CHAPTER 4

LINK STRUCTURE DISPLAY

4.1 INTRODUCTION

In this chapter we point out our thoughts on the importance link structure display for enhanced user personalization. Visualized link hierarchy can enable user in faster attainment of the desired information. In the virtual learning environment this can make the topic/study more interesting, as he is aware of the site structure and from where the information could be fetched fast. The organization of this chapter is as follows: In section 4.2, the motivation behind this idea is briefed. In section 4.3, we present an outline of the methods that could be used for visualization. The novel idea of our tree based visualization and how it facilitates faster navigation is presented in section 4.4. Section 4.5 is a comparison of our approach with related work. Section 4.6 gives the conclusion of this chapter.

4.2 MOTIVATION

World wide web being the large repository of webpages users find it difficult to leverage information fast. For reaching a desired page they may traverse many hyperlinks. Users of a webpage are in need of techniques which enable faster and easy traversal of the required page with minimum hyperlink visit. Visualization techniques can enable user achieve this. Data mining techniques help in extracting the required data from usage logs. Using some methodologies if we can arrange it in the requested format and display the link hierarchy user personalization could be achieved. On view of the displayed structure itself the user can get an idea of his visits pattern. This
visualization also enables him to get an idea of the site structure without crawling over each and every hyperlink.

The display of link hierarchy enables the user to identify their position in relation to the whole website. This knowledge may enable faster traversal and thus personalization. Qingcai Chen and Hongzhi Guo (2008) modeled a link structure based approach for visualizing site map for faster navigation. Their proposed future work of topic hierarchy is considered in our methodology.

4.3 METHODS USED FOR VISUALIZATION

Through the visualized structures users can easily sketch relationships between the pages they have already visited. This also facilitate faster traversals and thus user personalization. Many methods are employed for the visualization of a website. Graph representations are commonly followed but we attempt to present the structure as a tree type of representation. Taking a sample site pages are represented as nodes in hierarchical order. Hyperlinks are shown at sublevels. Visualization enables the user for faster navigation as he is aware of his state in relation to the web site as whole. Here we present a tutorial on some of the methods used to display the link hierarchy of a website.

The traditional Breadth First Search (BFS) method could be used for the visualization of the link hierarchy of a website. In the BFS method a level wise traversal approach is adopted. Starting from the root which is the home page, first level of nodes traversed, then the second level and so on till the last level is reached. The number of visits of a particular page are represented by the weight on the page. For personalization we need to select the maximum weighed path faster. In the BFS method this could be found
only by traversing the number of nodes in a level wise manner only. These values have to be taken, compared and we should find the maximum value/hit path for calculating pages for personalization. BFS do not ensure faster traversal or user personalization. Best First Search (BFS) is also a similar algorithm that traverse a graph in search of one or more goal nodes. This algorithm uses a heuristic to determine the most promising object.

In the MAPA system [Durand and Kahn 1998] for website navigation hierarchical structure is extracted using shortest weighted path method. These shortest weighed paths are calculated based on the user assigned and heuristically determined weights. This visualization also does not contribute much to personalization of a particular user.

Kleinberg (1999) proposed an algorithm named Hyperlink Induced Topic Search which effectively calculates the weights of webpages. The algorithm calculates weight by constructing an adjacency graph of the page under consideration. For calculating the weight of a page, authoritative weight as well as the hub weight is considered. HITS is an efficient algorithm for page rank calculation but this does not ensure personalization. Google follows this algorithm for ranking pages of a search. The main drawback of HITS is that there is elevated possibility of old web pages ranked high as it takes total weight on a webpage for ranking.

Munzner (2000) showed that visualization of a website can help people in exploring and explaining data. Their study constructed spanning tree representation of websites by Depth First Traversal of the graph. The idea was not personalization but from the graph when the spanning tree was formed they followed a reshuffling of nodes based on top matching priority. The resulted final spanning tree looked more similar to Breadth First Search than Depth First Search.
Jianhan Zhu et al. (2004) proposed a new method to visualize link hierarchy of a website. They modified the page rank approach of Page et. al. (1998). They had treated the weight of a link as the number of user traversals. The link hierarchy with different conceptual levels is formed then. They ranked the pages in the order of user traversals on hyperlinks based on their proposed Page Rate Algorithm which is a modified version of the Page Rank algorithm. A random surfers’ pattern also is considered for ranking pages using this algorithm.

According to Jianhan Zhu et al. (2004), Page Rate of a Page A is:

\[
PR(A) = \frac{1 - d}{N} + d \sum_{j=1}^{IN(A)} PR(P_j) \cdot \frac{n(P_j, A)}{\sum_{i=1}^{OUT(P_j)} n(P_i, P_j)}
\]  

(4.1)

where \( IN(A) \) – number of incoming links termed as in-links to page A

\( P_j \) - Page linking to page A

\( n(P_j, A) \) - number of user traversals from \( P_j \) to the page A

\( OUT(P_j) \) - number of out links of the page \( P_j \)

\( \sum_{j=1}^{OUT(P_j)} n(P_j, P_i) \) - sum of all user traversals on all the out-links of page \( P_j \).

The damping factor \( d \) is set between 0 and 1 which is a measure of the probability of visit of a Page. \( N \) is the total number of pages in the structure. The visualized structure enhances faster navigation/search but as it is not specific user centric it does not enhance personalization.
Florian Verhein (2008) explained the frequent pattern growth algorithm which extracts repeated sequence through a pattern growth method. The algorithm can be depicted through the generated FP tree structure. The repeated sequences can be extracted by traversal of the FP tree.

**Table 4.1 Category data set**

<table>
<thead>
<tr>
<th>CID</th>
<th>Sub categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>{ p,q }</td>
</tr>
<tr>
<td>B</td>
<td>{ r,s,t }</td>
</tr>
<tr>
<td>C</td>
<td>{ p,r,s,t }</td>
</tr>
<tr>
<td>D</td>
<td>{ p,q,r,s }</td>
</tr>
<tr>
<td>E</td>
<td>{ q }</td>
</tr>
<tr>
<td>F</td>
<td>{ p,q,s }</td>
</tr>
</tbody>
</table>

Considering the category data set given in Table 4.1 with Category ID written as CID, the frequent pattern growth is depicted in Figure 4.1. The pattern grows by reading one category at a time and adding the value to node list. When repeated sub categories are found the patterns may overlap in the mapped tree.

From Figure 4.1, the (i) after scan of CID=A, \{ p,q \} is read and tree with two nodes p and q is generated. In (ii) after scan of CID=B, again nodes are generated. No common sub categories are found. In (ii) after scan of CID=C, since repeated items are there the count on the node gets incremented. Common prefixes are linked through paths.

The FP tree after scan of each category becomes more complex in structure. It has the main defect of FP tree may not fit in to memory for larger pattern databases. Our proposed method overcomes this by representation of a weight tree. The graph structure of the web page is represented as a weight
tree with associated weight on each node. We follow a user centric sequence generation to ease traversal.

Figure 4.1 FP Tree generation process
4.4 PROPOSED TREE BASED METHOD FOR VISUALIZATION

We propose a user-centric approach for the link hierarchy of a webpage. As a sample we analyzed the example of our virtual learning environment VL Schools. Total number of traversals of a signed user on a particular link is the main factor contributing to weight. The other factors considered for weight calculation is explained in chapter 3. The website is organized into different levels of hierarchy in the weight tree. Home page is set at the root level and then the hyperlinks in their hierarchical order.

![Diagram of VL School hierarchy](image)

**Figure 4.2 Link hierarchy representation of VL School**

Levels are marked as L1 to L4. In Figure 4.1, L1 is the homepage of VL Schools. At level 2 the four courses are given HTML, PHP, JSP and ASP and is marked as A, B, C and D respectively for convenience. Sub pages
of HTML namely courses on Introductory topics, Frames and Forms form the next conceptual level L2. For convenience they are marked as a1, a2 and a3 respectively. The in links, that is the number of incoming links to A, IN(A), in this case is found to be 1. Out links to Page A is found to be 3 in this case. Calculation based only on the in link and out link values and weights do not exactly converge to personalization of the page. This could contribute to page rate calculation but for visualization of the hierarchy as a tree structure which should help in easy navigation we adopted a different method.

In the traditional BFS method for traversal, an order is maintained with initial start at left child then traverse to right and then to grand children. We studied the user behaviour for a particular period of time from the segmented usage data with only needed fields for the particular user. As the time factor is also included, this yields ranking based on current user behaviour. This study based on the changing user behaviour over time gives more accurate page rates of user interest than in the other algorithms. This eliminates the possibility of giving high ranks to older pages.

Haibin Liu & Vlado Keselj (2007) proposed the idea of combined mining of web server logs and the contents of the retrieved web pages for the classification of navigation style of users. In our approach we are not depending on the server/usage log data alone in account of possibility of error in ranking due to false logins or crawlers. For ranking of pages, the count of number of visited pages are mined from usage log data rather than taking the number of logins. Sometimes users may login and can logout the session without browsing any page. Users may also login, do the necessary works on the site and leave the system without a proper logout. By taking the count of visited pages for weight calculation accuracy in calculation is maintained.
4.4.1 User Centric Page Rank Calculation

According to equation 4.1, Page Rate of the page a3 could be calculated as

\[ \text{PR(a3)} = \left((1-0.5)/18\right)+0.5\times(\text{PR(A)}\times(44/(44+72+63))) \]

We know that

\[
\begin{align*}
\text{PR(A)} &= 0.5/18 + 0.5 \times (\text{PR(S)} \times (110/(110+100+153+73))) \\
\text{PR(S)} &= 0.5/18 + 0.5
\end{align*}
\]

By solving the above equations we get

\[
\begin{align*}
\text{PR(S)} &\approx 0.52 \\
\text{PR(A)} &\approx 0.0934 \quad \text{and} \\
\text{PR(a3)} &\approx 0.0391
\end{align*}
\]

Page Rate of C which is the highest weighted node in the particular level is found to be

\[ \text{PR(C)} = 0.5/18 + 0.5 \times (\text{PR(S)} \times (153/(110+100+153+73))) \]

\[ \approx 0.1189 \quad \text{which shows page rate of C is slightly higher than that of A.} \]

We observed that the in link and out links to a particular node also can contribute to the ranking of pages. In such types of learning environments users travel forward rather than backward. Weight at a node is the particular user’s total traversals considering both forward and backward. Taking this into consideration we modified the equation for page ranking in our Easy User Navigation System by considering constants \( x \) and \( y \) for setting the importance of in links and out links respectively. The values of \( x \) and \( y \) is
set such as \( x+y=1 \). The in links are given higher priority in ranking so in normal cases \( x \) value is set to 0.75 and the value of \( y \) is set to 0.25. In our user centric approach user wishes to traverse forward rather than back links. So the links back to the node under consideration is given less weightage. It can not be ignored also. Thus it is calculated as the denominator with less value for the constant.

By considering these constants the modified equation for user centric page ranking is:

\[
UPR(A) = \frac{1-d}{N} + d \sum_{j=1}^{IN(A)} UPR(P_j) \frac{x \cdot n(P_j, A)}{\sum_{i=1}^{OUT(P_j)} y \cdot (P_j, P_i)}
\]

(4.2)

Where \( x \) and \( y \) are constants and \( 0 < x < 1 \) and \( 0 < y < 1 \) and \( x+y=1 \).

Since we consider the hierarchy for a particular user

\( n(P_j, A) \) - the total number of traversals by user from \( P_j \) to page \( A \).

\( \sum_{i=1}^{OUT(P_j)} n(P_j, P_i) \) - sum of all traversals by the user on all the out-links of page \( P_j \).

\[
\begin{align*}
UPR(A) &= 0.5/18 + 0.5 \cdot UPR(S) \cdot ((0.75 \cdot 110)/(0.25 \cdot (110+100+153+73))) \\
UPR(a3) &= 0.5/18 + 0.5 \cdot UPR(A) \cdot ((0.75 \cdot 44)/(0.25 \cdot (44+72+63)))
\end{align*}
\]

We know that \( UPR(S)=PR(S)=0.52 \)

Substituting we get

\[
UPR(A) \approx 0.2244
\]
UPR(a3) \approx 0.1104 \text{ which yield better weight and rank relationship.}

Similarly calculated the value of UPR(C) and value is found to be

UPR(C) \approx 0.3014

which is high as the weight. The computation of weights/rank is entirely dependent on the probability of visit of a particular link by the specific user. This strategy could be used for user centric ranking and thus could enhance user based ranking than the approach by Jianhan Zhu et al.(2004).

4.4.2 Tree Based Link Hierarchy Personalization

The displayed tree structure could be better visualized as personalized structure by the following procedure. Weights of a particular page are the mined server log values of visit of the user.

The proposed algorithm for personalizing the link structure is:

Input: The number of Levels, Nodes at each level and associated Weights

Output : Personalized structure in the ranked order

Step1: At level 2 a sort procedure is employed and categories are put in sorted order. The category with highest count is dynamically assigned as the first tab which personalizes the system.

Step 2: Let S1 be the first item at level2. Select the left child of S1 and traverse all the grandchildren.

Find the total count for each child.
Step 3: Repeat the step for right other grandchildren of S1

Step 4: The category with highest count is found and is highlighted to mark personalization/ ranking

This structure is displayed as hierarchical one at the left frame of the page to enhance navigation. On login itself the user will get an idea of his previous visit and the hierarchy of pages from the displayed structure.

When we get multiple pages with equal weights at level 1 after sorting the one which appears first is assigned the first position.

![Diagram](image)

**Figure 4.3 Personalized Link hierarchy of VL School**
Figure 4.4 Link Hierarchy with equal weights on multiple nodes

Figure 4.5 Link Hierarchy after personalization

In our method in the level1 ordered sub tree, total weights at sublevels of both nodes whose weight are same are calculated. The one with
The visits of the particular user as well as that of other users can be clustered and analysed. The pattern analysis and how this could be used for future prediction of visits is explained in Chapter 5.

4.5 COMPARISON WITH RELATED WORK

Comparison of our approach with related work is done in this section. First we compare the ranking of pages of our method with page rank and Jianhan Zhu’s method. Then we compare the traversal pattern and graph building with Breadth First Search method.

4.5.1 Comparison of the Proposed Work With Page Rate Algorithm

Page Rate algorithm which is an enhancement of the Page Rank algorithm ranks the pages by considering the in link and out link structures of a node. We compare our algorithm for user based page ranking with page rate algorithm in the following ways:

1. Both approaches ranks the pages based on authority. The damping factor ‘d’ in Page Rate algorithm is a random surfer’s pattern of search while in our approach it is the specific user’s navigation pattern.

2. In Page Rate algorithm change in user behaviour affects transition probabilities between web pages and hierarchy construction. Our algorithm focuses on the server logs to learn change in user behaviour and accordingly link structure reorganization/personalization is done.
Observing the fact that in a user centric perspective forward traversal has to be given more weightage than out links we have proposed the concept of constants for this in our system. In links are assigned higher constant value which gives better user navigation supporting results than the page rate method.

4.5.1.1 Result analysis

The results obtained with first pass of the algorithm is given in Table 4.2. The obtained page rate values with page rate algorithm and UPR method shows the fact that page values can easily be differentiated.

Table 4.2 Comparison of Page Rate and UPR method

<table>
<thead>
<tr>
<th>Page</th>
<th>PR</th>
<th>UPR (our method)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.0934</td>
<td>0.2244</td>
</tr>
<tr>
<td>B</td>
<td>0.0871</td>
<td>0.3015</td>
</tr>
<tr>
<td>C</td>
<td>0.1189</td>
<td>0.2058</td>
</tr>
<tr>
<td>D</td>
<td>0.0722</td>
<td>0.1626</td>
</tr>
<tr>
<td>E</td>
<td>0.0391</td>
<td>0.1104</td>
</tr>
</tbody>
</table>

The proposed method by considering constants x and y for setting the importance of in links and out links respectively, generates significant values to user centric page ranks and this effectively helps user in ranking pages according to his preference.

4.5.2 Comparison of our Link Structure Visualization Method With Breadth First Search and Other Shortest Weight Path Methods

In the Breadth First Search method of traversing a level based traversal is followed. Only after completing traversal at each level the nodes
in the next level are visited. In this method the total visits made by a user on a particular page is assigned as weight of that link. Related pages have the possibility of having higher values of weight between links. Breadth First Search is not a weight based balancing method. In BFS equal weights will be assigned to all the links. More over BFS method can only assign weights but can not be used for predicting user behaviour on a link.

The shortest weighted path method draws an inverse relationship between the weight and probability of the pages to be related. Here we select the minimum number of user traversals to form the graph. The method proposed by Thorup [2004] is a betterment to the shortest weight path method in comparison with time complexity but this also is inefficient to draw a perfect relationship with weight and user traversals. Yusi Wei[2014] proposed a improvement of Thorup algorithm by reducing the depth of component tree. This method also is efficient in terms of faster execution and finding of shortest path but does not satisfy our requirement of user personalization. When we get multiple pages with equal weights the one which appears first is assigned the first position. Both the nodes sublinks total weights are calculated then and the one with highest weight value is reassigned the first order.

4.5.2.1 Result analysis

Table 4.3 shows the experimental evaluation of the proposed method of weight tree traversal with the traditional BFS method. BFS needs complete level wise traversal for reaching the popular node. But on employing our method faster traversal results for the mostly visited page is achieved. It is observed that as level increases BFS produces very poor results compared to our method of traversal. Our methods reaches the mostly visited
node in almost half time compared to BFS. This enables the user faster navigation and thus better personalization.

Table 4.3 Comparison of the proposed method and BFS

<table>
<thead>
<tr>
<th>No of Nodes/level</th>
<th>45/7</th>
<th>58/7</th>
<th>90/11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traversal time in seconds for mostly visited page with BFS</td>
<td>54</td>
<td>60</td>
<td>125</td>
</tr>
<tr>
<td>Traversal time in seconds for mostly visited page with our system</td>
<td>38</td>
<td>40</td>
<td>71</td>
</tr>
</tbody>
</table>

Analysis of memory usage efficiency of VL Schools, a virtual learning system, having 1846 pages with 2409 links is given in Table 4.4. The usage log file is 350 MB in size. We constructed link hierarchies with BFS, Thorup method and our method. Our method gave more weightage to in links. The administrator of the system evaluated the three link hierarchies. Evaluation of desired page visit in terms of memory usage is given below.

Table 4.4 Analysis in terms of memory usage percentage

<table>
<thead>
<tr>
<th>Method</th>
<th>Pages/Level</th>
<th>25/4</th>
<th>150/7</th>
<th>178/7</th>
<th>520/9</th>
<th>891/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>BFS</td>
<td>Memory usage percentage</td>
<td>16</td>
<td>38</td>
<td>42</td>
<td>63</td>
<td>51</td>
</tr>
<tr>
<td>Thorup Methods</td>
<td>14</td>
<td>32</td>
<td>44</td>
<td>54</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>Our Method</td>
<td>10</td>
<td>24</td>
<td>36</td>
<td>38</td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>

It is observed that both the methods have more memory usage as the number of levels increase. Our methods perform better than the other two methods and is found to be efficient in memory usage as level increases.
When memory usage is measured with number of pages also our method is found to perform better than the other two methods.

4.6 CONCLUSION

This chapter describes our approach on link structure display for easy user navigation. From the usage log data total user traversals on each page is taken and is marked as the weight. The authors constructed the link hierarchy based on this data. Our algorithm for user centric page ranking giving more weightage to in links is also explained in this chapter. The results show that our method gives easily differentiable values of ranking. We tried to display the link structure sampled as tree. The displayed structure proves that user personalization could be better achieved with the proposed method. As a future work we wish to learn the influence of this User Centric Page Ranking on other websites. We wish to propose less complex algorithm for tree based personalized pages.