**International Journal of Engineering Research and Development**
**e**-ISSN: 2278-067X, **p**-ISSN: 2278-800X, www.ijerd.com
Volume 6, Issue 2 (March 2013), PP. 09-13

# Utility Based Frequent Pattern Mining in an Incremental Database

Mercy Geraldine J[1], Anu Disney D[2]
[1]Head, Department of CSE, Srinivasan Engineering College, Perambalur, Tamilnadu, India.
[2]M.E. (C.S.E – Final Year), Srinivasan Engineering College, Perambalur, Tamilnadu, India.

**Abstract:-** Weighted Frequent Pattern Mining (WFPM) has brought the notion of the weight of the items into the Frequent Pattern mining algorithms. WFPM is practically much efficient than the frequent pattern mining. Several Weighted Frequent Pattern Mining methods have been used. However, they do not deal with the interactive and incremental database. A $IWFP_{TWU}$ algorithm has been proposed to allow the users to decide the level of interest and provides the direction for mining the interesting patterns. The Incremental Weighted Frequent Patterns based on Transaction Weighted Utility ($IWFP_{TWU}$) considers both the weight and the frequency of the item. The $IWFP_{TWU}$ arranges the items in a decreasing order of the Transaction Weighted Utilization (weighted Support). This makes uses of a single scan of the database for the construction of the $IWFP_{TWU}$ tree.

**Keywords:-** Incremental Database, Utility Support, Weighted Frequent Pattern, Weighted Support

## I. INTRODUCTION

Mining the important, correlated patterns efficiently is one of the main goals in data mining. Most of the weighted frequent pattern mining or association rule mining algorithms have adopted an Apriori algorithm based on the downward closure property [1]. They have suggested the sorted closure property [2], the weighted closure property [3] or other techniques [4] in order to satisfy the downward closure property.

Apriori based algorithms use candidate set generation and test approaches. It can be very expensive to generate and test all the candidates. Performance analyses of the existing algorithms have shown that frequent pattern growth algorithms are efficient at mining large databases and more scalable than Apriori-based approaches. However, there has been no weight based mining using the pattern growth algorithm because the downward closure property is broken by simply applying the FP-growth methodology. We will develop weighted frequent pattern mining based on the pattern growth method. Frequent pattern mining algorithms such as H-Mine, Frequency ordered prefix tree, etc, have better performance when a minimum support is high and the database is sparse. The main problem with these algorithms is that they can still generate an exponentially large number of patterns when a minimum support becomes lower. As the number of frequent patterns increases, the performance becomes worse. Although closed pattern mining approaches such as CLOSET[5], CARPENTER[6] etc are used, huge frequent patterns are still generated in large dense databases with a low minimum support. This problem also occurs in weight-based mining. We will describe ways to adjust the number of patterns through user feedback.

Previous weighted frequent pattern mining approaches use weight as one of the measures. The items are given different weights in the transaction database. However, they only focus on how to maintain the downward closure property. Although patterns satisfying pruning conditions are used in previous weighted pattern mining algorithms, they may have different characteristics, such as different levels of support and different levels of weight[4][7]. Some algorithms that suggest efficient and scalable mining methods in which users can decide their levels of interest and give directions for mining their own interesting patterns has to be developed. In addition, the databases are dynamic in nature and grow in a tremendous speed. Hence an algorithm that deals with the incremental and interactive nature of the database has been developed.

The $IWFP_{TWU}$ makes the use of the frequency and the weight of the items in the database. The weight of the item may be assigned according to the importance of the item. The $IWFP_{TWU}$ aims at arranging the transactions in the descending order of the weighted support or weighted frequency. It make use of a single scan of the database. The tree reuses the mined data, hence can be more effective. The tree structure makes it convenient to extract the most interesting patterns from the database easily.

## II. RELATED WORK

The introduction of the notion of weight into the frequent pattern mining has emerged as a major research area. Various frequent pattern mining methods/algorithms and incremental frequent pattern algorithms were used earlier. This section discusses about the earlier works in frequent pattern mining, incremental mining and the weighted frequent pattern mining.

Apriori algorithm[8] was introduced by Agarwal, et al in 1993. The Apriori algorithm involves the generation of several number of the candidate keys to find the frequent patterns in a given database[8].However, this requires more space for storage of the candidates generated and also it requires more number of computation. The FP growth algorithm has been proposed [7] to reduce the number of candidates that are generated[7]. It requires the database to be scanned and the items are entered into the header table .Then, the transactions are sorted in the descending order of the frequency to form the FP growth tree [7],[5].Both the Apriori algorithm and the FP growth algorithm do not apply to the dynamic database and requires more computations.

In reality databases grow rapidly. Hence it requires that the incremental data in the database have to be taken into consideration while mining the data .Number of algorithms have been developed to extract the data from the incremental databases. An algorithm to scan the original database and the incremental datasets separately was introduced in[8].The Fast Update Algorithm process the incremental data as they arrive. This processing may require two scans in the case the data is frequent in the incremental datasets. To reduce the number of scans the New Fast Update Algorithm (NFUP)[9] was introduced. This reduces the number of scans as it does not look into the original database once it is scanned. However, this does not offer a compact data structure. Hence, a compact data structure called the CAN Tree [10] was introduced. It is a tree based approach that arranges the items in some canonical order. Although the CAN-Tree is a compact data structure, it requires more time for merging the incremental data at the appropriate positions. This paved way for the advent of other algorithms like CATS- TREE[11] and COFI Tree. These algorithms consider the incremental nature of the databases; however, they do not take into account the weight of the items.

The weight of the items has to be taken into consideration so that the algorithm can be more effective in real world applications. The weight of the pattern is the average of the weight of the itemsets that constitute the pattern. i.e., the weighted support of the pattern $P=(X_1,X_2\ldots..X_n)$ is defined as,

$$Weight(P) = \frac{\sum_{q=1}^{length(P)} Weight(Xq)}{length(P)} \quad \ldots \quad (2.1)$$

A weighted support of a pattern is defined as the value that results from multiplying the pattern's support by the weight of the pattern. The weighted support of the pattern, *P*, is given as follows,

$$\mathbf{WSupport_{(P)}} = \mathbf{Support(P) * weight(P)} \quad \ldots (2.2)$$

The Transaction Weighted Utility (TWU) of an item can be calculated as follows:

$$\mathbf{WUtility_{(X)}} = \mathbf{Frequency(X) * weight(X)} \quad \ldots (2.3)$$

**Table I** Transaction Table

| TID | |
|---|---|
| T1 | a,b,c,d,g,h |
| T2 | a,e,f |
| T3 | b,e,f,g,h |
| T4 | a,b,c,d |
| T5 | a,b,g,h |
| T6 | a,b,d,e |

**Table II** Header Table

| Item | W | F | TWU |
|---|---|---|---|
| A | 0.6 | 7 | 4.2 |
| B | 0.5 | 6 | 3 |
| C | 0.2 | 3 | 0.6 |
| D | 0.3 | 4 | 1.2 |
| E | 0.5 | 3 | 1.5 |
| F | 0.3 | 2 | 0.6 |
| G | 0.8 | 5 | 4.0 |
| H | 0.38 | 2 | 0.76 |

**Table III** Sorted Header Table

| Item | W | F | TWU |
|---|---|---|---|
| A | 0.6 | 7 | 4.2 |
| G | 0.8 | 5 | 4.0 |
| B | 0.5 | 6 | 3 |
| E | 0.5 | 3 | 1.5 |
| D | 0.3 | 4 | 1.2 |
| H | 0.38 | 2 | 0.76 |
| C | 0.2 | 3 | 0.6 |
| F | 0.3 | 2 | 0.6 |

Several algorithms that push the weight constraint into the pattern mining have been introduced [10].Each item is given a different weight according to their importance in the databases. Unlike the previous frequent pattern algorithms, the weighted frequent pattern mining algorithms not only focus on the frequency of the item but also the weight of the items/ patterns. While the Weighted Frequent Pattern Mining tries to push the weight constraints into the Frequent Pattern Mining, it also aims at maintaining the downward closure property. The algorithms like LPMiner[13] mines the frequent patterns based on some support constraints. However, this does not consider the weight of the items. The WLPMiner considers the notion of weight while mining the frequent patterns. The WFIM [14] is the most widely used weighted frequent pattern mining approach. WFIM

maintains the downward closure property thus avoiding the least frequent patterns. But WFIM requires two scans of the database to discover the frequent patterns. These Weighted Frequent Pattern Mining algorithms apply only for the static databases. Hence, an efficient algorithm for mining the weighted frequent pattern in an incremental database is essential.

## III. PROPOSED SYSTEM

### A. PROPOSED SYSTEM

Incremental Weighted Frequent Pattern based on Transaction Weighted Utility ($IWFP_{TWU}$) makes use of a compact prefix tree data structure to store the items. A prefix tree is an ordered tree with any predefined order such as the ascending or descending order or any lexicographic order. The items from the transactions are scanned and then inserted into the prefix tree in some predefined order, thus reducing the time of mining the relevant and most interesting patterns.

In the $IWFP_{TWF}$ tree the data or the transactions are arranged in the descending order of the weighted utility of the items. The weighted Frequency is obtained by the product of the frequency of the item and the weight of the item in the database. The $IWFP_{TWU}$ based mining involves two different steps as shown in Fig1.They are as follows:

**STEP 1**: Compress the data into an $IWFP_{TWU}$ tree
**STEP 2**: Handle the incremental datasets
**STEP 3**: Mine the frequent patterns using the $IWFP_{TWU}$ algorithm

### B. Creation of $IWFP_{TWU}$ Tree

$IWFP_{TWU}$ tree is a compact data structure which is a form of the enhanced FP tree. The $IWFP_{TWU}$ based mining starts with the creation of the $IWFP_{TWU}$ tree. The $IWFP_{TWU}$ tree is created in the same way as that of an enhanced FP growth tree with a single scan of the database. The transactions in the database are scanned initially and the items are added into the header table that maintains the transaction weighted frequency. After every transaction has been scanned, the header table gets rearranged in the descending order of the weighted frequency or the weighted support. The header table thus maintains the frequency of the items too. Consider the set of transaction in Table I. The header table before sorting is shown in Table II. The header table after sorting the items in the descending order of the Weighted Frequency has been shown in Table III. The tree is created with the null node as the root and then inserts the items of the transactions at its child position and the process goes on

The overall process can be given in the following steps:

**STEP 1:** Scan the transactions in the dataset or the database and arrange the items in the weighted utilization/utility descending order
**STEP 2:** Increment the count of the item in the header table H
**STEP 3:** Calculate the Weighted Utilization for each item in the database
**STEP 4:** Arrange the transactions in the decreasing order of the weighted utilization
**STEP 5:** Insert the items of the transaction into a prefix tree and ensure that it maintains the downward closure property.

The downward closure property states that if a pattern is infrequent, then all its super patterns are also infrequent. The $IWFP_{TWU}$ tree performs the insertion and deletion operations. The downward closure property has to be maintained while constructing the $IWFP_{TWU}$ tree. Maintaining this property helps to prune the infrequent data or the items.

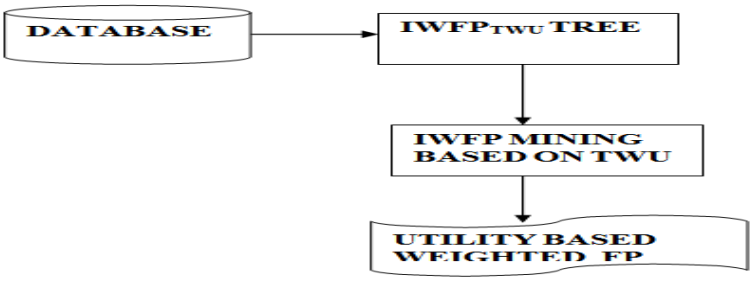


**Figure 1** Block diagram of the $IWFP_{TWU}$ based mining

### C. Processing the Incremental Data

The incremental insertion is performed on the arrival of $db^+$ database. The transactions that are arranged in the descending order of the weighted utilities are added as the leaf nodes of any super pattern that is available in the existing $IWFP_{TWU}$ tree. However, before inserting the data or the transactions into the prefix tree

or $IWFP_{TWU}$ tree, it is required to increase the count of the item in the header table, H. The items in the transactions are inserted into the prefix tree after sorting the items in the transactions in the descending order of their weighted utility.

The incremental deletion is performed when a $db^-$ arrives for the deletion of the data. The deletion of the data is simple when compared to that of the deletion process. The data to be deleted are deleted from the prefix tree leaving a vacant space, which is filled later by one of its child node. The downward closure property has to be maintained during the deletion process too. The deletion of the items or transaction requires decrementing the frequency count of the items in the prefix tree by one and then deleting the item or the itemset from the available $IWFP_{TWU}$ tree.

### D. Mining Using $IWFP_{TWU}$ Algorithm

The mining of the data using the $IWFP_{TWU}$ tree requires the $IWFP_{TWU}$ tree as the input. The $IWFP_{TWU}$ algorithm makes use of two other manual inputs: the user defined threshold value and the pattern to be mined. The mining process in the $IWFP_{TWU}$ algorithm commences by getting the user threshold along with the pattern from the user. The entire $IWFP_{TWU}$ mining algorithm is performed on the $IWFP_{TWU}$ tree rather than mining the database. Initially $IWFP_{TWU}$ generates a prefix tree for the item or the itemset to be mined. The items in the prefix tree are tested against the threshold using a candidate test. The weighted support of the pattern to be mined is compared against the threshold value provided by the user. This paves way for mining the relevant pattern and to prune the irrelevant pattern. The items that pass the candidate test generation form the nodes of the conditional tree. The conditional tree provides the frequent patterns in the descending order of the weighted utilization of the items.

## IV. EXPERIMENTAL EVALUATIONS

The $IWFP_{TWU}$ makes use of the concept of enhanced FP for creating the tree. This algorithm can provide the user with more relevant and interesting data. The $IWFP_{TWU}$ is more efficient as it considers both the weight and the frequency of the items in the database. The efficiency of this algorithm can be evaluated based on the memory efficiency and the runtime efficiency.

### 4.1 Memory Usage

The $IWFP_{TWU}$ normally uses a tree based data structure for storing the data. This makes the storage size more compact. The $IWFP_{TWU}$ tree is created over a single scan of the database. The incremental database is scanned and added at the appropriate position, thus making the structure compact and reducing the time of merging each and every data later.

### 4.2 Runtime Efficiency

The $IWFP_{TWU}$ tree is based on the descending order of the Transaction Weighted Utility. The runtime of the $IWFP_{TWU}$ tree considers the time for the creation of the tree and the time for mining the patterns from the tree. In this case the db+ is incremented by 0.05 million and the $IWFP_{TWU}$ takes considerably lesser time than that of the $IWFP_{WA}$ (based on weight ascending order). Fig 2 shows the runtime required for mining the data using the $IWFP_{TWU}$. Moreover, using BSM for restructuring reduces the time of restructuring.

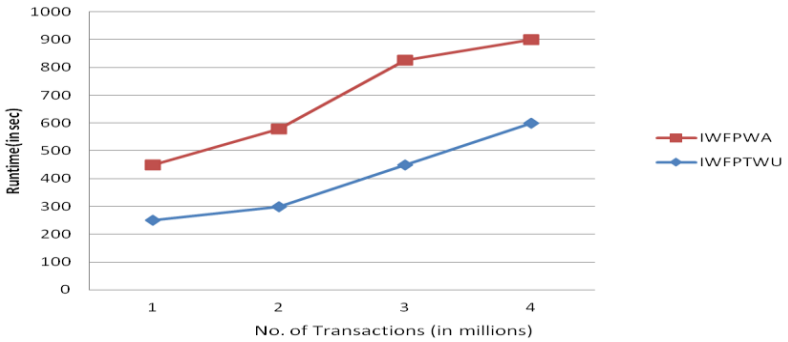


**Figure 2** Database is increasing $db^+$=0.05 million transaction

## V. CONCLUSIONS

This paper proposes a novel tree structure and a single pass algorithm for mining Incremental Weighted Frequent Patterns in an incremental database. It is studied that the $IWFP_{TWU}$ allows the user to mine the frequent patterns over a single scan, while the earlier existing algorithms requires more number of scans. Moreover, pushing the weight constraint into the frequent pattern mining can be proved to be much beneficial in various

practical applications. The $IWFP_{TWU}$ considers the weight as well as the frequency of the item in the database. The $IWFP_{TWU}$ tree makes use of "build once-mine many" concept that reuses the computed data and is suitable for interactive mining. The implementation of the $IWFP_{TWU}$ tree has shown that it involves slightly higher number of computations, but it is negligible when compared to the gain achieved in the mining phase. The efficiency can be further improved by taking the cost of the item into consideration even while storing the data i.e, considering the utility of the item at the time of creating the tree.

.

## REFERENCES


[1]. R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pp. 207-216, June 1993.

[2]. B. Liu, W. Hsu, and Y. Ma, "Mining Association Rules with Multiple Minimum Supports," Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 337-341, Aug. 1999.

[3]. F. Tao, "Weighted Association Rule Mining Using Weighted Support and Significant Framework," Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 661-666, Aug. 2003.

[4]. C. H. Cai, A. W. Chee Fu, C. H. Cheng, and W. W. Kwong. "Mining Association Rules with Weighted Items," Proceedings of the Sixth International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2005), July 1998.

[5]. J. Pei, and J. Han, "CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets," Proceedings of the Fifth ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, pp. 21-30, May 2000.

[6]. F. Bonchi, and C. Lucchese, "On Closed Constrained Frequent Pattern Mining," Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM 2004), pp. 35-42, Nov. 2004.

[7]. Aditya Telang, Chengkai Li, and Sharma . Agrawal, T. Imielinski, and A. Swami, (1993), "Mining Association Rules between Sets of Items in Large Databases," Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pp. 207-216.

[8]. J. Han, J. Pei, Y. Yin, and R. Mao,(Jan 2004), "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach," Data Mining and Knowledge Discovery, vol. 8, pp. 53-87.

[9]. W. Cheung , "Frequent Pattern mining without candidate generation or support constraint." Master's thesis, University of Alberta, SPRING '03, doi.ieeecomputersociety.org/10.1109/IDEAS.2003.1214917.

[10]. D. W. Cheung, J. Han, V. T. Ng, and C. Y. Wong, (Feb 1996), "Maintenance of discovered association rules in large databases: an incremental updating technique," In Proc. 12th Intl. Conf. on Data Engineering, New Orleans, LA, pp. 106-114.

[11]. Carson Kai-Sang Leung, Quamrul I. Khan, Tariqul Hoque,(2005), "CanTree: a tree structure for efficient incremental mining of frequent patterns", ICDM'05

[12]. F Gharib Tarek, Hamed Nassar, Mohamed Taha , Ajith Abraham, (2010) "AN EFFICIENT ALGORITHM FOR INCREMENTAL MINING OF TEMPORAL ASSOCIATION RULES", IEEE Transaction on Data and knowledge Engineering.

[13]. M. Seno, and G. Karypis, (Nov/Dec 2001) "LPMiner: An Algorithm for Finding Frequent Itemsets Using Length-Decreasing Support Constraint," *Proceedings of the First IEEE International Conference on Data Mining (ICDM 2001)*, pp. 505-512.

[14]. U. Yun, and J. J. Leggett, (April 2005), "WFIM: Weighted Frequent Itemset Mining with a Weight Range and a Minimum Weight," Proceedings of the Fourth SIAM International Conference on Data Mining, pp. 636-643.