# Web Usage Mining: An Approach

Pooja
Assistant Professor
Rakshpal Bahadur Management Institute
Greater Noida

## ABSTRACT
PL-WAP tree mining algorithm was a great improvement over the WAP tree algorithm which was earlier used for mining in the server web logs. This was basically done to find out frequent sequence patterns. In WAP-tree mining, we used to engage in continuous reconstruction of intermediate WAP trees for each level of recursion to extract frequent patterns. In the WAP technique we could not formulate a method by which we could infer which sequences would be the suffix sequences of a given last event. This was the major drawback of the algorithm. In order to solve this problem, binary codes were introduced to uniquely represent the exact position of each and every node in the WAP-tree. Whereas PLWAP tree is not efficient when the server log data gets frequently updated.Therefore incremental data mining approach is needed to handle this problem. PL4UP proposed later is an incremental mining methodology added to the PLWAP. But the numerous parameters considered, sets generated, insertions and deletions performed at each stage makes it less efficient. and it works efficiently only when the percentage updation in size is less than fifty. In this paper, a novel method is proposedto efficiently handle the updations right from the pre-processing stage and the key factors in the implementation of the method are described. A web usage mining tool was also developed to experimentally validate the method.

## General Terms
Sequential mining, sequences, patterns

## Keywords
Web mining, incremental data mining, frequent nodes

## 1. INTRODUCTION
Web data mining is an interesting area of web mining which analyzes data from web related data and discoversuseful knowledge from it. Web usage mining is a kind of web mining. The other types being web content mining and web structure mining. The former deals with analyzing web data according to the data content found in the various kinds of web sources. While web structure data looks into the hyperlink connected structure of the internet to find out useful pattern relations. Web usage mining is a form of web data mining that deals with the practices of the various data mining methods to find out useful web navigation patterns from the web server log file data. Web usage mining could contribute greatly to understand and provide service to the various needs of Web oriented applications way better. In today's e-commerce era, the process of effectively analyzing user browsing sequences is critically important. In many cases, website owners often provide the intra navigation scopes in an ineffective manner. Often they turn out unsuccessful in their initial organization of the contributing pages. This leads to poor product response from the customers' part. The key patterns generated could be cleverly used for customizing a set of resources on a server such as web pages, files etc aiming prospective customers. Help from marketing experts could be taken in casting suitable adverts in appropriate places so as to enhance the publicity of the products to potential customers. Mainly web usage mining is done on web server log data. Every web server stores its log information in one of the standard log file formats recommended. The web log usually contains fields like URL, internet protocol address, date and time, http request etc. This data normally contains unwanted information- noise. Since web usage mining is very much sensitive to data with noise contained in it, we have to pre-process the data acquired according to the algorithms we intend to apply.

## 2. METHODOLOGY
The initial task in the development of web usage mining tool is to preprocess the input log file. The file could be available in many different formats. There are many standard formats of server log files available today. The major drawback of the existing model was mentioned earlier, that is the problem with updations. Every time updations take place in the web log, we need process and append that to the existing WASD, then apply the support value to the WASD again. Then again from the scratch, we need to build the tree. This is because, the earlier tree which was constructed only had nodes which were frequent, but after updations some nodes which were found to be frequent would not be frequent anymore and also new nodes might have become frequent after the updations have taken place. For improving efficiency in case of updations, an incremental model is proposed which could maintain the tree as it is without rebuilding it each time. This is achieved by building the tree initially with all the nodes which may be frequent or not frequent. Then each time the log file is updated, the added transactions are first sessionized and added to the existing WASD. Now each new transaction is added to the existing tree. The challenge is, how to avoid the nodes which are not frequent while mining. We need to ignore those nodes while mining. After all the resulting generated patterns should be the same. This is achieved by maintaining a binary value in the event-node queue. Every time updations take place, this binary value gets updated. If the total number of nodes in the event-node queue is greater than the minimum support value, then the binary value is marked as true, else as false. While mining the tree, we follow the event-node queues to check for appending to the already obtained m-prefix sequence. If the binary value is not true, then the event node queue is ignored as such, only the event-node queues which have the binary value true are considered. In this way we can actually implement the incremental model.Starting with some element ei from the header table, it tries to find an event type which has required support to append to an already computed m-frequent sequence using the event-node queues from the header table. The next event about to get checked is found by adding the support counts of the first occurrences of an event type in each suffix tree of the current root set. Initially only the root node is in the current root set.

IncrementalPLWAPMine()

Input : PLWAP tree constructed_Tree, Header Linkage Table HTL, minimum support value min_sup, suffix tree root sets Curr_RSet (Curr_RSet contains only the root initially), Frequent sequence from previous step F

**Output** : Frequent sequence generated for this step F*

**Other variables used** : Var_Check_Ancestor stores whether the node is ancestor of following nodes in the queue, Var_Counter stores the total number of events $e_i$in the suffix trees

Begin

1.If Curr_RSet is empty, return

2.For each event $e_i$in HTL, find the suffix tree of $e_i$in constructed_Tree, do

a. Check the binary value $B_s$true or not, if not true, then skip this queue and proceed to the next.

Else, If true, then

i.Following the event type $e_i$-queue,

1.If$e_i$is descendant of some event in Curr_RSet and not descendant of Var_Check_Ancestor,

a .$e_i$gets inserted to new root set generated for next step R*

b. Add the count value of $e_i$to Var_Counter

c. Replace Var_Check_Ancestor with $e_i$

2. If Var_Counter is greater than min_sup,

a. Append ei to the set F to form F*, andouput F*

b.Call Algorithm IncrementalPLWAPMine() passing R* and F*

End // IncrementalPLWAPMine()//

As the depth of the tree increases, the frozen header list becomes an extra load on the mining algorithm. Suppose, think that we are having nodes in our root set which are way down below the root level. So it is very intuitive that its suffix trees would be way down below. But, in the conventional implementation, we start from the beginning of every event-node queue. And most of the nodes would be its ancestors. So it is a drawback that we have to check each and every node in each queue that we froze before starting the mining process. To avoid this, a threshold was set in the number of levels. Once the threshold is reached, the event queues are generated again. The threshold is set relative to dataset size in consideration.

# 3. RESULTS

The web usage mining tooldevelopedincorporates the improved incremental PLWAP technique shows better performance in comparison with the original PLWAP itself and the later PL4UP proposed by Ezeife et al[3]which in itself an incremental mining model.The efficiency of the algorithm[11] is low because, we need to perform the insertions and deletions every time on the PLWAP tree when updations take place. This algorithm gives satisfactory performance only when the percentage updation in size is less than fifty. But even in that range, that is updations of less than 50%, the new algorithm outperforms the PL4UP.

Initially, server log file of 1000 entries was taken for the experiment. The performance was compared with the original PLWAP.The following results were obtained. Table 1 shows the table showing the comparison and Fig 1 shows the corresponding graph.

**Table 1: Comparison with the original PLWAP mining algorithm**

| Percentage increase in the size of Log File | PLWAP Tree execution time(In Milliseconds) | | | | Incremental PLWAP Tree execution time(In Milliseconds) | | |
|---|---|---|---|---|---|---|---|
| | Prepare WASD | Apply minimum support | Build tree and generate frequent patterns | Total Execution Time | Add transactions to the WASD | Add transactions to the PL-WAP tree | Total Execution Time |
| 20 % | 9889 | 889 | 158 | 10936 | 5410 | 245 | 5655 |
| 40 % | 11861 | 903 | 176 | 12940 | 6284 | 188 | 6472 |
| 60 % | 12764 | 1890 | 162 | 14816 | 8202 | 337 | 8539 |
| 80 % | 14270 | 1913 | 151 | 16334 | 9430 | 183 | 9613 |
| 100% | 15245 | 2034 | 167 | 17446 | 10353 | 374 | 10727 |

The table shows, different phases the tool undergoes while processing a log file. The preparation of WASD in PLWAP is the sessionizing the log file entries into transactions taking into account various conditions.

Applying minimum support value eliminates the nodes from the wasd which are not frequent. In the incremental PLWAP, time taken to preprocess the incoming transactions is in the first column and the next shows the time to add them to the existing tree and updating the header table. The graph is as shown below in Fig 1.

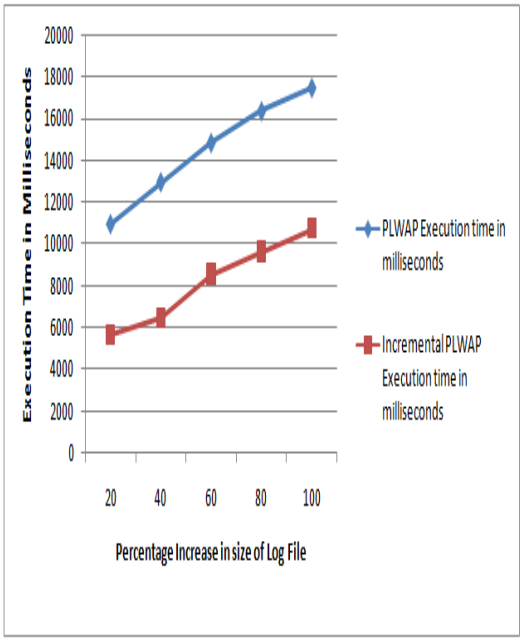


**Fig 1: Comparison when Log File contains 1000- 2000 entries at a minimum support of 40%**

Then a server log file containing 4000 entries was used as input to the tool. Minimum support was tested at 80%. The performances of both the original PLWAP tree and the incremental tree were compared as shown in table 1.2

The incremental PLWAP model outperforms the original WAP tree by a very large scale. The increase in the size of the dataset makes the pre-processing of the log file a lengthy task. Incremental model only need to preprocess the updated incoming transactions. Graph is plotted comparing the performances as shown in Fig 2.

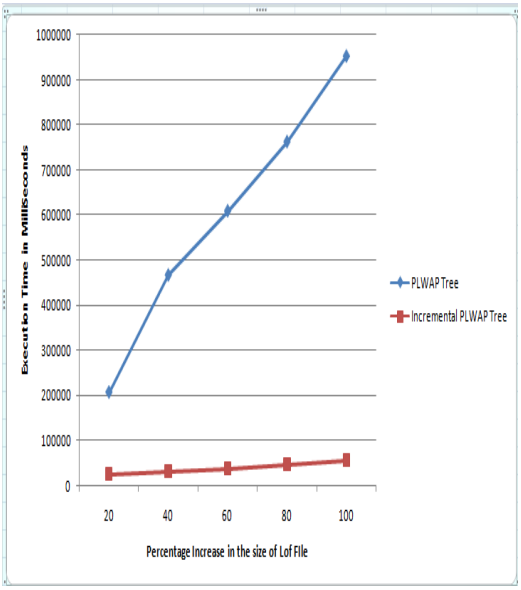


**Fig 2: Comparison when Log File contains 4000-8000 entries at a minimum support of 80%**

**Table 2:Comparison of the performances of both the original PLWAP tree and the incremental tree**

| Percentage increase in the size of Log File | PLWAP Tree execution time (In Milliseconds) | | | | Incremental PLWAP Tree execution time (In Milliseconds) | | |
|---|---|---|---|---|---|---|---|
| | Prepare WASD | Apply minimum support | Build tree and generate frequent patterns | Total Execution Time | Add transactions to the WASD | Add transactions to the PL-WAP tree | Total Execution Time |
| 20 % | 169709 | 36493 | 557 | 206759 | 24427 | 609 | 25036 |
| 40 % | 418806 | 47306 | 206 | 466318 | 29286 | 721 | 30007 |
| 60 % | 524112 | 84282 | 210 | 608604 | 36243 | 847 | 37090 |
| 80 % | 680831 | 80827 | 224 | 761882 | 45829 | 791 | 46620 |
| 100% | 863373 | 87826 | 289 | 951488 | 54787 | 823 | 55610 |

## 3.1 Comparing with PL4UP

The PL4UP[11] which is an incremental mining model of PLWAP was compared with the Incremental model developed. The experimental data of performance of PL4UP was obtained from the paper by Ezeife et al.[3]. The comparison is shown in Table 3.

The table shows, the ratio of the execution times of PLWAP to the incremental model and that to the PL4UP. Figure 3 shows the performance graph of both the algorithms based on the ration against the original PLWAP tree.

**Table 3: The experimental data of performance of PL4UP**

| Percentage increase in the size of Log File | Ratio of execution times of PLWAP to incremental PLWAP with Log file of 4000 entries | Obtained Ratio of execution times of PLWAP to RePL4UP |
|---|---|---|
| 20% | 8.25 | 6.52 |
| 40% | 15.54 | 8.63 |
| 60% | 16.40 | 8.88 |
| 80% | 16.44 | 9.17 |
| 100% | 17.11 | 9.89 |

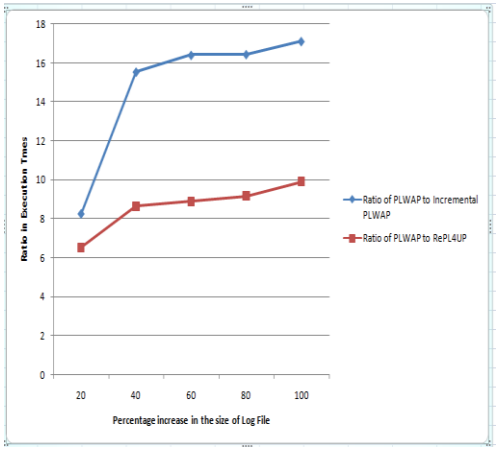


**Fig3:Comparison of PL4UP with the improved incremental model**

The graph in Fig 3 shows that the improved incremental model developed gives better performance than the existing PL4UP. The improved incremental model was validated experimentally by developing the web usage mining tool which implemented the algorithm.The new improved incremental PLWAP tree technique which was implemented in the web usage mining tool that was developed gave better performance than earlier incremental methods proposed.Pre-processing stage takes a big toll on the overall performance of the tree algorithms. When constant updations take place continuously, we need to opt for incremental approaches. The algorithm preprocesses the updated data quickly into web access sequence database. The tree gets the new branches added to it, the counts updated and the header table gets updated. The new frequent nodes are recognized with the help of a binary variable which is part every event node queue. The non-frequent nodes are ignored while mining stage with the help of the same binary variable. This way it achieves better performance which was experimentally validated.

## 4. CONCLUSION

Sequential pattern mining from server web logs can reveal interesting knowledge about the user navigation patterns. Many techniques and methodologies are already in practice for sequential mining. Among them, tree based techniques have so far given the best results. Some tree based techniques were discussed which proved to take large amount of time to extract the patterns.A new PLWAP tree based methodology was proposed which captures the entire content of the tree in a single go. Non frequent nodes are also used to build the tree and while mining they are ignored. No unwanted information is stored in the proposed tree. RePL4UP requires more details for every coming step and so do require more time and space for performance. It gives results in lesser time than the PL4UP and also overcomes the disadvantage of PL4UP that it gives satisfactory performance only when the updation in size of the existing dataset is less than 50%. The results were represented both in tables and graphs and were experimentally validated. The results given by the proposed improved incremental PLWAP technique is highly motivating.As future work, PLWAP algorithm could be practically tested on traditional databases other than weblogs. Another prospective is that web usage mining could be combined with content for effective knowledge representation of web link navigations. Other areas such as distributed mining could also be merged with the proposed technique.

## 5. REFERENCES

[1] Brijendra Singh, Hemant Kumar Singh;"Web Data Mining Research: A Survey", 978-1-4244-5967-4/10/$26.00 ©IEEE, 2010.

[2]Sang T.T. Nguyen;"Efficient Web Usage Mining Process for Sequential Patterns", Proceedings of iiWAS,2009.

[3] Ezeife, C.I., and Lu, Y. "Mining Web Log Sequential Patterns with Position Coded Pre-Order Linked WAP-Tree".Data Mining and Knowledge Discovery. 10, 1, 5-38.DOI=10.1007/s10618-005-0248-3, 2005.

[4] R. Agrawal and R. Srikant, "Mining sequential patterns," In: Proceedings of the 11th Int'l conference on data engineering, pp 3–14, Taipei, 1995.

[5] Masseglia, F., Poncelet, P., and Cicchetti, R.,"An efficient algorithm for web usage mining. Networking and Information Systems Journal (NIS)", 2(5/6):571–603,1999.

[6] Nanopoulos, A. and Manolopoulos, Y., "Mining patterns from graph traversals. Data and Knowledge Engineering", 37(3):243–266, 2001.

[7]Han, J. and Kamber, M., Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, 2000.

[8] Pei, J., Han, J., Mortazavi-Asl, B., and Zhu, H.,"Mining access patterns efficiently from web logs".In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'00). Kyoto, Japan, 2000.

[9]Han, J., Pei, J., Yin, Y., and Mao, R, "Mining frequent patterns without candidate generation: A frequent pattern tree approach", International Journal of Data Mining and Knowledge Discovery. Kluwer Academic Publishers, 8(1): 53–87, 2004.

[10] Borges, J. and Levene, M.,"Data mining of user navigation patterns". In Masand, B. and Spliliopoulou, M., editors, Web Usage Analysis and User Proling, Lecture Notes in Artificial Intelligence (LNAI 1836), pages 92{111. Springer Verlag, Berlin, 2000.

[11] Liu, Y., Huang, X., and An, A. "Personalized Recommendation with Adaptive Mixture of Markov Models".The American Society for Information Science and Technology. 58, 12, 1851–1870. DOI=10.1002/asi.20631, 2007.