Implementation of WAP through an Innovative and Efficient Technique

Shorya Agrawal
Research Scholar, RGPV
Bhopal, India

Nirved K. Pandey
Director, GEC Gwalior
Gwalior, India

Amit Kanskar
HOD, NRI Bhopal
Bhopal, India

Nishant Khare
Asstt. Professor, NRI Bhopal,
Bhopal, India

ABSTRACT
Web Access Pattern (WAP) tree mining is finding of sequence pattern from web access log. It has gained importance in view of increasing usage of World Wide Web. Access to web pages generates access log wherefrom meaningful information is extracted. WAP stores web accesses in a prefix tree. In order to mine data, this tree is recursively traversed in bottom up fashion for frequent items that starts with suffix sequences. Repeated construction of sub-trees for finding frequent itemset is necessary in this method. This paper proposes an improved technique termed as WRDSP (WAP Related Dotted Sequence Path) for creation of such graph in which each item needs to be constructed only once. For each attribute, single node only needs be created in proposed approach whereas many nodes may be required for each attribute in conventional WAP approach. To mine frequent pattern from such graph does not require repeated traversal of links already traversed, which is a big saving in memory and time.

Keywords
Association Rule Mining (ARM), Frequent Item sets (FIS), Web Access Pattern (WAP), Web Access Pattern Relative Dotted Sequence Path (WRDSP) and Web Access Mining (WAM).

1. INTRODUCTION
Web usage mining is a particular category of web mining [1, 2]. This mining provides web access information from web logs [3, 4]. This information is gathered automatically into access logs from the web servers. Scripts such as CGI scripts provide additional information such as user subscription information and survey logs. Usage mining enables web mining concerns to procure productivity information pertaining to future of their business function ability.

Information so gathered has two fold important uses. First use is that this information provides companies trends of relative future requirements for the items, which gives better salability of the products of the companies. The second use is that same information provides web designer a clue about how to present the existing information. On the basis of this information from WAP mining, major objectives such as usage processing, content processing and structure processing may be achieved. Former two are concerned with conversion of web information like text, images scripts and others into useful forms. This helps with the clustering and categorization of web page information based on the titles, contents and image available. The latter information i.e. structure processing consists of analysis of the structure of each page contained in the web site.

In order to include time constraint, sliding time window and user defined taxonomy an algorithm termed as Generalized Sequential Pattern Algorithm was developed [5, 6]. GSP Algorithm (Generalized Sequential Pattern algorithm) [7] is an algorithm used for sequence mining. The algorithms for solving sequence mining problems are mostly based on the Apriori (level-wise) algorithm [8].

The shortcoming of multiple scans was removed to some extent in WAP tree [9]. The construction of WAP tree may be described in nutshell as follows. The WAP-tree stores the web log data in a prefix tree format similar to the frequent pattern tree (FP-tree) [10, 11]. WAP algorithm first scans the web log to compute all frequent individual events, and then it constructs a WAP-tree over the set of frequent individual events of each web log. This WAP tree is recursively mined by building a conditional WAP tree for each conditional suffix frequent pattern found. The process of recursive mining of a conditional suffix WAP tree ends when it has only one branch or conditional tree becomes empty.

Trying to achieve more time efficient mechanism, another approach known as Prefix Span Pattern tree (PSP) evolved [12]. GSP and PSP are similar except prefix tree is used in PSP approach. At each step, in order to count support candidates, the database is browsed and frequent sequence set is built. Candidate sequence may be found by traversing a branch from to leaf.

PLWAP is another attempt in the direction of improvement in WAP tree based mining [13, 14]. The proposed algorithm builds the frequent header node links of the original WAP-tree in a pre-order fashion and uses the position code of each node to identify the ancestor/descendant relationships between nodes of the tree. It then, finds each frequent sequential pattern, through progressive prefix sequence search, starting with its first prefix subsequence event [15]. However, PLWAP-tree algorithm can't denote ancestor-descendent relationship of nodes in PLWAP-tree clearly.

There have been many other attempts upon improving the time complexity in web access log mining [16, 17, 18, 19]. So far no attempt may claim that it has achieved optimum performance. Proposed WRDSP based approach to mine sequence pattern is another such fresh attempt. We claim that for moderate sized web log, this approach achieves near optimum results.

The next sections are organized as follows. The proposed approach has been discussed in section 2. Section 3 deals with performance analysis. It compares WRDSP based mining and WAP. Remaining sections deal with conclusion and references.

2. THE WEB RELATIVE DOTTED SEQUENCE PATH (WRDSP) BASED WAM

In this section, first we describe proposed technique. WRDSP stands for WAP Relative Dotted Sequence Path. The method for calculation of frequent item set named as WRDSP based WAM is demonstrated with the help of an example. Algorithm runs in three major steps. These steps are as follows:

1. Each log record is transformed according to descending frequency order.
2. WRDSP graph is created, where each link is labeled with its WRDSP value.
3. Frequent Item Sets are determined from WRDSP graph generated in step 2.

First step is also employed in WAP tree creation hence is not elaborated. Second step creates WRDSP graph, which contains all nodes that have qualified to belong to the group that have support above threshold level. Creation of WRDSP graph begins with creation of a starting node also known as null node along with all other qualified nodes. After this, nodes are linked with each other and with the starting node as per transformed frequent items in each web log. The label of each link is its WRDSP. Each web log is now considered. Items are linked from null node one by one. First item of the each web log is always connected with null node. WRDSP path is created or updated with each item based on WRDSP prefix path as mentioned earlier. In this way, graph has all necessary links that are labeled as per their WRDSP values. The illustration of first and second major steps is done using following running example.

2.1. Running Example of WRDSP Calculation

	Items Web log	(Ordered) frequent items
100	a,b.d.a.c	a,b,a,c
200	e,a,e,b,c,a,c	a,b,c,a,c
300	b,a,b,f,a,e,c	b,a,b,a,c
400	a,f,b,a,c,f,c	a,b,a,c,c

Table 2-1: Sample database

Assume that the minimum support threshold is 75%. As per the first step, algorithm makes the first pass through the database and finds singleton itemsets (items) with enough support as follows.

a: 4, b: 4, c: 4

Item ‘a’, ‘b’ and ‘c’ have the maximum frequency in the given web log. All other items, whose support was below threshold value, have been dropped. In the second pass, algorithm transforms items in each web log in descending order of determined frequency.

In second step, WRDSP graph is created. Graph has starting node also known as null node along with other nodes i.e. a, b and c. Each web log is considered one by one. Each web log updates an existing WRDSP count or/and determines WRDSP value of a newly created link.

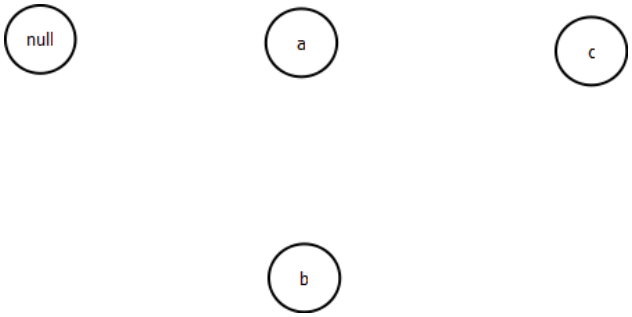


Fig 1: items with null node

In present case for tid = 100, a link is created from null to item a with WRDSP 1(1). First 1 indicates the path name and Second (1) in parenthesis indicates count value of item a. The count denotes no. of occurrences of this item. After this links are similarly created from a to b with WRDSP = 1.1(1), from b to a with WRDSP = 1.1.1(1) and finally from a to c with WRDSP = 1.1.1.1.1(1).

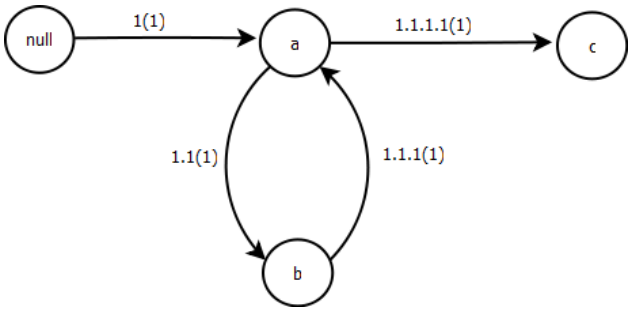


Fig 2: Graph after first web log

Next web log with tid = 200 has transformed frequent items as a,b,c,a,c. Item ‘a’ has already been connected with starting node null. Hence its count is incremented by 1, which updates WRDSP of link as 1(2). Similarly WRDSP of link ab is updated as 1.1(2). Item c is to be connected from item b for the first time. So far item b has one connection only, which is from item a. Prefix till item a so far is 1.1.1. Hence WRDSP of link bc is 1.1.2(1). Similarly WRDSP of link ca is 1.1.2.1(1) and WRDSP of link ac is 1.1.2.1.1(1).

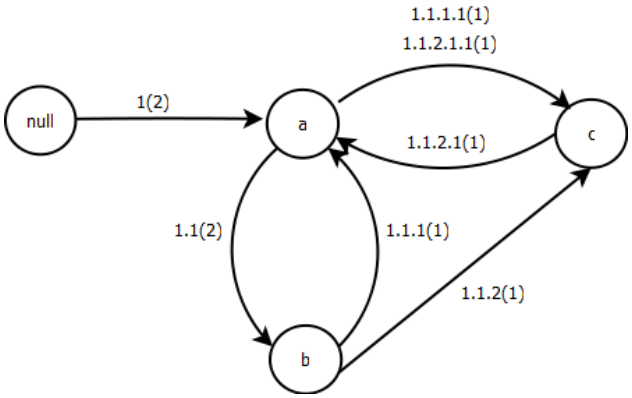


Fig 3: Graph after Second web log

For tid = 300, transformed frequent set consists of item b, a, b, a and c. From null item b is to be connected for the first time. This causes WRDSP of link null to b to be 2(1). It is to be noted here that WRDSP of link null to b had prefix as 2. Though another link a emits from null, it had prefix as 1, which is not same as 2. With prefix 2, null is the first link from item b. Hence the WRDSP label 2(1) is justified. and WRDSP of link ba to be 2.1(1). Link ab get its WRDSP as 2.1.1(1) on the basis of similar reasoning. Similarly WRDSP of link ba is 2.1.1.1(1) and WRDSP of link ac is 2.1.1.1.1(1).

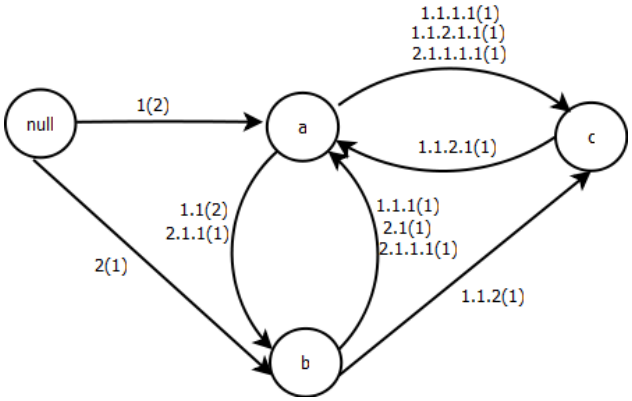


Fig 4: Graph after forth web log

For tid = 400, transformed frequent items are a, b, a, c and c. Item 'a' has already been connected with starting node null. Hence its count is incremented by 1, which updates WRDSP of link as 1(3). Similarly WRDSP of link ab is updated as 1.1(3), WRDSP of link ba is updated as 1.1.1(2) and link ac is updated as 1.1.1.1(2). Item c is to be connected from item c for the first time. So far item c has no connection. Prefix till item c so far is 1.1.1.1 Hence WRDSP of link cc is 1.1.1.1.1(1).

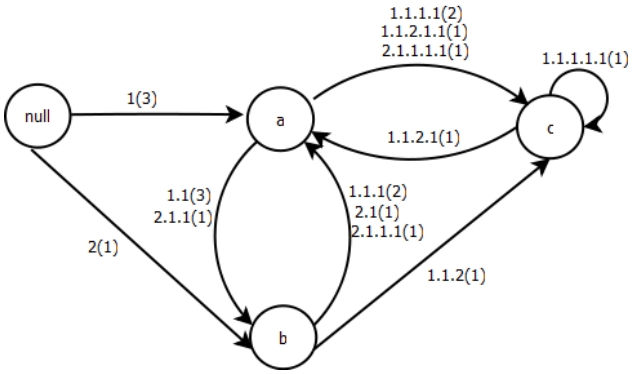


Fig 5: Graph after last web log

Third and final step results in generation of conditional frequent pattern by reading WRDSP-tree. This is achieved in 3 steps. Initially the items are arranged from least frequent items to most frequent items. This information is available from first major step. In this running example, items are c, a and b.

Step 1: Path is traced from c to root (null), which is available using WRDSP graph generated in second major step. As shown in fig. 2-7. There are Four paths from c to the root. First link joins c from c. WRDSP of this link is 1.1.1.1.1(1). The prefix of last dotted value is 1.1.1.1 which is provided by link that joins c to a. Note that this prefix is not same as WRDSP value of link ac, which is 1.1.2.1. Hence the path towards root is c, c, a so far. This process is repeated. At a prefix of the link is 1.1.1, which is provided by link joining a to b. Hence path from c is extended as c, c, a, b. Subsequently the path becomes c, c, a, b, a. Similarly the second path from c to root becomes c, a, c, b, a. For items forming first path i.e. c, c, a, b, a, count values are initialized to the values of the WRDSP of last item. Thus for item c associativity from: c, a, c, b, a is initialized to the WRDSP count of link c to a, which is 1. Similarly associativity of item a from b,a is initialized to 1, associativity from b to a is initialized to 1. The last item a is associated with null. Counts are final for each of the association and need not be updated. This is a significant improvement over WAM based WAP tree. In WAP tree, while finding associativity, calculation at each link may be repeated several times. This happens as items themselves are repeated because one tree is formed for each path. In WRDSP based approach, each link traced so far will never be required to be considered for next associativity calculation. So far the associativity is as follows.

c → caba(1)

a → ba(1)

b → a(1)

This is shown with darkened links.

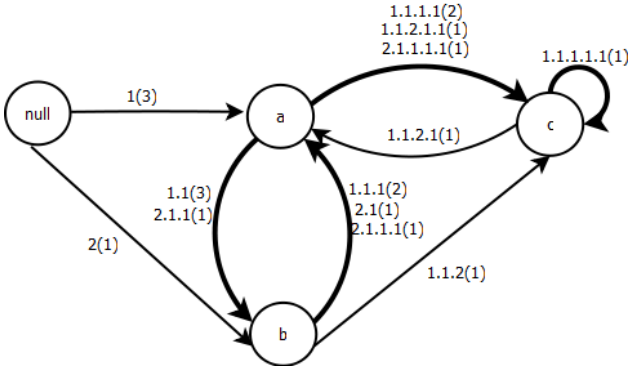


Fig 6: Link associativity item c to a

Taking up, second link from c to root i.e. c, a, b, a the associativity of each item is similarly updated to obtain given set.

c → aba(1), caba(1)

a → ba(2)

b→ a(2)

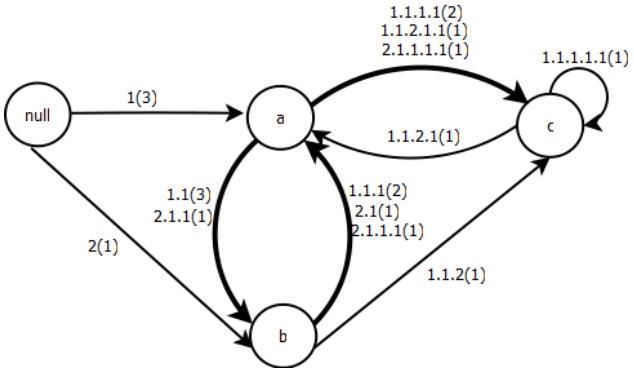


Fig 7: Link associativity item c to a

Third Link ca results in new associativity relation cacba. The updated associativity status becomes:

c → acba(1), aba(1), caba(1)

a → cba(1), ba(2)

b → a(3)

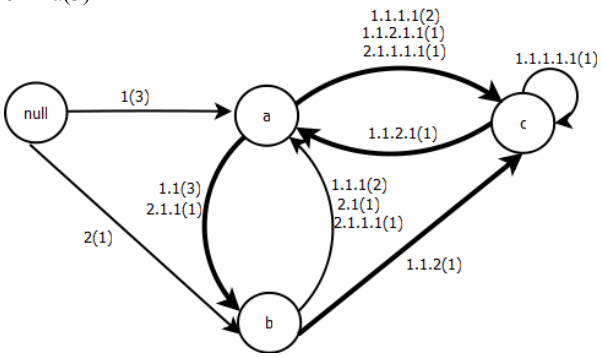


Fig 8: Link associativity item c to a

Process is repeated at link cabab, making the final status as follows.

c → abab(1), acba(1), aba(1), caba(1)

a → bab(1), cba(1), ba(2)

b → ab(1), a(3)

This is shown with darkened links.

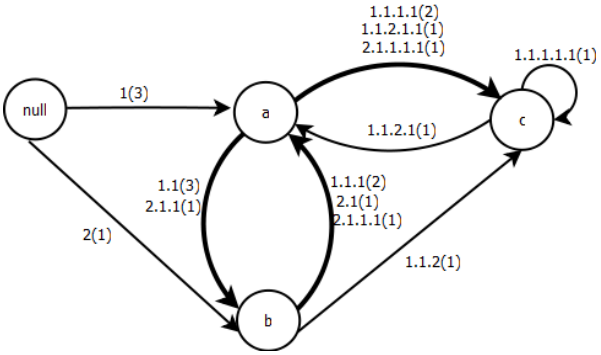


Fig 9: Link associativity item c to b

Step 2: Frequency of each item is counted with each least frequent item. The frequencies of these items are added. In the case of running example, items a, b and c are associated with item c. The frequencies of association of these items are 4, 4 and 2 respectively. As threshold value is 3 or more, only item c is retained in final association status. This process is repeated for all items.. Final associativity status is as follows. Common values are found and their frequencies are calculated. Results are displayed below.

c → a(4), b(4), ab(4), ba(4), aba(4)

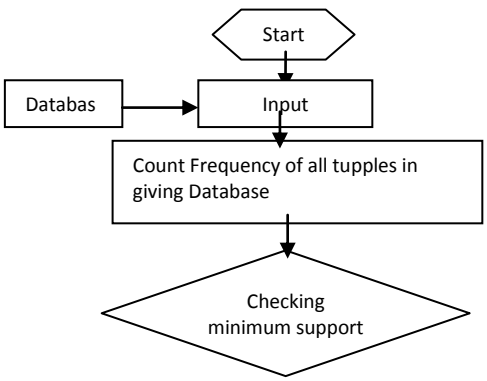
a → a(4), b(4), ba(4)

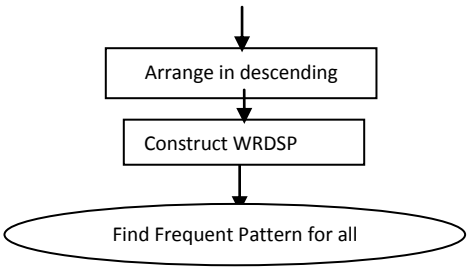
b → a(4)

Step 3: For each association, all combinations of the item that includes the item being considered itself are generated. In the running example, for association b → a(4), single frequent item sets (FIS): ab(4) is generated. For association c → a(4), b(4), ab(4), ba(4), aba(4), FIS: ac(4), bc(4), abc(4), bac(4) and abac(4) are generated. Similarly for item a, FIS: aa(4), ba(4) and baa(4) are generated. This achieves the final goal.

2.2. WRDSP Flow chart

The creation of the proposed graph may be described as follows:





3. THE PERFORMANCE ANALYSIS

In this section, WRDSP based WAM is compared with existing classical WAP tree technique. WAP tree algorithm has been discussed in section 1. WRDSP based WAM was described in previous section.

3.1. WAM in comparison with WAP

WRDSP based WAM has been compared with well-known algorithms for mining frequent closed itemsets i.e. WAP tree [5]. Experiments were performed on a 266-MHz Pentium PC machine with 128 megabytes main memory. Absolute number of runtime are not directly compared with those in some published reports running on the RISC workstations because different machine architectures may differ greatly on the absolute runtime for the same algorithms. Run time implies the total execution time, that is, the period between input and output, instead of CPU time measured in the experiments in some literature. Run time is a more comprehensive measure since it takes the total running time consumed as the measure of cost, whereas CPU time considers only the cost of the CPU resource. Also, all reports on the runtime of WAP tree include the time of constructing WAP trees from the original databases.

The test comparison is performed by C10.S5.N2000.D60K. This data set is generated by IBM synthetic data generator, which is a standard web database used in WAP tree and many other technique.

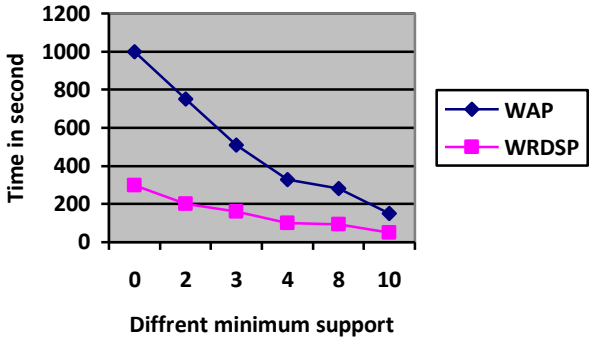
The scalability of WRDSP with WAP tree as the support threshold decreases from 10% to 0.1% is shown in Figure 3-1 and table 3-1.

Algorithm/ Support	**0**	**2**	**3**	**4**	**8**	**10**
WAP	1000	750	510	330	280	150
WRDSP	300	200	160	100	90	50

Table 3-1: Execution times for dataset at different minimum supports.

WRDSP scales much better than WAP tree because as the support threshold goes down, the number as well as the length of frequent itemset increase exponentially. On the basis of the results it could be affirmed that WRDSP algorithm performs much faster than WAP tree construction and other approach.

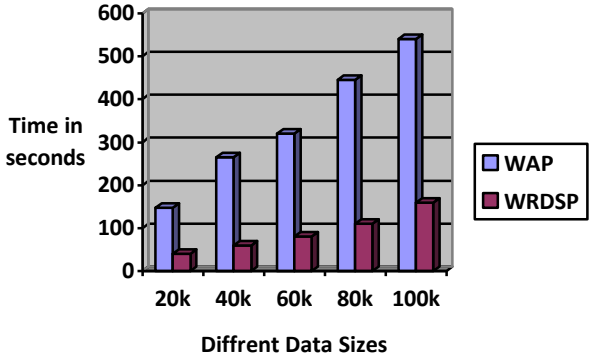
Fig 10: Scalability with threshold



WAP tree mining incurs higher storage cost (memory or I/O). Even in memory only systems, the cost of storing intermediated trees adds appreciably to the overall execution time of the program. It is however, more realistic to assume that such techniques are run in regular systems available in many environments, which are not memory only, but could be multiple processor systems sharing memories and CPU's with virtual memory support.

Databases with different sizes from 20 K to 100 K with the fixed minimum support of 7% are used, which gives following results.

Fig 11: Execution times trend with different data sizes.



Algorithm in time/ Web log size	**20K**	**40K**	**60K**	**80K**	**100K**
WAP	148	265	320	445	540
WRDSP	40	58	82	110	162

Table 3-2: Execution times trend with different data sizes.

4. CONCLUSION & FUTURE SCOPE

In this paper, a technique for determining association rules using WRDSP graph has been proposed. The merit of WRDSP approach lies in the formation of such a graph in which each item of the web log has single occurrence. This limits the number of links required. In this way, all web access pattern may be generated in an optimal way. The resulting performance analysis justifies the projection. However as the size of the database becomes larger from moderate level, performance does not scale in the same proportion. As a future task, WRDSP based approach is to be adjusted for large databases. Another issue to be dealt in future is tailoring WRDSP based approach to tackle top k association rather than all associations above min support.

5. REFERENCES

[1] Cooley, R., Mobasher, B., and Srivastava J., "Data preparation for mining World Wide Web browsing patterns", In Journal of Knowledge & Information Systems, Vol.1, No.1, 1999.

[2] Spiliopoulou, M. and Faulstich, L., "WUM: A tool for Web utilization analysis", In Proc. 6th Int'l Conf. on Extending Database Technology (EDBT'98), Valencia, Spain, March 1998.

[3] Borges, J. and Levene, M., "Data mining of user navigation patterns", In Proceedings of the KDD Workshop on Web Mining San Diego California, pages 31–36, 1999.

[4] Etzioni O., "The world wide web: Quagmire or gold mine", Communications of the ACM, 39(1):65 – 68, 1996.

[5] Agrawal, R. and Srikant, R., "Fast algorithms for mining association rules", In Proc. 1994 Int. Conf. Very Large Data Bases, pages 487{499, Santiago, Chile, September 1994.

[6] Agrawal, R. and Srikant, R., "Mining sequential patterns", In Proc. 1995 Int. Conf. Data Engineering, pages 3{14, Taipei, Taiwan, March 1995.

[7] Srikant, R. and Agrawal, R., "Mining quantitative association rules in large relational tables", In Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data, pages 1-12, Montreal, Canada, June 1996.

[8] Agrawal, R., Imielienski, T., and Swami, A., "Mining Association Rules between Sets of Items in Large Databases", Proc. Conf. on Management of Data, 207–216,ACM Press, New York, NY, USA 1993.

[9] 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.

[10] 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, 8(1):53–87, Jan 2004.

[11] Han, J., Pei, H., and Yin, Y, "Mining Frequent Patterns without Candidate Generation", In: Proc. Conf. on the Management of Data (SIGMOD'00, Dallas, TX).ACM Press, New York, NY, USA 2000.

[12] 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.

[13] Ezeife, C. and Lu, Y., "Mining web log sequential patterns with position coded pre-order linked wap-tree", International Journal of Data Mining and Knowledge Discovery (DMKD) Kluwer Publishers, 10(1):5–38, 2005.

[14] Lu, Y. and Ezeife C., "Position coded pre-order linked wap-tree for web log sequential pattern mining", In Proceedings of the 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2003), pages 337–349. Springer, May 2003.

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

[16] Ezeife, C.I., Lu, Yi and Liu, Yi, "PLWAP Sequential Mining: Open Source Code", Proceedings of the First International Workshop on Open Source Data Mining, pages 26-35, 2005.

[17] Ezeife, C.I. and Chen, Min, "Incremental mining of Web sequential pattern using PLWAP tree on tolerance Min Support", Database Engineering and Applications Symposium, 2004. IDEAS '04. Proceedings. International,2004 , Page(s): 465 – 469.

[18] Liu, Lizhi and Liu, Jun, "Mining Maximal Sequential Patterns with Layer Coded Breadth-First Linked WAP-Tree", 2009 Second Asia-Pacific Conference on Computational Intelligence and Industrial Applications, pages- 61-65.

[19] Parmar, Jatin D and Garg, Sanjay, "Modified Web Access Pattern (mWAP) Approach for Sequential Pattern Mining", IJCSNS International Journal of Computer Science and Network Security, VOL. 9 No.6, June 2009.