Reference Scan Algorithm for Path Traversal Patterns

Chintandeep Kaur
Department of Computer Science & Engineering,
Thapar University, Patiala –147004 (India)

Rinkle Rani Aggarwal
Department of Computer Science & Engineering,
Thapar University, Patiala –147004 (India)

ABSTRACT
The paper focuses the path of exploring an algorithm which involves mining frequently used user traversals pattern while using the World Wide Web by the user. As the web's becoming more and more popular, web servers gather the large amount of data from the web in the form of web server logs. The web server log provides important information aiding to analysing the user surfing behavior and thus extract important information. Web Usage Mining analyses the Web access logs to find how the users surf a website so as to bring out interesting patterns which can be used for improving user profile, web site design, business and marketing decision support and web server system. Firstly, the server log is scanned and the maximal forward references are obtained for which effect of backward references is not considered and secondly, the maximal forward references obtained in the first step are used to finally obtain the large references which are frequently accessed patterns by the user.

General Terms
Maximal forward reference, Algorithms, Large reference, user-Traversal patterns

Keywords
Reference scan, Server log.

1. INTRODUCTION
As the popularity and vastness of World Wide Web is increasing with each day so is the importance of web mining. With the increasing number of websites and web users, web data is being collected and stored by the server as web server data constituted with different fields. It is considered that the analysis of this web server data can provide us various information's like user surfing behavior which can help in user profiling, web site designs and making better business and marketing decision making our website more popular and user friendly [1]. For performing this task it is necessary to collect a good amount of data for analysis before coming to any productive conclusions. As the web server data tends to be too large there is a need devise an efficient algorithm to first extract useful data and then mine it to get patterns which are helpful for the website. This paper has considered a new data mining capability which constitutes of mining the access patterns where the objects are linked with each other giving an interactive access in a distributed information providing environment say World Wide Web (WWW) [2] where the users access the websites by travelling from one page to the other with the help of connecting facility provided say hyperlinks. The fact that mining of user traversal patterns will not only help in improving our web-site's design (say, more user friendly for most used pages, better designs of pages) but also help in making business decisions (say placing of advertisement at appropriate page). The user access patterns or mining traversal patterns provides with frequently used pages which are done by analyzing the user behavior from the web server log. Various algorithms have been proposed for mining the user traversal pattern but in this paper algorithmic aspects proposed are in such a manner that it makes traversal pattern mining much better [3].

As these information giving services are expanding with each day with fierce competition to face with analysis of user access patterns is a necessity so as to come to qualitative conclusions with meaningful patterns to end with and thus providing services which help in achieving user satisfaction. The user can traverse the web-site in different ways say for instance it may reach web-page because of its location rather than its content or it does not follow the hierarchy and randomly traverses the web site and reaches that web page. The differences between traversal patterns increase the complexity of obtaining useful and important information from the traversed data. Thus came the need for drawing various algorithms for mining the traversal pattern of the user and incurring useful patterns. Various algorithms have been proposed for mining traversal patterns. In this paper a new approach has been proposed for mining the large references. The traversal patterns are achieved first by mining the maximal forward references from the web server log and after this the maximal forward reference are used to obtain the large references which are the most frequently used paths by the user for a particular website. Eventually the aim is to obtain the large reference.

This paper basically contains a proposal of a new algorithm for getting the large reference sequence by first obtaining the maximal forward reference using the algorithm MF for obtaining traversal subsequences from the log data and later on applying a new algorithm being proposed called as reference scan to obtain the frequently occurring traversal patterns. Every traversal subsequence makes up a maximal forward reference representing the initial point from where the user starts accessing the website. The sequence obtained from the log data is raw and it is finally converted to maximal forward references which is the maximum path travelled leaving out the backward references. The backward reference is the path from where user goes to a link already visited which is only meant for the user to traverse the website easily and thus to obtain user accessed pattern. After this step an algorithm is devised to get the final set of large reference sequence which is derived from the maximal forward reference obtained in the former algorithm. The large references are those sequences which is the path most frequently visited by many users in the database. These large references are the final sequence which can be used for further analysis and improvement of website and web-data. Also these sequence need to be consecutive in maximal forward reference itself. Now for fetching large reference sequence or frequently used sequence an algorithm has been proposed namely Reference-Scan or RS which basically works around minimum support required reducing the database scans and finally giving us the result.

Problem formulation is given in section 2; section 3 contains the algorithm for user access pattern where algorithm MF to identify maximal forward references is described. Section 4 contains finding large reference with section 4.1 explaining our proposed algorithm and finally section 5 contains summary of our work.

1.1 Related work
Mobasher [4] gave a new web mining technique WEBMINER which offered transaction models to extract useful information

from the server logs. Spiliopoulou and Faulstich [5] collected the individual routing paths into an aggregated tree and pruned the uninteresting patterns considering only those patterns which had desired characteristics. Chen [6] gave two algorithms for determining web traversal patterns: FS (full-scan) and SS(selective scan).H. Yao [7] proposed a foundational approach to mining item set utilities from databases. This approach allows user preferences of item set as subjective values. The objective value of an item is defined according to the information stored in a transaction i.e. the quantity of the item sold in the transaction. From very large logs, Z. Chen [8] proposed two effective algorithms for finding maximal forward references longest sequences of Web pages visited by a user without revisiting some previously visited page in the sequence, and their performance is relatively analyzed. An efficient web traversal pattern mining algorithm based on suffix array is given by T. Jing [9]. In [10] Yen presented the modified incremental data mining algorithm for discovering web traversal patterns when the user sequences are inserted into and deleted from original database. This algorithm uses lattice structure to keep the previous mining results such that just new candidate sequences need to be computed. Hence, the web traversal pattern can be obtained rapidly when the traversal sequence database is updated. But it is unsuccessful when web site structure is changed. Zhou [11] proposed high utility path traversal pattern mining, which introduces the concept of utility into path traversal pattern mining model. A utility-based algorithm for web path traversal was improved by C. F. Ahmed et al. [12]. They used a pattern growth sequential mining to prune a huge number of candidates. It effectively divides the search space by small projected databases recursively using the divide and conquers technique. Therefore, it saves several scanning of the whole database which is required by the exiting algorithm.

2. PROBLEM DESCRIPTION

As it has been told before that World Wide Web is made up of incredible amount of information which is in a hierarchical structure where the pages are considered as nodes and they are linked with each other via hyperlinks represented using arrows and the users can move forward and backward with the help of hyperlinks and icons rendered to them. Some pages may be visited again by the user because of the location of the web page than its content, say for example to visit a sibling node the user usually uses the backward icon and then forward icon instead of going straight to the URL. So finally to get useful traversal patterns of the user from the server data the need is to prune the effect of these backward visits and extract the useful patterns of importance. In this paper it is being assumed that the backward traversals are just for the ease of moving to the previous page and the main concentration is on the discovery of meaningful forward user access patterns. A backward reference is basically for accessing the previously visited page by the same user. When a previous access is made by the user the path user was moving on terminates. This gives us a forward reference path which we term as maximal forward reference. After obtaining the maximal forward reference it starts again from the starting point from where we started to obtain the forward reference path and then resume in obtaining a new user traversed path. Also if a null source node appears termination should be made and a new path should be found again.

Now, moving on to the explanation of the algorithm (MF) used to obtain the maximal forward references described in the next section with help of an example explained below. As said that the web pages also known as the nodes and here are denoted with alphabets which are used to represent the user accessed path.

Suppose the log contains various user traversed path say : {A,B,C,D,E,F,G,H,I,J} as in Figure 1. Now, after applying the algorithm MF the following maximal forward references are obtained as output that have been accessed by the user {ABCD, ABEFG, ABEH, AIJK}.After obtaining the maximum traversed patterns the frequently occurring substrings from the maximal forward references are obtained. The frequently occurring substring is known as large reference. A large reference sequence is a traversal pattern which appears number of times.

The overall procedure is as follows:
Step 1: Determination of the maximal forward reference using algorithm MF is done.
Step 2: After this extraction of the large reference sequence using the algorithm RS is done.
After performing the above two steps the large references are obtained which are frequently occurring sequences.

3. ALGORITHM FOR USER ACCESS PATTERN

In this section the maximal forward reference are obtained using the algorithm MF with explanation of the procedure followed for the same. Maximal forward references are those strings which tell the maximum length till which the user has accessed a particular website before it traversed back to a previously accessed webpage. For this an already proposed algorithm MF has been taken under consideration. A server log consist of various fields as Host, rfc931 username, date-time, request, status-code, bytes, referrer, user-agent.. Now a pair of source and destination field is obtained from the server log and this pair can help in getting the maximal forward references. First, the server log database is sorted on the account of user id's for each user where the (source, destination) pair is ordered by time. The output obtained from MF is stored in a database DF as this output will act as input to the algorithm for reference scan giving us large reference sequence.

3.1 Finding maximal forward references

Algorithm *MF:*

Step 1: Set $i = 1$ and string Y to null for initialization,
where string Y is used to store the current forward reference path. Also, set the flag $F = 1$ to indicate a forward traversal.

Step 2: Let $A = si$ and $B = di$.
If A is equal to null then
/* this is the beginning of a new traversal */
Begin
Write out the current string Y (if not null) to the database *DF;*
Set string $Y = B$;
Go to Step 5.
end

Step 3: If B is equal to some reference (say the j-th reference) in string Y then
/* this is a cross-referencing back to a previous reference */
begin
If F is equal to 1 then write out string Y to database D_F;
Discard all the references after the j-th one in string Y
$F = 0$;
Go to Step 5.
end

Step 4: Otherwise, append B to the end of string Y.

/* we are continuing a forward traversal */
If F is equal to 0, set F = 1.

Step 5: Set i = i+l. If the sequence is not completed scanned then go to Step 2.

Table 1: An example execution of MF

Sequence	String Y	Database D_F
1	A	-
2	AB	-
3	ABC	-
4	ABCD	-
5	ABC	ABCD
5	AB	-
6	ABE	-
7	ABEF	-
8	ABEFG	-
9	ABEF	ABEFG
10	ABEH	-
11	A	ABEH
12	AI	-
13	AIJ	-
14	AIJK	-
15	AIJ	AIJK

Now, analyzing the above given table as per the traversal pattern in figure 1, it can be seen that while moving a backward traversal is obtained in sequence 5 resulting in ABCD as the first maximal forward reference written to database (as in step 3). The scanning of the pattern is done till the reverse sequence is encountered and then the final sequences are obtained which are written to string Y and ultimately fed and stored to the database DF. The final sequences from the figure 1 are {ABCD, ABEFG, ABEH, AIJK}.

4. FINDING LARGE REFERENCE

For obtaining the large references various algorithms have been proposed which work on the technique of hashing and pruning. For this paper firstly two algorithms were analysed which were Full Scan and selective Scan. Both these algorithm work on the concept of data hashing and pruning which basically involves scanning of the database with trimming of the database in each step. During the analyses it was observed that the database scan in these algorithms occur at each step. So as to reduce the database scan occurring at each step idea of this new algorithm has been proposed.

In full scan the database is scanned in each pass so as to get the candidate references whereas in selective scan the large references can be determined in batch so as to reduce the number of database scan that are made. After obtaining the maximal forward references in the database DF, now the large references are derived which are the most frequently occurring patterns that are traversed by the user many a times. The large references are the frequently occurring subsequences of maximal forward reference. For a sequence to qualify as large reference sequence it needs to have a minimum support which is a variable and can change with the size of the database fed as input by the user.

4.1 Algorithm on selective scan (SS)

To describe algorithm FS, we shall first get the basic ideas of the DHP algorithm. DHP [13] utilizes a hashing technique and so is very efficient in the generation of candidate itemset (C_k). Also DHP assists with pruning techniques which reduces the database size considerably. Lk denotes the set of all large k-references and ck is a set of candidate k-references, ck is usually a superset of Lk**.** By scanning through the database D_F**,** FS gets L1 and makes a hash table (i.e., H*2)* to count the number of occurrences of each 2-reference. Similarly to DHP, starting with k = 2, FS as in figure 2 generates ck based on the hash table count obtained in the previous pass, determines the set of large k-references, reduces the size of database for the next pass, and makes a hash table to determine the candidate *(k* + 1)-references. Database size can decrease significantly with each pass. While devising the algorithm it was found that it is better to obtain the Ck from Lk-1*Lk-1 rather than using hash able to generate Lk. To count the occurrences of each A-reference in ck to determine Lkj we need to scan through a trimmed version of database DF**.**

From the set of maximal forward references, among k-references in Ck large k-references are obtained. After the scan of the entire database, those k-references in ck with count exceeding the threshold become Lk**.** If Lk is non-empty, the iteration continues for the next pass, i.e., pass k + 1. Same as in DHP, every time when the database is scanned, the database is trimmed by FS to improve the efficiency of future scans.

4.2 Algorithm on selective scan (SS)

Algorithm SS is similar to algorithm FS in that it also employs hashing and pruning techniques to reduce both CPU and 1/0 costs, but is different from the latter in that algorithm SS, by properly utilizing the information in candidate references in prior passes, is able to avoid database scans in some passes, thus further reducing the disk 1/0 cost.. Recall that algorithm FS generates a small number of candidate 2-references by using a hashing technique. In fact, this small C2 can be used to generate the candidate 3-references.

5. PROPOSED ALGORITHM

As in the above section an overview about the full and selective scan algorithms has been given which basically rely on data hashing and pruning where it scans the database at each step so as to create a candidate table (C_1,C_2,C_3…C_n) and after that by trimming the value with minimum support it obtains the large reference table (L_1,L_2,L_3…L_k). Now the second candidate table is obtained by the cross product of large reference table obtained in step 1 and the occurrence of the two item set in main table is considered. After this removal of the items having the least minimum support is done so as to obtain the trimmed database. The further procedure involves generation of next item set and so on. This algorithm uses hash table to store the values. In full scan the database is scanned in each step so as to generate both candidate and large reference table but in full scan the database scans are reduced so as to produce the large references in batches.

As it is known to us that the website is built in a hierarchical structure starting with the initial webpage and then the rest of the webpage's are connected via the hyperlink. The information mostly lies on the last level of the tree structure with the previous levels containing the folders and the sub folders. Keeping this fact in mind an algorithm has been devised known as reference scan in which the last level is from where we start and move towards the upper levels so as to get our large reference sequences. The algorithm is reducing the database scan and is making our search more efficient and fast. The proposed algorithm is presented below and it is illustrated with an example in the coming section.

5.1 Algorithm Reference Scan (RS)

Input – Array of structure of TID, seq, min supp
Output -- Large reference

1. Find max i.e. maximum length of seq from inputs.
2. Repeat for flength from max to 1
3. Find number_of_sequences from flength

4. if(number_of_sequences<minimumsupport)
5. continue;
6. else
7. Create subsets of all web-pages in forward direction of length equal to flength.
8. Compare all subsets with all input to find occurrence of subsets
9. Get subset with maximum occurrence and occurrence>minimum support
10. End if-else
11. End loop

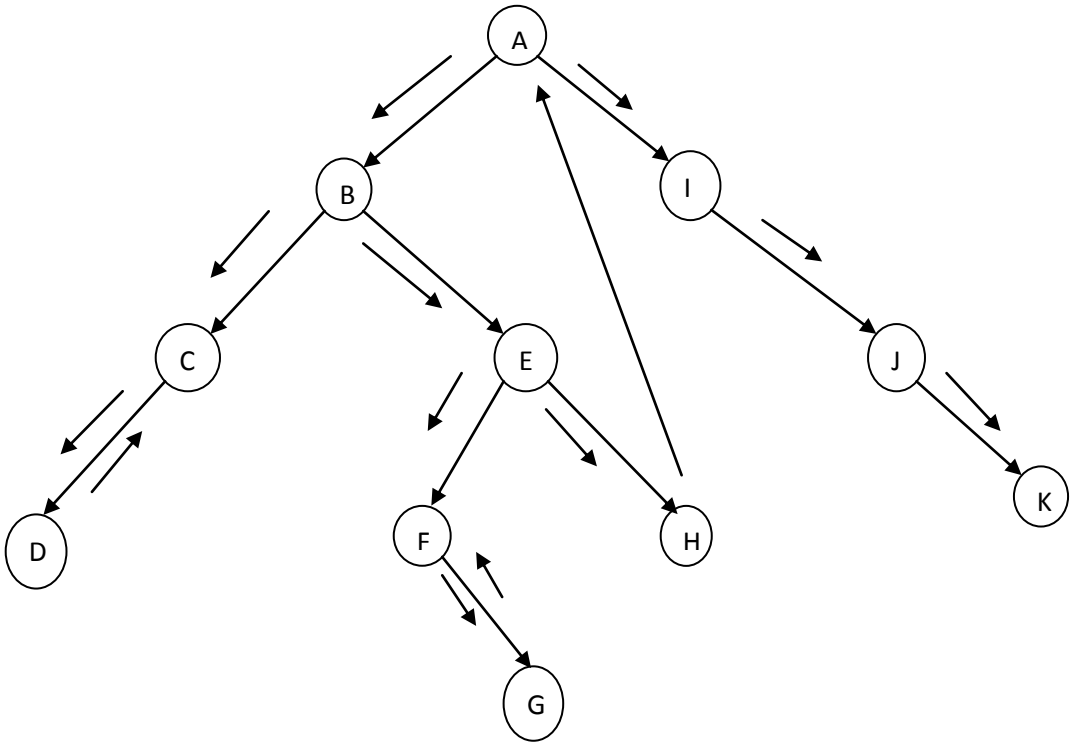

Figure 1: User traversal graph

C_1

TID	Database
100	ABD
200	BCDE
300	ABCDE
400	CDE

Scan D

Itemset	Supp.
{A}	2
{B}	3
{C}	3
{D}	4
{E}	3

L_1

Itemset	Supp.
{B}	3
{C}	3
{D}	4
{E}	3

C_2

Itemset
{BC}
{BD}
{BE}
{CD}
{CE}
{DE}

Scan D

Itemset	Supp.
{BC}	2
{BD}	3
{BE}	2
{CD}	3
{CE}	3
{DE}	3

L_2

Itemset	Supp.
{BD}	3
{CD}	3
{CE}	3
{DE}	3

C_3

Itemset
{BDE}
{CDE}

Scan D

C_3

Itemset	Supp.
{BDE}	2
{CDE}	3

L_3

Itemset	Supp.
{CDE}	3

Figure 2: Example execution of full scan

5.2 ILLUSTRATIVE EXAMPLE

The algorithm proposed in this section consists of various steps which will be explained one by one. Now known that web has a tree like structure with pages being represented as nodes denoted by alphabets and the hyperlinks represented by arrows. The algorithm starts by taking an example consisting of two fields, transaction id (tid) and the nodes that have been accessed by user during that transaction.

Table 2: Example execution of RS

Tid	Database
100	ABCD
200	ABEFG
300	ABEH
400	AIJK

The above table is the database output consisting of maximal forward reference obtained from algorithm MF which acts as input as an example execution explanation for the algorithm. This table contains the user accessed path as :{ A, B, C, D, E, F, G, H, I, J, K}

Step 1

Table 3: flength count

Tid	Database	Flength
100	ABCD	4
200	ABEFG	5
300	ABEH	4
400	AIJK	4

In the first step of the algorithm the number of nodes denoted by alphabets here that are accessed during each transaction by the user are counted say for instance in the first transaction with Tid 100 there are 4 nodes that are being accessed, in next transaction with Tid 200 there are 5 nodes again, so on and forth. After recording the Flength of each transaction next step is executed

Step 2

Table 4: occurrence of flength

Count(Flength)	Value
Count(1)	0
Count(2)	0
Count(3)	0
Count(4)	3
Count(5)	1
Count(6)	0
Count(7)	0
Count(8)	0
Count(9)	0
Count(10)	0
Count(11)	0

After step 1 where flength value is obtained for each transaction the next involves obtaining the number of occurrences of each flength implying we will count how many times the string with flength value 1 occurs as it can be seen that there is no transaction containing only 1 node so its value is 0. Now, check the occurrence of flength value 2 and as it can be seen in table 4 there is only 1 such transaction with tid 400 that contains 2 nodes. After doing this count the occurrences of flength value 3 which as can be seen in table 4 is two transactions containing 3 nodes till the value of flength reaches 11. As in all there are 11 nodes that can be accessed by the user.

Step 3

As the occurrence of different values of flength has been obtained in the above step, now the flength value count will be compared with minimum support value. A minimum support value is the one for which the count of flength has to be greater so as to qualify to be analysed further. The minimum support value keeps on increasing and is user defined. First of all the maximum flength value is taken and its occurrence is also considered and compared with minimum support value. Now starting with 11 its occurrence is 0 and is compared with minimum support value 1 (assumed). After this the count for occurrence of flength value keeps on reducing and each statement is false until we reach count for flength value 5 which is 1,after comparing this value with minimum support the value is still false. Now reduce the count, occurrence of value 4 is 3 which is higher than the minimum support.

Step 4

After obtaining the flength value which has more occurrences than the minimum support, create the subsets for the same order say in this case subset of the order 3 will be created in forward direction of the traversal: {A, B, C, D, E ,F, G, H, I, J,K}.The subsets will be as follows:{ABC, ABD, ABE, ABG, ABH,ABI,ABJ,ABK,ACD,ACE,ADE,AEG,AGH,AHI,AIJ,AJK,ABF,AEF,AFG,ACF,ADF,AFH,AFI,AFJ,AFK,BCD,BCE,BCG,BCH,BCI,BCJ,BCK,BDE,BEG,BGH,BHI,BIJ,BJK,BDE,BDG.BDH,BDI,BDK,BDJ,BEH,BEI,BEJ,BEK,BGH,BGI, BGJ,BGK,BHJ,BHK,BCF,BFG,BEF,BDF,CDE,CDG,CDH,CDI,CDJ,CDK,CEG,CGH,CHI,CIJ,CJK,CDF,CEF,CFG,CFH, CFI,CFJ,CFK,DEG,DEH,DEI,DEJ,DEK,DGH,DHI,DIJ,DEF, DFG,DFH,DFI,DFJ,DHK,DJK,EHI,EIJ,EJK,GIJ,GJK,EGH,EGI,EGJ,EGK,EFG,EFH,EFH,EFI,EFJ,EFK,GHI,GHJ,GHK,HIJ,HJK,CEG,CEH,CEI,CEJ,CEK,CGI,CGJ,CGK,IJK}.

Step 5

Each of the traversal patterns subset created above will be taken and matched to the database one by one. Also the occurrence of each and every subset will be counted in the database and the subset with a value greater than minimum support qualifies to be the final useful pattern for the website user and the owner. Now after comparing the above given subsets with our database in table the pattern ABE occurs twice and is the final pattern and the result.

6. CONCLUSION

Finally, after implementing the proposed algorithm and the existing algorithms the results have been shown in tabular form as depicted in table 5.

7. SUMMARY

As World Wide Web is becoming popular each day, mining the data from web is becoming tough. So as the mined data are useful patterns certain algorithm needs to be devised. In this paper useful user traversed pattern are first extracted from the server log and from that we obtain the maximal forward references using the algorithm MF which is the maximum path accessed by the user of a website ignoring the affect of backward traversal. After this the large references using the algorithm RS are obtained which are the most frequently occurring user traversal pattern. Both the algorithm have been explained with examples and as the comparison of our algorithm is made with the existing algorithm it is found that

the proposed algorithm is computationally more efficient and less database scans need to be made.

Table 5: Comparison of Proposed and Existing Algorithm

FACTORS	Full/Selective scan	Reference scan
Consecutive Patterns	This algorithm only works on consecutive patterns	This algorithm also considers links which are not consecutive.
Database Scan	With each pass database scan need to be done	Database scan needs to be performed at step 4 where the subsets are compared with the database.
Database size	It works well only on small size database; with large size of database it is not that efficient.	It works well both on small and large databases but its performance reduces with large size databases
I/O Overhead	This algorithm incurs us high I/O costs	This algorithm does not bring us high I/O overheads as there are less database scans.
Database	As this algorithm uses DHP technique with each pass the size of the database reduces	There is no reduction in the size of database.

8. REFERENCES

[1] Behzad. M.A. 2001. Discovering and mining user web-page traversal patterns. Master's Thesis, Simon Fraser University

[2] J. December and N. Randall, 1994 "The World Wide Web Unleashed". SAMS Publishing.

[3] Park, J.S., Chen, M.S. and P. S. Yu, P.S. 1998. Efficient Data Mining for Path Traversal Patterns. In IEEE transactions on Knowledge and Data Engineering, 10(2), 209-221.

[4] Mobasher, B.,Jain, N. and Han, E.H. and Srivastava, J. 1996. "Web mining: pattern discovery from World Wide Web transactions, Technical Report: TR96-050, University of Minnesota

[5] Spiliopoulou, M., and Faulstich, L. C. 1999. Wum: A web utilization miner. In the proceedings of EDBT Workshop on the Web and Data Bases (WebDB'98), Springer Verlag, 109-115.

[6] Park, J.S., Chen, M.S. and Yu, P.S., 1996. Data Mining for Path Traversal Patterns in a Web Environment. In proceedings of 16th international conference on Distributed Computing Systems, 385-392.

[7] Yao, H. , Hamilton, H. J., and Butz, C. J. 2004. A Foundational Approach to Mining Itemset Utilities from Databases. In the Proceedings of the 4th SIAM International Conference on Data Mining, 482-486

[8] Zhixiang, C., Fowler, R.H. and Fu, A.W.-C.2003. Linear Time Algorithms for Finding Maximal Forward References. In the proceedings of Information Technology: Coding and Computing, 160-164

[9] Jing, T., Zou, W.L. , and Zhang, B.Z. , 2004 An Efficient Web Traversal Pattern Mining algorithm Based On Suffix Array. In the Proceedings of the 3rd International Conference on Machine Learning and Cybernetics,1535-1539

[10] Yen,S.J., Lee,Y.S., and Hsieh, M.C. 2005. An efficient incremental algorithm for mining Web traversal patterns. In the Proceedings of the 2005 IEEE International Conference on e-Business Engineering (ICEBE'05), 274-281.

[11] Zhou, L., Liu, Y., Wang J., and Shi, Y. 2007.Utility-based Web Path Traversal Pattern Mining. In the Proceedings of Seventh IEEE International Conference on Data Mining Workshops, 373-378.

[12] Ahmed, C. F., Tanbeer, S. K., Jeong B.S. and Lee, Y.K. 2009.Efficient mining of utility-based web path traversal patterns. In the proceedings of 11th International Conference on Advanced Communication Technology (ICACT'09), 2215-2218.

[13] Park, J.S., Chen, M.-S. and Yu, P.S. 1995. An Effective Hash Based Algorithm for Mining Association Rules. In Proceedings of the 1995 ACM SIGMOD international conference on Management of data SIGMOD '95, 175-186.