A Novel Weighted Support Method for Access Pattern Mining

Gopalakrishna Kurup Raju[1] and Achuthan Nair Rajimol[2]
[1]Department of IT, Kannur University, Kannur, Kerala, India
[2]Marian College, Kuttikkanam, Idukki, Kerala, India

Abstract: *Sequential Pattern Mining is an important data mining technique that finds out all frequent sequential patterns in a sequence database. Applications in wide range of important domains make Sequential Pattern Mining an interesting area of research. Conventional approach for sequential pattern mining treats each and every item in the sequence with equal importance and thus fails to reflect the individual significance of items. Weighted Sequential Pattern Mining is an approach that treats different items in the sequences with different weights so as to reflect the importance of each item. Thus, weighted method models real life sequence database in a better manner and more efficient than the conventional sequential pattern mining. Weighted sequential pattern mining can be used to mine web access patterns more efficiently from web log data. This paper proposes a new weighted access pattern mining algorithm to mine weighted access patterns in a web log database. The proposed method uses frequency of user visit to give weights to web pages during the mining process. Through extensive experimental evaluation the algorithm is proved to be promising.*

Keywords: *Access Pattern Mining, Sequential Pattern Mining, Web Access Pattern, Weighted Pattern Mining, Weblog Mining.*

Received January 6, 2013; Accepted July 5, 2013

1. Introduction

Sequential pattern mining, introduced by Agrawal and Srikant in [1] is an important data mining tool used for finding all sequential patterns that satisfy a given support threshold. Sequential pattern mining finds application in scientific and business domains, such as stocks and market basket analysis, natural disasters (e.g. earthquakes), DNA sequence analysis, gene structure analysis, web log click stream analysis etc.

It can also be effectively used to capture frequent navigational paths among user trails which play an important role in web recommendation, web caching, web personalization and so on. Web Access Pattern is a sequential pattern in a large set of pieces of web logs. A web log contains all browsing details of users interacting with the web. For the purpose of study of sequential pattern mining, pre-processing is applied to the original log file and Web Access Sequence Database (WASD) is generated, which is a sequence of pairs: user-id and access information [6, 12].

There are mainly two heuristics in sequential pattern mining - Apriori based and Pattern Growth based methods. Apriori methods that use generate and test approach, face the problem of tedious support counting and generation of explosive number of candidates when mining large sequence databases. Pattern Growth methods grow frequent patterns by mining increasingly smaller projection databases, and thus, are faster than apriori-based algorithms [8, 13].

Main limitation of the traditional approach for mining frequent patterns and sequential patterns is that all items are treated uniformly. But, in real life examples, items have different importance. Items with low support become more important due to some features of the item itself.

For this reason, weighted pattern mining algorithms have been suggested and it gives different weights to items according to their significance. Recently several proposals incorporating weight constraints into both Frequent Pattern Mining [2, 18, 19, 20] and Sequential Pattern Mining [3, 4, 5] have been introduced.

The main focus of weighted frequent mining concerns the downward closure property. Downward closure property, which is the main idea used in the pruning step, says that a non-frequent sequence can never lead to a frequent pattern. But, the downward closure property is usually broken when different weights are applied to different items. In weighted sequential pattern mining, the anti-monotone property cannot be directly used. Even if a sequential pattern is weighted infrequent, its super patterns may be weighted sequential frequent because a sequential pattern which has a low weight can get a high weight after another item with a higher weight is added. Through the use of a Global Weight, the anti-monotone property can be maintained [5, 10, 18].

Many of the previous work in this area [5, 7, 17] [18, 19, 20] assumed predefined weights for each item. But predefined weight is not much meaningful in web log analysis, as the importance of pages depends upon

the user access itself. In this paper a new Weighted Access Pattern Mining method, FWAP, is proposed that sets weights to each item based on the access sequence itself. Moreover, the earlier works involve two database scans in the generation of weighted sequential pattern. First scan is used to find out the frequent item set and the second one for generating the set of access patterns using only the frequent ones. The proposed method involves only one database scan. In our method non frequent items are not deleted from the WASD. Instead, they are skipped while the sequences are being processed. This enables the algorithm to easily get adapted to incremental mining.

The rest of the paper is organized as follows. Section 2 provides a study of related works, section 3 describes related theory and the new algorithm and section 4 presents the experimental analysis of the algorithm. Conclusion is given in section 5.

2. Related Works

In [2] C. F. Ahmed and S. K. Tanbeer presented a tree structure IWFPTWA (Incremental Weighted Frequent Pattern Tree based on Weight Ascending order) and an algorithm IWFPWA for incremental and interactive WFP mining. Performance analysis by authors showed that the proposed tree structure and the mining algorithm are efficient for incremental and interactive weighted frequent Pattern mining.

In this work the authors claim that their method requires only one database scan. But this is due to the fact that their method assumes that the list of possible items and their weights are already given. In frequent pattern growth method having two database scan, the first scan is for finding out the elements and their support. Also the method sorts the elements in a transaction in the order of weights. But this is not possible in the case of web access sequences as the order of page access is very important.

U. Yun proposed a weighted sequential pattern mining framework and developed WSpan algorithm based on the prefix projected sequential pattern growth approach in [19]. The work defines two pruning methods to detect more appropriate weighted sequential patterns. A weight range is given to the items according to the priority or importance. The performance analysis shows that WSpan is efficient and scalable.

Major challenge when making improvement in traditional Association Rule Mining by introducing weight is the invalidation of downward closure property. In [17], F. Tao et al. proposed a set of new concepts to adapt weighting in the new setting. Among them the major proposal is of using "weighted downward closure property" as a replacement of the original "downward closure property". A new algorithm called WARM (Weighted Association Rule Mining) is developed based on the improved model. The algorithm is shown to be both scalable and efficient.

A Srivastava et al. proposed a method for web caching using weighted association rule mining in [15]. Their work shows that Association Rule Mining (WAR) technique can be used to capture both users' habit and interest. Recent accesses are given more weights as they provide information regarding the current taste of a user. They used a weighted apriori method to generate the frequent URL set and suggested how the result can be used to speed up web access.

An efficient method for modelling user navigation history was proposed by C Makris et al. in [10]. The proposed system groups navigation sessions into clusters. Then each of the clusters is represented by a weighted sequence and using these sequences a generalized weighted suffix tree is constructed. This structure is used as a web page recommendation tool. Proposed estimation of user's navigational intention can be used either in an on-line recommendation system or in a web-page cache system. The method demands a constant amount of computational effort per one user's action and consumes a relatively small amount of extra memory.

R. Forsat et al. developed a novel web recommendation algorithm in [7]. The authors extend the traditional association rule problem by associating a weight with each item in a transaction to reflect the interest of each item within the transaction. Recommendation algorithm is based on the proposed weighted association rule. The weighted association rules of each URL will be extracted from the web log data and similarity between active user sessions will be calculated upon the weighted rules. Recommendation engine will then find the most similar rules to the active user session with the highest weighted confidence.

3. Proposed Work

3.1. Problem Definition and Background Theory

3.1.1. Sequential Pattern Mining

There are mainly two heuristics in sequential pattern mining based on (i) the way in which candidate sequences are generated and stored and (ii) the way in which support counting is done - apriori based and pattern growth based methods. Apriori based methods use generate and test procedure to generate candidate sequences and then to test their support. So they face the problem of generation of explosive number of candidates and tedious support counting when mining large sequence databases having numerous and/or long patterns. Pattern Growth methods grow frequent patterns by mining increasingly smaller projection databases. Data structures used for the representation of database and search space partitioning play an

important role in the efficiency of pattern growth methods.

Let *E* be a set of access events, which represents web pages accessed by users. A web access sequence is an ordered sequence that gives the order in which a particular user access web pages in a session.
A web access sequence *S* may be defined as:

$$S = e_1 e_2 \dots e_n \mid e_i \in E \wedge 1 \leq i \leq n \qquad (1)$$

where, e_i and e_j are not necessarily different for $i \neq j$. That is, repetition of items is allowed. Length of access sequence $S = e_1 e_2 \dots e_n$ is defined as:

$$len(S) = |S| = n \qquad (2)$$

An access sequence *S* with length *n* is called an *n*-sequence [12]. The empty sequence ε is a special web access sequence of length 0 and $\varepsilon.S = S.\varepsilon = S$ for any sequence S where '.' is the concatenation operator [16]. A Web Access Sequence Database *WASD* is a multi-set of web access sequences including the possible empty sequence. That is,
WASD = $\{S_1, S_2, \dots, S_m\}$ where S_i is a web access sequence.

A web access sequence $S'=s_1's_2'\dots s_n'$ is a subsequence of *sequence* $S = s_1 s_2 \dots s_m$, if and only if $n \leq m$ and there exist $i_1, i_2, \dots, i_n$ such that $1 < i_1 < i_2 \dots < i_n \leq m$ and $s'_j = s_{ij}$ for all $1 \leq j \leq n$. The empty sequence ε is a subsequence of any sequence [16].

A web access sequences *S* in WASD is said to support pattern *p* if *p* is a subsequence of *S*. The support of pattern *p* in *WASD*, denoted as $Sup_{WASD}(p)$, is the number of web access sequences in WASD that support pattern *p*.

That is, $Sup_{WASD}(p)$ is defined as,

$$\mathrm{Sup}(p) = \frac{|\{S_i \mid p \subseteq S_i, S_i \in \mathrm{WASD}\}|}{|\mathrm{WASD}|} \qquad (3)$$

Given a support threshold ξ in interval [0:1], a pattern *p* is frequent with respect to ξ and a web access sequence database *D*, if $Sup_D(p) \geq \xi \times |D|$, where $|D|$ is the number of web sequences in *D*, $\xi \times |D|$ is called the absolute support threshold and denoted as η. The web access pattern mining problem is to find all frequent web access patterns in *D* with respect to ξ [12].

Given the database *D* and a symbol *a* in *E*, the *a*-projection database of *D*, denoted as *Da*, is the multi-set of *a*-projections of the web access sequences in *D* that support *a*.

$$Da = \{a - projections\ of\ S \mid a \subseteq S \wedge S \in D\} \qquad (4)$$

The *a*-prefix of a sequence *S* is the prefix of *S* from the first symbol (the leftmost symbol) to the first occurrence of *a* inclusive. The *a*-projection of *S* is what is left after the *a*-prefix is deleted. If *a* occurs only once as the last symbol in *S*, the *a*-prefix is *S* and the projection is the empty sequence ε [16].

For example, if *D* = {*fcbaca; bcbaca; ccbabag*}, the set of *a*-prefixes is {*fcba; bcba; ccba*} and the *a*-projection database *Da* is *{ca; ca; bag}*

Given the database *D* and a symbol *a* in *E*, the same sequence may repeat in *a*-projection database. The support of *a* in *D* is same as the number of sequences in *a*-projection database. *For a non empty database D, the set of frequent pattern is the union of all sequences that are prefixed by* ***a*** *for each a* $\in$ *E having support* $\geq \eta$ [16].

3.1.2. Weighted Sequential Pattern Mining

In conventional sequential pattern mining methods, all sequences in the database are treated with same importance. But, in real examples sequences differ in their significances. For this reason, weighted sequential pattern mining was introduced [2, 4, 5]. Here, items within a sequence are given different weights according to their importance in the sequence database.

Item weight *w(i)* is defined as a value attached to an item to represent its significance. Let $x=(x_1, x_2, \dots, x_n)$ be a set of distinct items and W be a set of non-negative real numbers. A pair (x,w) is called a weighted item where $x \in I$ is an item and $w \in W$ is the weight associated with *x*. A transaction is a set of weighted items, each of which may appear in multiple transactions with different weights.

Weight of an itemset $I=\{x_1, x_2 \dots, x_k\}$ is derived from the weights of its enclosing items [17]. One simple way is to calculate the average value of the item weights. If k is the length of sequence,

$$W(I) = \frac{\sum_{i=1}^{k} W(Xi)}{k} \qquad (5)$$

Thus, the weight of the sequential pattern is the average value of the weights of items in a sequence. The weighted support, *wsup*, of a weighted sequential pattern is defined as the resultant value of multiplying the pattern's support with the weight of the pattern.
If *p* is weighted pattern, the weighted support of *p*, *wsup(p)* is defined as

$$wsup(p) = sup\,(p) \times W(p) \qquad (6)$$

A sequential pattern is called a weighted frequent sequential pattern if the weighted support of the sequential pattern is not less than a minimum threshold.

The main concern in weighted mining is the breaking down of the anti-monotone property when simply applying weights. Anti-monotone property is the crucial property used for pruning the infrequent patterns in pattern mining [12]. That is, even if a sequential pattern is weighted infrequent, its super pattern may be weighted frequent because super patterns of the sequential pattern with a low weight can get a high weight after adding other items or item sets with higher weight [5, 10, 15].

3.2. The FWAP Method

In conventional Web Access Pattern Mining, order of page access is taken into consideration. But in practice the frequency of visit to a web page, the time spent on each pages, current topic of interest etc are also important in pulling out the access patterns. Depending on the importance, weights can be attached to each web page. Weight attached to web pages allows generation of more appropriate access pattern. This helps in reducing the volume of access patterns generated and in turn reduces the space requirement.

Given an access sequence database *WASD* and a support threshold, the problem of weighted access pattern mining is to find the complete set of all weighted access patterns whose weighted supports are not less than the support threshold.

In Weighted Access Pattern Mining methods, there are two main components (i) weight assignment and (ii) pattern mining. Weight assignment module deals with assigning weights to items and item sets. Pattern mining module is for mining weighted access patterns from the WASD by applying the concept of weight and weighted support.

3.2.1. Weight Assignment

Frequency of visit to a particular page of a user is very important information as it shows the user's interest on that page. In the proposed Frequency Weighted Access Pattern (FWAP) mining, frequency of user visit is used to give weights to each item in an access sequence. The mining algorithm considers the weight of a sequence and uses it for finding out the support.

Weight of an access event a_i in an access sequence S is defined as,

$$weight_S(a_i) = \frac{|\{a_i \mid a_i \subseteq S \wedge S \in WASD\}|}{|S|} \quad (7)$$

The weight of an access sequence S in a Web Access Sequence Database is the average value of the weights of the sequence. Given an access sequence $A = \{a_1, a_2, \ldots..a_m\}$, weight of the access sequence A is formally defined as follows.

$$\text{weight}(A) = \frac{\sum_{i=1}^{m} W(a_i)}{|S|} \quad (8)$$

where $a_i \in S$ and $S \in WASD$.

The weighted support of an access sequence is defined as the resultant value of multiplying the sequence'sweight with global support of the sequence.

$$wsup(S) = weight(s) \times GlSup(S) \quad (9)$$

Global support of a sequence is the support of the sequence in the whole database.

$$GlSup(S_i) = \frac{|\{S_i \mid S_i \subseteq S \wedge S \in WASD\}|}{|WASD|} \quad (10)$$

A weighted access pattern is an access sequence that satisfies the predefined weighted support.

3.2.2. Mining of Weighted Access Patterns

The proposed method uses pattern growth techniques which is proved be to better than apriori methods. Patterns are generated by suffix building using projection databases. Downward closure property is maintained in the method by bounding the support by global weight of each item.

WAP-Tree is a tree structure introduced in [12] for holding access sequences in a very compact form to enable access pattern mining. FOL-Mine is an efficient sequential pattern mining algorithm proposed in [14]. It is based on the concept of WAP-tree but proved to be more efficient than all previous WAP-tree based mining algorithms. FOL-Mine uses a special linked structure to hold access sequences for processing and proved to be efficient [14].

The proposed FWAP method uses a modified form of the structure used in [14] to hold the access sequences. The nodes of the structure are modified to hold the generated weight information of each item. The proposed method needs only one database scan to load access sequences into the structure and to generate associated weight information.

FOL-list is the basic data structure used in [14] to hold the first occurrence information of items during the mining of patterns in the intermediate projected databases. FOL- list manages the suffix building very efficiently. The node structure suggested in [14] is modified to process the weighted support of sequences.

The proposed method comprises of three algorithms. The first algorithm *main*(), given in Figure1, is the main algorithm of the method. It reads in Web Access Sequences (WAS) from WASD, assign weight to each element and updates information regarding items and their frequencies. The second algorithm FWAP is the recursive algorithm used for generating the weighted frequent patterns of WASD. Detailed steps are provided in Figure 2. The third algorithm GEN-FO, in Figure 3, is used for generating the list of first occurrences, LFO. Algorithm FWAP makes use of this procedure to effectively manage the projected databases.

3.2.3. Data Structures Used

Data structures play an important role in improving the efficiency of pattern growth methods. Main data structures used in the proposed method and their purpose are described below.

Item List (IL): Linked list containing the items and their frequencies present in the *WASD*.

Current Item List (CIL): Linked list used for storing the items and their frequencies present in the current access sequence under consideration.

List of First Occurrence (*LFO*): Linked list used for storing the first occurrences of a given item in the database. Each node contains the sequence-id (seq-id), and position of the occurrence (pos). If *a* is the given item and *D* is the input database, then *LFO* represents the *a*-projections of *a*, *Da*. Thus, *LFO* is used to manage the projection database very efficiently. During the generation of *LFO* infrequent elements are automatically skipped. So this arrangement of access sequence database does not require the deletion of infrequent elements unlike other earlier access pattern mining algorithms [11, 12, 16, 21, 22]. So, the database structure is suitable for incremental mining too.

WASlist: Each web access sequence is stored in a linked list with each node is of the form: *struct node{int item; int weight; next *node}.*

Headllist[]: The start address of each linked list containing item and its weight is registered in *Headlist[m*], where m = $|WASD|$.

3.2.4. Algorithm Main

The *main* algorithm reads access sequences one by one from *WASD* and stores it in the linked structure with weight of each item set to 0. During this scanning itself Current Item List, *CIL*, is loaded with items present in the current sequence and their frequencies. After finishing the scanning of one full access sequence, the list of all elements *IL* is updated using *CIL* to reflect the presence of elements and their frequencies in the current access sequence. Once this is over weight of each element in *CIL* is calculated by dividing the frequency by the length of the sequence and weight in the current *WASlist* is updated.

Once the processing of access sequences is over weighted frequent item set is generated using *IL*. If the frequency of an item is greater than the absolute support, it is considered as a frequent event. Then the recursive mining algorithm FWAP is called for generating the complete set of weighted access patterns. Algorithm of *main* is given in Figure 1.

3.2.5. Algorithm FWAP

FWAP is the recursive mining algorithm to generate weighted access pattern. It works as follows:- Consider the first element in the weighted frequent list. Use algorithm *Gen-FO* to locate the first occurrences in the current projected database. The weighted support returned along with the list of first occurrence is divided by the size of database, $|WASD|$, to maintain the downward closure property. If the weighted support *wsup* satisfies the support threshold, the weighted frequent element is stacked and FWAP is recursively called for further suffix building until support fails. Now the access pattern is generated and FWAP is again called with next weighted frequent element. This process is continued till no more patterns are to be generated. Algorithm of FWAP is provided in Figure 2.

Algorithm main
Input:
An access sequence database, *WASD*
A support threshold
Output:
Set of weighted access patterns
Method:
1. while eof (*WASD*)
Read in an access sequence $S= a_1a_2...a_n$, assigning
weight(a_i)=0
length =0
For each element a_i in *S* do
Increment *length*
If a_i is not in C*IL*
make an entry in *CIL* for a_i with *count*(a_i)= 1
Else
Increment the count of a_i
{end if}
{*end for*}
Update the list of items *IL* with the *CIL*
For each a_i, update
weight($a_{i)}$=*count*(a_i) / *length*
{*end while*}
2. For each item a_i in IL
if WSup(a_i)>support threshold
move a_i to Σ, the set of frequent item set
3. Call FWAP
4. Return

Figure 1. Algorithm *main.*

Algorithm: FWAP
// E – The set of Patterns;
S – Stack of intermediate Frequent Patterns used for suffix building
//
Parameters:
Current frequent pattern, *p*
List of fist occurrence, *L*
Absolute support, *η*
Method:
1. For each weighted frequent item, a_i
i. Call *Gen-FO* to generate the first occurrences list, L_1,
ii. If the *WSup*(a_i) > *η*
add *p*.a_i to *E*, set of pattern
Add *p*.a_i to stack for suffix building.
p= *p*.a_i
Call FWAP(*p,L1, η*)
{endif}
iii Delete the current *L*
{end for}
2. return

Figure 2. Algorithm FWAP

3.2.6. Algorithm GEN-FO

The algorithm GEN-FO works as follows: - In the initial call, input First occurrence List *LFO* is empty. So the algorithm uses the linked database itself to locate the first occurrence of the given item. In all the subsequent calls *GEN-FO* uses the input *LFO*, the one which is generated in the previous recursive call, to locate the first occurrences. Thus, LFO indicates the

possible extensions in the projected database. This ensures the efficient suffix building. Weight of the element at each occurrence is added up. At the end, header of the first occurrence list is updated with the sum. New occurrence list L1 is returned to FWAP for further processing. Algorithm of GEN-FO is given in Figure 3.

Algorithm GEN-FO.
Parameters:
i. The list of first occurrence L (null in the initial call otherwise the list L generated in the previous call)
ii. current weighted frequent element, *a*
Method:
1. Wsup=0; Initialize L_1
2. If L is empty
Locate first occurrences of *a* from the linked database.
Generate L_1 with each node holding *seq-id* and *pos*
Else
Locate the first occurrences of the element *a* in *Da* using L
Generate L_1 with each node holding *seq-id* and *pos*
{end if}
3. Add the weight of the item at each occurrence
4. Update the header of the list L_1 with total weight× GL-Support
5. return L_1

Figure 3. Algorithm GEN-FO.

4. Experimental Evaluation and Analysis

In this section we present a set of experiments that are performed for evaluating the performance of the proposed method. During the performance evaluation, focus was given to three different aspects,

- Comparison of weighted and non weighted approach, both in terms of execution time and the number of patterns generated,
- Comparison of number of patterns generated and execution time at various support threshold
- And Scalability of the method.

In all the three aspects, the proposed method showed better performance. A detailed discussion of the results of the experiments with charts is provided below. Due to the space limitation only one output table is given.

For conducting the experiments three different datasets are used. Synthetic datasets T25I10D10K and T10I4D100K are generated using the synthetic data generation program of the IBM Quest data mining project at *http://www.almaden.ibm.com/cs/quest/*, which has been used in most sequential pattern mining studies [9, 11, 12, 14, 16, 21, 22]. T25I10D10K is a 948 KB database with 10000 sequences and T10I4D100K is of 3.83 MB with 1 lakh sequences. msnbc dataset is available in *UCI Machine Learning Repository* containing 1000000 sequences. The data comes from Internet Information Server (IIS) logs for msnbc.com and news related portions of msn.com for an entire day. Each sequence in the dataset corresponds to page views of a user during that twenty-four hour period.

All experiments were performed on Intel Dual Core machine with 2GB RAM and running Microsoft Windows XP Professional version 2002. FWAP mining algorithm is implemented in Microsoft visual C++ 6.0.

4.1. Comparison of Weighted and Non-Weighted Approach

Effectiveness of the Weighted Access Pattern Mining Method (FWAP) over the non-weighted approach is evaluated through both (i) comparing processing time and (ii) comparing the number of patterns generated. The proposed method is compared with existing non weighted method to verify the advantage of the proposed method both in terms of memory and speed. For this purpose, non-weighted access pattern mining method FOL-Mine [14] is selected. FOL-Mine is a pattern growth method based access pattern mining algorithm. The authors of [14] proved FOL-Mine to outperform all earlier WAP-Tree based methods [9] [11, 12, 16, 22].

4.1.1. Comparison of Processing Time

Experiments are done on different datasets to compare the execution time at different support values. Special stubs were inserted in the program to calculate the CPU time requirement for the execution of program. Figure.7 shows the comparison of execution time of both methods graphically. The output of the experiment is provided in tabular form in Table.1. The result shows that the proposed FWAP method takes lesser execution time. Moreover as the support threshold decreases the execution time of the non weighted approach, FOL-Mine, increases at a higher rate. Since weight is attached to each page accessed in an access sequence, trivial access sequences are not generated as patterns by FWAP

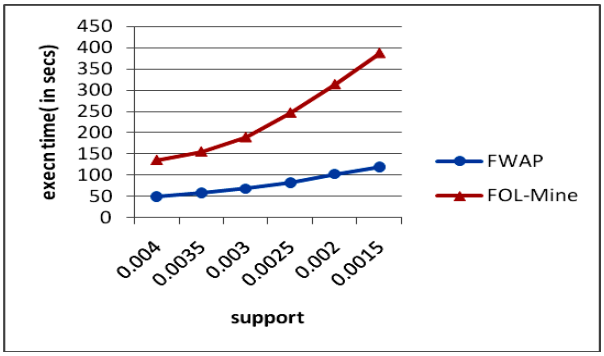

Figure 7. Execution Time Trend for FWAP and FOL-Mine for the Database T10I4D100K.

Table 1. Execution Time & Number of Patterns Generated for FWAP and FOL-Mine for the Database T10I4D100K.

support	execution time (sec)		Number of patterns	
	FWAP	FOL-Mine	FWAP	FOL-Mine
0.004	49	135	26	2001
0.0035	58	155	39	2761
0.003	68	189	58	4552
0.0025	82	247	104	7703
0.002	102	314	152	13255
0.0015	119	388	234	19126

4.1.2. Comparison of Number of Patterns Generated

Storing the huge volume of patterns generated is the primary concern in access pattern mining. To show how effectively the weighted pattern mining reduces the number of patterns, experiments are done with both weighted (FWAP) and non-weighted (FOL-Mine) access pattern methods. Experiments are conducted on real data set *msnbc* and synthetic dataset T10I4D100K. The result of the comparison over the database *msnbc* is given in Figure 8. Result of the experiment on T10I4D100K is provided in Figure 9. From the graphs it is evident that the proposed method reduces the number of pattern generated by the introduction of new weight constraints.

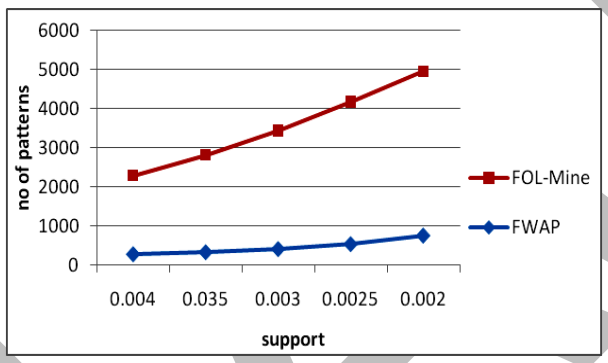

Figure 8. Comparison of Number of Patterns Generated by FWAP and FOL-Mine for the Database *msnbc*.

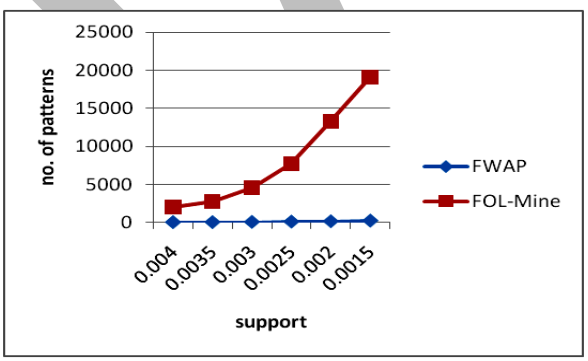

Figure 9. Comparison of Patterns Generated for FWAP and FOL-Mine for the Database T10I4D100K.

4.2. Comparison of Number of Patterns Generated and the Execution Time Requirement at Various Supports

In weighted access pattern mining we are introducing the additional constraints of weight into the mining process. This enables the method to pull out more meaningful patterns. Less significant patterns are neglected and this reduces the number of generated patterns. Managing the huge volume of pattern generated is an important concern in pattern mining.

The number of patterns generated in access pattern mining increases rapidly as the support threshold changes. This section of the paper evaluates the relation of execution time to the number of patterns generated at various support thresholds.

Detailed illustration of the number of patterns generated at various support and the required execution time is provided graphically.

The result of experiment using the dataset *msnbc* is given in Figure 10. The result for the data set T10I4D100K is provided in Figure 11. Number of patterns generated at various support for T25I10D10K is provided in Figure 12.

For all the three datasets, the proposed method itself shows a better performance. That is, by pulling out only the meaningful pattern the proposed method reduces the memory requirements.

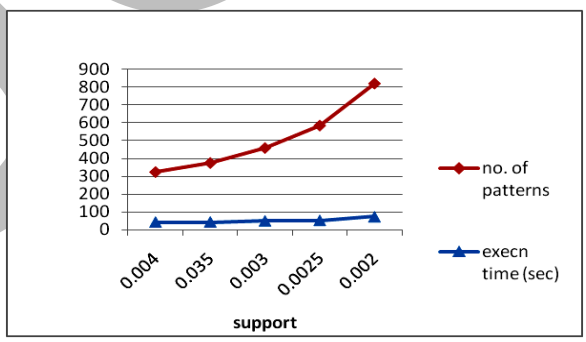

Figure 10. Patterns Generated at Various Support for the Database *msnbc*.

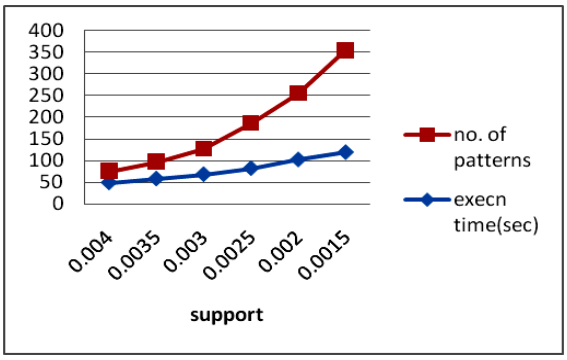

Figure 11. Patterns Generated at Various Support for the Database T10I4D100K.

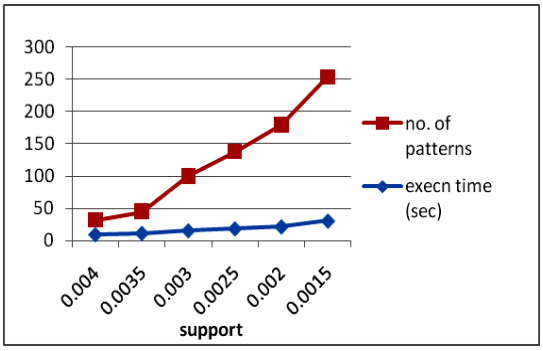

Figure 12. Patterns Generated at Various Support for the Database T25I10D10K.

4.3. Scalability Experiments

Scalability test tells how the algorithm performs when the size of input database increases. To test the scalability of the proposed algorithm experiments are done with both T25I10D10K and T10I4D100K. In the case of T25I10D10K dataset, scalability test are done by changing the database size from 2k to 10k. Results of this experiment are provided in Figure 13.

For the database T10I4D100K, database size is changed from 20k to 100k to conduct the scalability test. Support threshold is maintained as .0015 in both cases. The results are shown in the Figure 14.

The results show that the performance of the proposed method is not affected by the size of the database and can efficiently work on larger databases. Moreover, graphs provided in Figures 13 and 14 show that FWAP is linearly scalable and has better scalability than FOL-Mine.

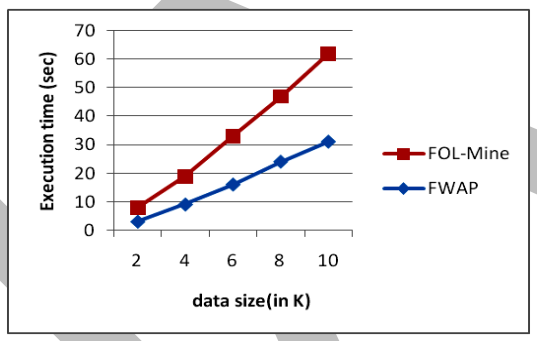

Figure 13. Scaling up Experiments on T25I10D10K Database at a Support Threshold of 0.015.

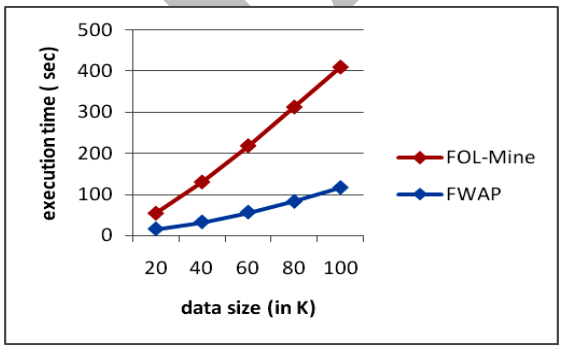

Figure.14. Scaling up Experiments on T10I4d100k Database at a Support Threshold of 0.0015.

5. Conclusion

A novel weighted access pattern mining algorithm is proposed in this paper. In weighted access pattern mining relative importance of each item in a sequence. Also is taken into consideration in addition to the order of item in the sequence, Experiments are conducted to compare the efficiency of the proposed method with existing non-weighted method. Extensive evaluation of the algorithm also proves the better processing time and lesser memory requirement by the proposed method. Scalability of the proposed method also has been proved and shown it to be linear.

References

[1] Agrawal, R., Srikant, R.: "*Fast Algorithms for Mining Association Rules in Large Databases*". Proc. 20th International Conference on Very Large Databases, Santiago, Chile, pp. 487-499, 1994.

[2] Ahmed, C. F., Tanbeer, S. K.: "Mining Weighted Frequent Patterns in Incremental Databases", *Trends in Artificial Intelligence*, pp. 933-938, 2008.

[3] Chang, J. H.: "Mining Weighted Sequential Patterns in a Sequence Database with a Time-Interval Weight". *Knowledge-Based Systems*, vol. 24, pp. 1-9, 2011.

[4] Chen. Y-L., Chiang, M.-C., Ko, M.-T.: "Discovering Fuzzy Time-Interval Sequential Patterns in Sequence Databases", *IEEE Transactions on Systems Man and Cybernetics – Part B: Cybernetics*, vol. 35, no. 5, pp. 959-972, 2005.

[5] Chen. Y-L., Huang, T.C.-H.: "Discovering Time-Interval Sequential Patterns in Sequence Databases", *Expert Systems with Applications*, vol. 25, no. 1, pp. 343–354, 2003.

[6] Cooley, R., Mobasher, B., Srivastava, J.: "Data Preparation for Mining World Wide Web Browsing Patterns". *Journal of Knowledge and Information Systems*, vol. 1, no.1, pp. 5-32, 1999.

[7] Forsat, R, Meybodii, M.R., Neiat, A.G.: "*Web Page Personalization Based on Weighted Association Rules*", Proc. International Conference on Electronic Computer Technology, pp. 130-135, 2009.

[8] Han et al.: "*FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining*", Proc. Sixth ACM SIGKDD International Conference on Knowledge discovery and Data Mining, pp. 355-359, 2000.

[9] Lu, Y.Ezeife, C.I.: "*Position Coded Pre-order Linked WAP-Tree for Web Log Sequential Pattern Mining*", Proc. 7th PAKDD, Seoul, Korea, pp. 337-349, 2003.

[10] Makris, C., Panagis, Y, Theodoridis, E, Tsakalidis, A.: "*Web-Page Usage Prediction Scheme Using Weighted Suffix Trees*", *LNCS 4726*, pp. 242–253, 2007.

[11] Pearson, E.A., Tang, P.: "*Mining Frequent Sequential Patterns with First-Occurrence Forests*", Proc. 46th ACM Southeastern Conference (ACMSE), Auburn, Alabama, pp. 34-39, 2008,

[12] Pei et al.: "*Mining Access Patterns Efficiently from Web Logs*", Proc. 4th PAKDD, Kyoto, Japan, pp. 396-407, 2000.

[13] Rajimol, A., Raju, G.:"Web Acces Pattern Mining – A Survey", *LNCS 6144*, Springer, pp. 24-31, 2010.

[14] Rajimol A., and Raju G.: "FOL-Mine – "A More Efficient Method for Mining Web Access Pattern", *Communications in Computer and Information Science*, vol. 191, no. 5, pp. 253-262, 2011.

[15] Srivastava, A., Bhosale, A.l, Sural, S.: "Speeding Up Web Access Using Weighted Association Rules", *LNCS 3776*, Springer, pp. 660-665, 2005.

[16] Tang, P,. Turkia, M.P., Gallivan, K.A.: "*Mining Web Access Patterns with First-Occurrence Linked WAP-Trees*", Proc. 16th International Conference on Software Engineering and Data Engineering (SEDE'07), Las Vegas, USA, pp. 247-252, 2007.

[17] Tao, F., Farid, M., Murtagh, F.: "*Weighted Association Rule Mining Using Weighted Support and Significant Framework*," Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 661-666, 2003.

[18] Yun, U., Leggett, J.J.: "*WFIM: Weighted Frequent Itemset Mining with a Weight Range and A Minimum Weight*". Proc. Fourth SIAM International Conference on Data Mining, USA, pp. 636–640, 2005.

[19] Yun, U.: "A New Framework for Detecting Weighted Sequential Patterns in Large Sequence Databases", *Knowledge-Based Systems*, vol. 21, no. 2, pp. 110-122, 2008.

[20] Yun, U.: "On Pushing Weight Constraints Deeply into Frequent Itemset Mining", *Intelligent Data Analysis*, vol. 13, no. 2, pp. 359–383, 2009.

[21] Zhou, B., Hui, S.C., Fong, A.C.M.: "CS-mine: An Efficient WAP-tree Mining for Web Access Patterns", *LNCS 3007*, Springer, pp. 523-532, 2004.

[22] Zhou, B., Hui, S.C., Fong, A.C.M.: "Efficient Sequential Access Pattern Mining for Web Recommendations", *International Journal of Knowledge Based and Intelligent Engineering Systems*, vol. 10, no. 2, pp. 155-168, 2006.

Gopalakrishna Kurup Raju received his Masters degree in Physics, Masters Degree in Computer Applications and Doctoral Degree in Computer Science from the University of Kerala, India, Master of Technology in Computer & Information Technology from Manonmaniam Sunderanar University, Tamilnadu, India. He is presently heading the Department of Information Technology, Kannur University, Kerala, India. He has more than twenty years of teaching experience. He has more than 100 publications in National and international journals / conference proceedings. His areas of interest include Web and Text mining, Fuzzy and Rough Set based algorithms in Data mining, Medical Image processing, Document Image processing and Digital Image based biometrics. He has successfully guided one PhD thesis.

Achuthan Nair Rajimol received her Bachelor Degree in Mathematics from Mahatma Gandhi University, Kottayam, Kerala, India in 1990, Masters Degree in Computer Applications from Anna University, Guindy, Madras, India in 1993. She is presently working as Assistant Professor in the Department of Computer Applications, Marian College, Kuttikkanam, Idukki, Kerala, India. She has 17 years of teaching experience and served as Member in Academic council and various Board of Studies in Mahatma Gandhi University, Kottayam, Kerala, India. Currently she is a research scholar in the School of Computer Sciences, Mahatma Gandhi University, Kottayam. Her Area of interest is Data mining and Web Mining.