Towards this end the application takes care of hiding unnecessary nodes and presenting desired nodes in such a way that the navigation cost of to the user is negligible. We consider a node expansion at given place as an "EdgeCut". In case of trees, in general an EdgeCut is a subset of edges because the removal of edge causes a new component to be created. Visualization of an EdgeCut is shown in fig. 3.

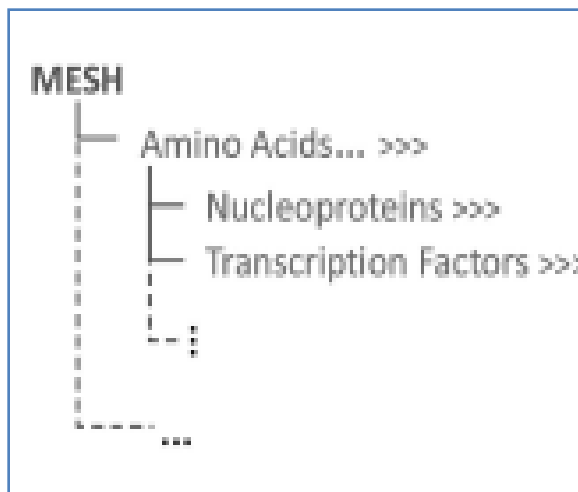


Fig. 3 – Visualization of EdgeCut[9]

Navigation tree with an EdgeCut is shown in fig. 9 on the user interface. The EdgeCut results in creation of sub trees and they are of two types namely lower and upper. Before an EdgeCut is performed, the navigation tree is converted into something known as active tree. It is achieved by annotating root node. The navigation tree is reduced both width and height wise due to the EdgeCut and presentation of the resultant tree. We do assume any user's preference on the results and every part of the tree can be reached by user as the navigation does not result in information loss in the proposed framework. We also use a convention i.e., ">>>" used to provide hyperlinks that facilitate user to perform EdgeCut operations in recursive manner. Generally we expect EdgeCut to be triggered by user on the lower component of sub trees. However, it is also possible that such operation may occur at upper component sub tree. Fig. 4 shows the resultant tree with hyperlinks in ">>>" convention.



Fig. 4 –Resultant Tree

ARCHITECTURE OF PROTOTYPE SYSTEM

The architecture of the prototype application is as shown in fig. 5. The architectural operations are classified into online and offline operations.

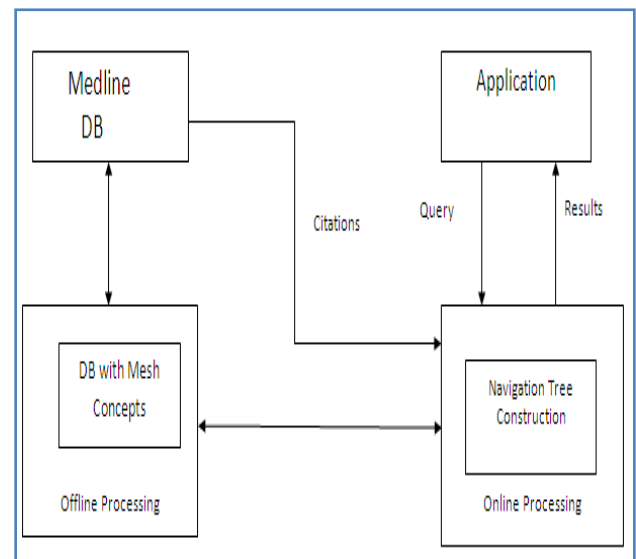


Fig. 5 – Architecture of Proposed Application

As seen in fig. 5, the proposed system operations are classified into online and offline operations. The offline operations include storing MeSH concepts hierarchies into local database, exchanging MeSH concepts between PubMed DB and Local DB. The online operations include question results generation, obtaining citations from MEDLINE database. Navigation tree construction and other navigation sub system operations such as active tree visualization are also online operations. The online operations actively start with query given by end user on biomedical database.

NAVIGATION AND COST MODEL

Once user issues a query keyword, the prototype application generates an initial active tree with single component tree. Its root is the root of the MeSH hierarchy. Later on user performs one of the following operations.

EXPAND: It takes place when user clicks ">>>" hyperlink that causes EdgeCut to be performed that shows a set of nodes.

SHOWRESULTS: User performs this operation to view results.

IGNORE:User can simply ignore a node after looking at its label and moves to next concept.

BACKTRACK:This action occurs when user performs undo on the last EdgeCut operation.

User continues these operations until he gets the intended results. We simplify this simple navigational model and call it "TOPDOWN". The TOPDOWN model has only three operations namely EXPAND, SHOWRESULTS and IGNORE.

```

Explore(I(n))

If n is the root

S ← Expand I(n) // that is S ← EdgeCut(I(n))

For each ni in S

Explore(I(ni))

else, if n is not a leaf-node, choose one of the following :

1. SHOWRESULTS I(n)
2. IGNORE I(n)
3. S ← EXPAND I(n)
   For each ni in S
   EXPLORE (I(ni))
   Else, choose one of the following: // n is a leaf node
   1. SHOWRESULTS I(n)
   2. IGNORE I(n)

```

Listing 1 – TOPDOWN Navigation Model

The cost model as specified in [2] considers the following to compute navigation cost.

- Number of EXPAND actions.
- Number of concept nodes shown by a single EXPAN action.
- Number of citations presented for a single SHOWRESULTS action.

For each of these things cost of 1 is considered. The cost of exploring a component sub tree I(n) rooted at node n is

$$Cost(I(n)) = P_E^N(I(n)) \cdot (1 - P_c(I(n))) \cdot L(I(n)) / + P_c(I(n)) \cdot (B + |S| + \sum_{S \in S} cost(I_c(S)))$$

where normalized $P_E(I(n))$ is represented as $P_E^N(I(n))$ thus the sum of normalization of subtree of the component after an EdgeCut is equal to 1.

BEST EDGE CUT ALGORITHMS

Optimal cost can be computed by recursively listing all possible sets of EdgeCuts. This starts from the root and traverses every concept in the tree. This algorithm is expensive. To overcome this OptEdgeCut algorithm is proposed which provides minimum expected navigation cost.

```

Algorithm. Opt-EdgeCut
Input: The navigation tree T
Output: The best EdgeCut
1 Traversing T in post order, let n be the current
2 while n ≠ root do
3 if n is a leaf node then
4 mincost(n, ∅) ← PE(n) * L(n)
5 optcut (n, ∅) ← {∅}
6 else
7 C (n) ← enumerate all possible EdgeCuts
for the tree rooted at n
8 II(n) ← enumerate all possible sub trees
for the tree rooted at n
9 foreach I(n) ∈ (n) do
10 compute PE(I(n)) and Pc(I(n))
11 foreach C ∈ (n) do
12 if C is a valid EdgeCut for I(n) then
13 cost (I(n) C) ←
PE(I(n)) * ((1 - Pc(I(n))) * L(I(n)) + Pc(I(n)) * (B + |S|))
14 else
15 cost (I(n), C) = ∞
16 mincost(n, I(n)) ← minC ∈ C(n) cost (I(n), Ci)
17 optcut (n, I(n)) ← Ci
18 return
Optcut(root, E) // E is the set of all tree edges

```

Listing 2 – OptEdgeCut Algorithm

The algorithm Opt-EdgeCut is supposed to compute the minimum expected navigation cost required for navigating the tree from bottom up in post order fashion.

Heuristic-ReducedOpt Algorithm

The Opt-EdgeCut algorithm is computationally more expensive and can't be practically used for most of the queries. Therefore, we proposed a new algorithm known as Heuristic-ReducedOpt as shown in listing 3.

```

Algorithm. Heuristic-ReducedOpt
Input: Component subtree I(n), number z of partitions
Output: The best EdgeCut
1 z' ← z
2 repeat
3 k ← ∑n ∈ T L(n) * PE(n) / z'
4 Partitions ← k-partition(I(n), k)
// call k-partition algorithm [14]
5 z' ← z' - 1
6 until |Partitions| ≤ z
7 construct reduced subtree I'(n) from Partitions
8 EdgeCut' ← Opt-EdgeCut(I'(n))
9 EdgeCut ← corresponding of EdgeCut' for I(n)
10 return EdgeCut

```

Listing 3 – Heuristic ReducedOpt Algorithm

This algorithm is based on k-partition algorithm [10]. This is adapted for our use here. The algorithm works in bottom-up

fashion. For every node (n), the algorithm prunes heaviest children one by one till the weight of n falls below k .

EXPERIMENTAL EVALUATION

For evaluating the proposed application, expansion time performance and average navigation cost are considered. The empirical studies are made in a PC with XP as operating system. Oracle 10 g is used as backend and Java is used to implement all algorithms. The proposed application achieves improvement in navigation cost when compared with Top1-LeaveWise. Minimum improvement is 16 percent and maximum improvement is 41 percent. Fig. 6 shows number of EXPAND actions comparison.

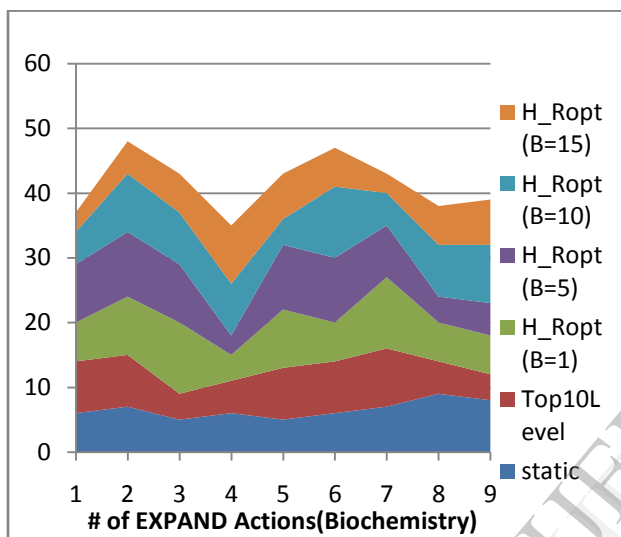


Fig. 6 – Comparison of number of expand operations

As can be seen in fig. 6, ten queries have been presented for six types of algorithms. The X axis takes question numbers while the Y axis represents value. Out of all the algorithms the proposed application and its algorithm. Fig. 7 shows the results of number of concepts shown.

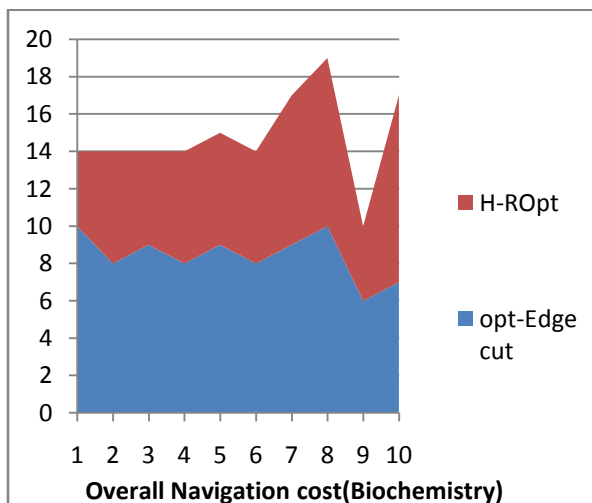


Fig. 7 – Comparison of number of concepts revealed

Fig. 7 shows the number of concepts shown when an EXAPND action takes place. The results revealed that our approach is superior to many other approaches.

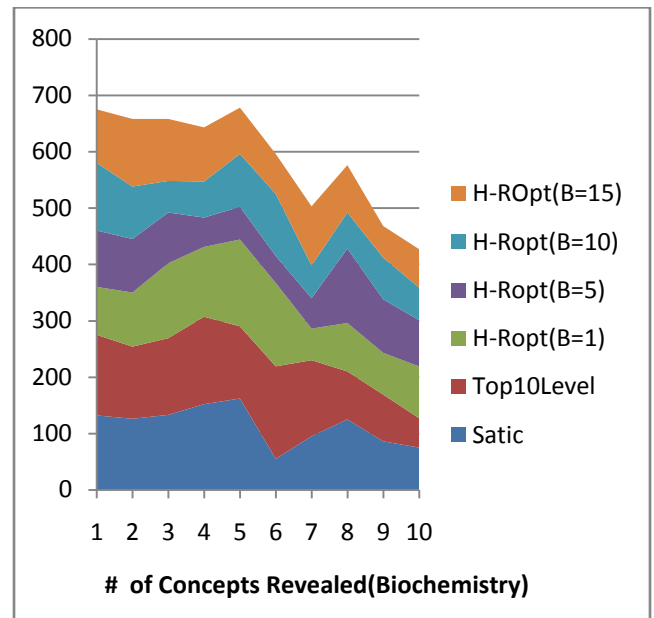


Fig. 8 – Comparison of overall navigation cost

As can be seen in fig. 8 comparison is made on proportional navigation cost of Heuristic-ReducedOpt over Opt-EdgeCut. Opt-EdgeCut algorithm has best results when executed on biochemistry database.

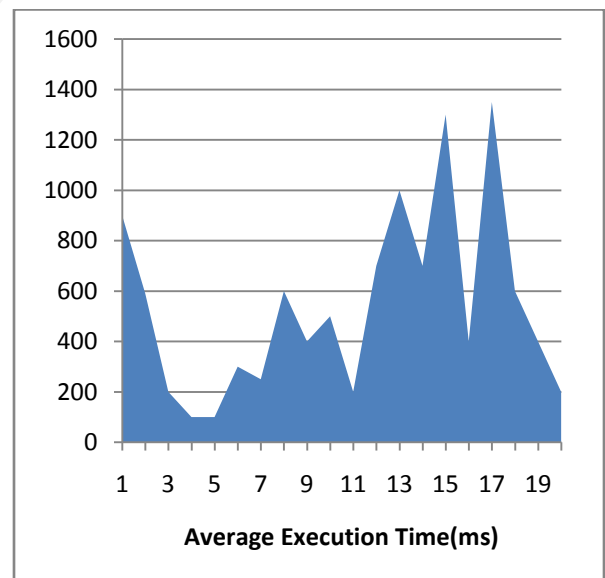


Fig. 9-Heuristic-ReducedOpt EXPAND performance.

As seen in fig. 9, the average time of Heuristic-ReducedOpt to execute and EXPAND action with respect to each query of table 1. The average values are taken from the number of EXPAND action provided in fig. 6.

CONCLUSION

This paper focuses on developing a framework that facilitates rich user experience while navigating query results. This framework is required as the large medical databases available over Internet return huge number of records as results. Navigating the results to obtain required information causes wastage of user's time. The proposed framework can effectively reduce the navigation cost and provide easy access to required results. Thus it can solve information overload problem effectively. The results organized are associated with MeSH annotations by user a dynamic navigation method. A new cost model is proposed that reduces the navigation cost as it can eliminate unnecessary navigation steps. A prototype application is built with web based interface, which facilitates users to search for required information and also navigate the results. The experimental results revealed that the user navigation cost is reduced substantially and rich user experience is achieved.

REFERENCES

1. J S. Agrawal, S. Chaudhuri, G. Das and A. Gionis: *Automated Ranking of Database Query Results*. In Proceedings of First Biennial Conference on Innovative Data Systems Research (CIDR), 2003.
2. K. Chakrabarti, S. Chaudhuri and S.W. Hwang: *Automatic Categorization of Query Results*. SIGMOD Conference 2004: 755-766.
3. Z. Chen and T. Li: *Addressing Diverse User Preferences in SQLQuery- Result Navigation*. SIGMOD Conference 2007: 641-652.
4. Medical Subject Headings (MeSH®). <http://www.nlm.nih.gov/mesh/>
5. Vivísimo, Inc. –Clusty. (2008) [Online]. Available: <http://clusty.com/>
6. A. Kashyap, V. Hristidis, M. Petropoulos, and S. Tavoulari: *BioNav: Effective Navigation on Query Results of Biomedical Databases*. (Short Paper), ICDE 2009, to appear. Available at <http://www.cs.fiu.edu/~vagelis/publications/BioNavICDE09.pdf>
7. HON (2010): Health On the Net Foundation: Medical information. Available online at: <http://www.hon.ch/cgi-bin/HONselect?cat+A> [viewed: 15 August 2012]
- [8] Medical Subject Headings (MeSH), <http://www.nlm.nih.gov/mesh/>, 2010.
- [9] Abhijith Kashyap, Vagelis Hristidis, Michalis Petropoulos, and Sotiria Tavoulari (2011), "Effective Navigation of Query Results Based on Concept Hierarchies". IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 23, NO. 4.
- [10] S. Kundu and J. Misra, "A Linear Tree Partitioning Algorithm," SIAM J. Computing, vol. 6, no. 1, pp. 151-154, 1977.