Volume 4, Issue 5, May 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Path Traversal Technique for an Efficient Web Navigation Mining

Sagar More*, Suraj Patil, Rohit Pachlor, Nihit Agrawal
Computer Engg. Dept.
MPSTME, NMIMS University
Shirpur Campus, India

Abstract— To improve web services Web Mining technology is applies. After applying web mining on web sessions we will get navigation patterns which are important for web users such that appropriate actions can be adopted. Mostly this pattern mining is success part of e-commerce and mobile commerce. Analysing this data will help the organizations to realize the lifetime value of their clients, and provide them with a more sophisticated structure of the web site and services. So Web Navigation Pattern can be easily shows the trends towards web pages that visited next for browsing session. Due to huge data in web discovery of patterns and there analysis for further improvement in website becomes a real time necessity. In our paper algorithm initiate from dataset consisting web sessions. From each web sessions we flow towards the navigation path which shows actual traversing of user in the website. Applying algorithm of construction of graph we will get initial graph showing each browsing session. Our aim is to improve mining efficiency by applying path traversal algorithm and get surfing pattern. Our algorithm modifies previous and also experimental results will show efficient surfing pattern which can be better for finding the user's interest. In modified algorithm we avoid unwanted and repetition of data. Here experimental analysis also use for improvement of web design.

Keywords — Web Navigation Mining, Frequent Path, Browsing Session, Surfing Patterns, Navigation Path

I. INTRODUCTION

As daily use of World Wide Web is increasing, mining of database is having more demanding. The database of web is in the form of web sessions with session id or session number. So for this type of mining we called Web Mining. And in the Web Mining when we uses to find path traversal pattern for decision management in website design, then it comes under the web usage Mining. Mining web traversal pattern is to discover most of access pattern of user from web sessions [1]. These patterns are mostly use in e-commerce or mobile commerce for improvement in web site design which shows the trends of user towards particular web site access. The pattern we get finds the correlation between web pages, and web files.

Following terms are uses to find web traversal pattern in the Web Navigation Mining.

Web session: It shows how the user visited from one web page to the next web page. We will consider dataset in the form of web sessions where each horizontal line is a web session.

In each web session, a single character e.g. A, B, C… etc. or a digit e.g. 1, 2, 22, 33, 121….etc. represent the web page.

Link: While user visited from one page to next web page then connection between two web pages is considered as a link. Consider user visits the *msn.com* and here first he visit to "front-page" and next visit to webpage "news" then connection between these two web pages in path traversal graph is represented as a Link for these two web pages. Remember "Link" for any path traversal graph is only for two web pages.

Via-Link: When one web user visited from one web page to next web page by using intermediate web page, then connection between these three web pages are shown by via-link in path traversal graph. Via-link always has three web pages out of which two are terminal web pages and one will be intermediate web page.

Let visitor visit the *msn.com* and firstly he reached to "front-page" then he wants to visit the technology page as "tech" but if he don't visit *tech* directly from *front-page* and he first visit *news* and then reach to *tech* then there will be concept of via-link is use.

Path Traversal Graph: It is the set of vertices where each vertex shows a web-page which are connected with each other by using link and via-link. Here condition is that $1 \leq vertex\ number \leq n$.

In the implementation of path traversal graph, links and via-link are associated with vertices which represent the web page. As per traversal time, each web page appears in traversal sequence and there is chance of repetition of web-page.

Contributions of this paper are described as follow:

In this paper we find the throughout surfing pattern from mining of path traversal graph. The TSP find proves more effective to predict one step forward visit to next web page. As we get predictions of visitor, website operator can

May - 2014, pp. 437-441

efficiently reconfigure the personalized website structure and take the help of throughout-surfing pattern for rearrange the contents of website. First we collect the user login and surfing session from web server. The surfing session contains the session id and consecutive browsing sessions. Each horizontal line in a session id shows the access pages of a particular website visitor. We draw the frequent path traversal graph which will shows all possible connection for each web page by using dataset of web browsing session. In the graph from to via-link information is important for user's to predict where a visitor will go at any vertex by the vertex he comes from. In proposed modified algorithm we will find TSP which avoids the repetitive database scan of data sessions and also neglect unwanted and duplicate data. Result of filtering module contain user navigation pattern in the form of hyperlinks.

II. RELATED WORKS

Mining path traversal patterns is to discover web access patterns in a distributed information-providing environment. M.S. Chen and et al. [2] proposed a method for mining path traversal patterns. In that paper author designed MF algorithm, where it uses concept of maximal forward references for original log sequence. Next they presented algorithm which determine the frequent traversal pattern by considering the maximal forward references which we got from last stage. They also found that frequent traversal patterns are almost similar to sequential pattern finding. Actually they are different from each other such that traversal pattern has sequences by maximal forward reference whereas customer sequence is considered for sequential pattern. Two algorithms FS (full-scan) and SS (selective-scan) are considered for finding the frequent traversal sequences where hashing technique is used and two operations i.e. prune and trim are performed.

FASTUP algorithm [3] proposed by Lin and Lee used sequential patterns which mainly works on transactions of database. Here previous one original database is kept as it is and new transaction added to update. Author uses the GSP algorithm which makes multiple passes over the database. In each pass frequent sequences are generated and support value is determined. So In FASTUP algorithm, database is scan and counts the supports for candidate *k*- sequences and also frequent *k*-sequences are generated. It partitions the candidate into two parts. In first part frequent sequence is appeared before database update. In second part sequence is not frequent before database updated. In first part only inserted transactions are scanned and non-frequent sequences are need to be prune the candidate of next length. In second part, new transactions and original are scanned to count the support for candidate.

J. Pei and et al. [4] proposed an algorithm, called WAP mine, which finds the access patterns from web logs containing web browsing session of particular visitor. The main task of WAP algorithm is to construct a WAP-tree. For construction of this WAP-tree need to scans the access sequence twice. Also WAP-mine supports conditional search for generation of conditional sequence base and conditional WAP-tree for mining of access patterns. Advantage of this algorithm is that, here WAP-mine not generates candidates and it avoids the problem of generation of more candidates in GSP [5] or FS (SS) algorithms.

Yen and Chen [6] adopted a graph-based approach to mine both association rules and sequential patterns. As mentioned in their conclusions, the graph structure may not fit in the main memory when the database is very large. In general, the database is huge and the algorithm is inadequate.

Also for finding access pattern of web user Markov chain model is most popular for characterizing sequential data [7]. Mainly this model is worked on the assumption that, next web page in session is subject to the value of previous session.

Incremental web traversal pattern mining algorithm [3] proposed by Jane Yen, Yue-Shi Lee mine web traversal pattern incrementally, they used previous mining result to discover new patterns such that the mining time can be reduced. So here focus only on choosing the well storage structure. This algorithm is used lattice structure to keep the previous mining results. In the lattice structure, only web traversal patterns are stored. To incrementally mine the web traversal patterns and speed up the mining processes, we extend the lattice structure to record more information. In extended lattice structure we can append support count which is used to calculate and accumulate support when incremental mining is proceeding. Though processing time is less, it acquires more space for storage because lattice structure is saved in hard disk level-by-level. For mentioning the relationship between the patterns found this algorithm is better one.

III. PROPOSED SYSTEM

When user surf on a website, its activities are recorded in the log files which are further useful for creation of Web browsing sessions for pattern mining task. Here dataset consisting web browsing sessions based on the maximal forward reference is the primary input for mining the pattern. In this paper we propose modified traverse algorithm which apply filtering process to extracted log from the data file, to remove unwanted and duplicate data. So that time and space complexity require is minimum as compared to previous algorithm i.e. apriori and traditional graph traversal algorithm.

In Apriori algorithm the huge set of candidate sequences is generated for large sequential database. The each sequential database has more detailed of access log. So it acquired more space causes the more complexity. Also to avoid repetition it requires the repetitive database scan. As it generates a combinatorial explosive number of candidates when mining long sequential patterns, it suffers from high memory load. Although algorithm for single database scan it may be difficult to hold all sequences of the database in the data structure.

For graphical representation of a website the directed graph contains edges and vertices corresponding to hyperlinks and documents respectively. By mining the throughout-surfing patterns, we can predict the next node to be visited, so there is need to know from where the visitor comes from. For that "from-to-via" link concept is used as we discussed already in

introduction. Also this via-link concept is mainly used for mining of TSP. Therefore novel data structure called path traversal graph is used that consisting set of vertices, edges, and via-links to store the information from web browsing sessions. Sometime this compact structure of path traversal graph is use help to improve the efficiency of TSP.

Our proposed modified methodology basically built from graph structure. This data structure mainly used for store and retrieve session information. In our system we collect user login or surfing session from web server. Each surfing session contain consecutive clicking of web pages of a particular web user. For storing user login and surfing session we use file system of the form csv or txt. Proposed system is work in two parts first part is graph construction and second one is graph traversal. In step of graph construction, we extract the log from the data file and apply filtering process to remove unwanted and duplicate data. Result of filtering module contain user navigation pattern in the form of hyperlinks. Each hyperlink is treating as node or vertex of graph. By using this session information we plot the graph which consisting the edges as connection between two vertex. Consider graph G having web browsing session (S_1, S_2, S_3….,S_n) where each S_1, S_2, S_3 is considered as vertex of graph and in real time these are actually web pages. Initially in graph building we find the root node, and then we find link between each node and these links are treated as edges of graph. After forming of edges we find via-link and using this via link we plot session surfing graph.

Following are the steps for Graph Construction:

1. Retrieve session values from the web server (this browsing session has consecutive sequence of web pages)
2. Filtering of unwanted and duplicate data
3. Sort the session according to the website pages.
4. Find session home page as a root node of graph.
5. Identify the vertex and edges for graph.
6. Identify the direct links and via links between vertex of graph.
7. Make join between vertices in the form of direct link and via link.

Consider the dataset which content user login session and their navigation pattern as shown in table 1. Each row of table shows the how web user visited the web page consequently.

Table 1: Web Browsing Session

Session ID	Web Browsing Session
S_1	(1, 3, 5, 6, 9, 11)
S_2	(1, 2, 5, 6, 8, 11)
S_3	(1, 2, 5, 7, 9, 11)
S_4	(1, 4, 5, 6, 9, 10)

By applying graph construction to above web browsing sessions we will get the following graph. In this graph dark lines shows direct links while thin lines shows via links. In graph links are always connection between two web pages and for via links there is three web pages are connected with each other. Graphical representation is shown in figure 2.

In Second step of our proposed system, first we consider path traversal graph and from that path traversal graph we calculate minimum support value for finding frequent path traversal graph. We used the breadth first search method for traversing the constructed graph. Following are graph traverse algorithmic steps for construction of path traversal graph:

1. Select root node same as in graph construction step and traverse descendant vertex for it.
2. Note descendant vertex of root node.
3. Dynamically calculate the minimum graph support to find frequent paths using distance finding methods.
4. Select the node of each via links having maximum value than the support vector.
5. By using this selected node draw frequent path.
6. Repeat graph construction method to show the graphical representation of sessions.
7. Repeat step number 3 to calculate minimum graph support.
8. Calculate via-links using support values.
9. Use recursive *tsp()* method to calculate throughout surfing pattern of graph.

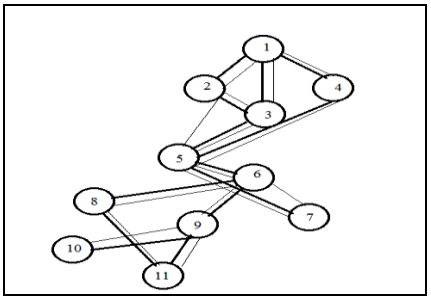

Fig. 2: Graphical representation of browsing Web Browsing Session

Following code show the *tsp()* function to calculate throughout surfing patterns.

```
Tsp()
{
for (start = 0; start <rootlinks; start++)
     {

       for (nodes= 0; nodes <totlnodes.size(); nodes++)
        {
            string []vailinks = totalnodes[j];

              for (int x = 0; x < vailinks.Length; x++)
                {
                  string  vailink = vailinks[x].Replace("(", "");
                  vailink = vailink.Replace(",", "");
                  int index = vailink.IndexOf(last);
                  if (vailink.Trim().Contains(last) && (index+1) != vailink.Length
                  && vailink.Contains (rootnode)==false)
                  {
                    string decendent = arr[i];
                    decendent = decendent + vailink.Substring(index + 1, 1);
                    result[i] = decendent;
                   }
                 }
          }

       }

          if (secarrat.Length != 0)
            tsp(secarrat);

 }
```

Fig. 3: Working of TSP function

IV. PERFORMANCE EVALUATION

Here we compare our proposed system with Apriori and Graph Traverse algorithm. We change the minimum support threshold and take the experimental results which are shown in following figure. While the minimum support becomes lower, the number of frequent patterns increases and the frequent patterns get longer. In Fig. 6, the execution time of our mining approach remains almost at the same level because it discovers the TSP by traversing the path traversal graph and almost all the run time is spent on constructing the path traversal graph. As the threshold value is less number of frequent pattern generation is more. As more frequent pattern, it shows maximum throughout surfing pattern. In short execution time varies inversely to support value of the algorithm.

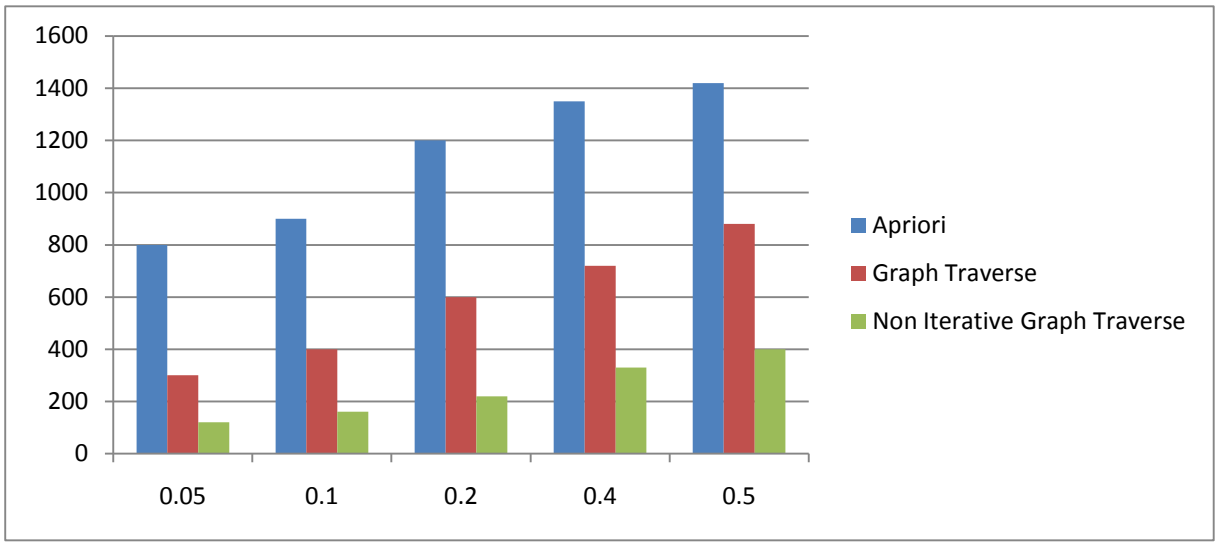

Figure: Performance Support vs. Execution Time

Scalability of the algorithms by varying number of web browsing session is illustrated in figure. Actually the execution time is directly proportional to number of browsing sessions. As the length of web browsing session or dataset increases, the cost of scanning dataset also increase. Same scenario is confirmed in experimental result. We will consider the

May - 2014, pp. 437-441

number of web browsing session on X axis and execution time required is on Y axis. Graph shows that our Modified algorithm is better than the traditional apriori algorithm and graph traversal algorithm.

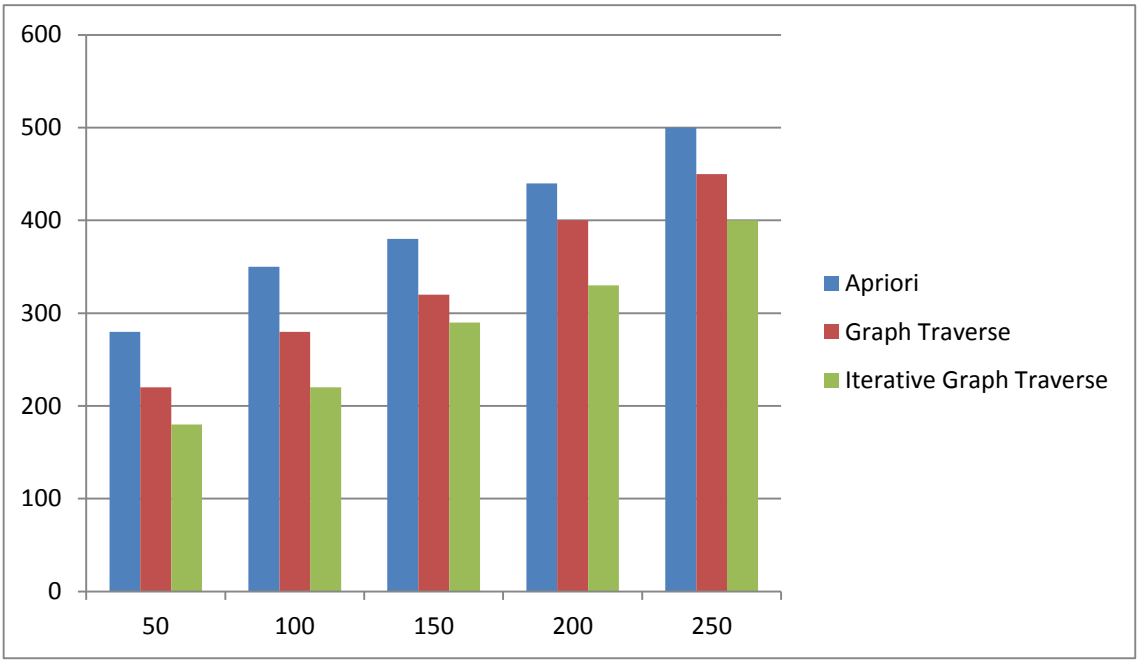

Figure: Performance of Browsing session vs. Execution Time

V. CONCLUSION

In our proposed system, mining web navigation patterns efficiently is the main goal. Here we first clear the concept of throughout surfing pattern which predict the path of website visitor. In next part we apply modified graph traverse algorithm to make mining from TSP in efficient manner. The modified algorithm use the filtering technique which removes duplicate and unwanted data in web browsing session and show the effectiveness as compared to formal algorithms. The experimental results proves that our modified graph traversal algorithm has better performance than the formal algorithm i.e. apriori and formal graph traverse.

REFERENCES

[1] Show-Jane Yen*, Yue-Shi Lee* and Min-Chi Hsieh, "An Efficient Incremental Algorithm for Mining Web Traversal Patterns", *Proceedings of the 2005 IEEE International Conference on e-Business Engineering (ICEBE'05)*

[2] Ming-Syan, Jong Soo, Philips Yu., "Efficient Data Mining for Path Traversal Patterns", *Proceeding of 1998 Knowledge and Data Engineering, IEEE Transactions on (Volume: 10, Issue: 2)*

[3] Ming-Yen Lin and Suh-Yin Lee, "Incremental Update on Sequential Patterns in Large Databases", *Tools with Artificial Intelligence, 1998. Proceeding of Tenth IEEE International Conference.*

[4] Jian Pei, Jiawei Han, Behzad and Hua Zua "Mining Access Patterns Efficiently from Web Logs", *Proceeding in 2000 on National Sciences and Engineering Research Council of Canada, Hewlett-Packard Lab.*

[5] R. Srikant, Rakesh Agrawal, "Mining Sequential Patterns: Generalization and Performance Improvements", *IBM Almaden Research Center (1996).*

[6] Show-Jane Yen and Arbee L.P. Chen "A Graph-Based Approach for Discovering Various Types of Association Rules", *IEEE Transactions on Knowledge and Data Engineering, Vol. 13, No. 5, September/October 2001.*

[7] Anna Gutowska and Luis L. Perez "A Comparison of Methods for Classification and Prediction of Web Access Patterns", *COMP540 - Final Report - Spring 2010.*

[8] Arthur.A.Shaw, N.P. Gopalan "Frequent Pattern Mining of Trajectory Coordinates using Apriori Algorithm", *proceeding in International Journal of Computer Applications in volume 22-9, May 2011.*

[9] Jatin Chhugani, Nadathur Satish, Changkyu Kim, Jason Sewall, and Pradeep Dubey "Fast and Efficient Graph Traversal Algorithm for CPUs: Maximizing Single-Node Efficiency", *2012 IEEE 26th International Parallel and Distributed Processing Symposium.*

