

WIP: mining Weighted Interesting Patterns with a strong weight and/or support affinity

Unil Yun and John J. Leggett
Department of Computer Science
Texas A&M University
College Station, Texas 77843, USA
{yunei, leggett}@cs.tamu.edu

ABSTRACT

In this paper, we present a new algorithm, Weighted Interesting Pattern mining (WIP) in which a new measure, weight-confidence, is developed to generate weighted hyperclique patterns with similar levels of weights. A weight range is used to decide weight boundaries and an h-confidence serves to identify strong support affinity patterns. WIP not only gives a balance between the two measures of weight and support, but also considers weight affinity and/or support affinity between items within patterns so more valuable patterns can be generated. A comprehensive performance study shows that WIP is efficient in weighted frequent pattern mining. Moreover, it generates fewer but more valuable patterns for users.

1. INTRODUCTION

To overcome problems of Apriori based algorithms [1, 2], such as generation and test of all candidates and repeatedly scanning a large amount of the original database, pattern growth based approaches [3, 4, 5] have been developed. Pattern growth methods mine the complete set of frequent patterns using a divide and conquer method to reduce the search space without generating all the candidates. Most algorithms use a support constraint to prune the search space. This strategy provides basic pruning but the resulting patterns have weak affinity after mining datasets to obtain frequent patterns. Although the minimum support can be increased, it is not effective in generating patterns with increased weight and/or support affinity. In this paper, we propose an efficient mining algorithm called WIP (Weighted Interesting Pattern mining) based on mining weighted frequent patterns. We define the concept of a weighted hyperclique pattern that uses a new measure, called weight-confidence, to consider weight affinity and prevent the generation of patterns with substantially different weight levels. The weight confidence is used

to generate patterns with similar levels of weights and the h-confidence serves to identify strong support affinity patterns. The Hyperclique Miner [9] which first used an h-confidence, adopted an Apriori algorithm. An extensive performance analysis shows that weighted interesting patterns are extremely valuable patterns, since they have strong affinity in terms of a weight and/or support. Users can adjust the number of frequent patterns or find more valuable patterns by using weighted frequent patterns or weighted frequent patterns with weight affinity and/or support affinity, instead of only obtaining frequent patterns by changing the support threshold. Weighting applications exist in which items have different importance and patterns with a similar level of support and/or weight are more meaningful. For example, strong support and/or weight affinity measures can be applied in DNA analysis and web log mining by giving more weights to specific DNA patterns or web log sequences.

2. Problem Definition and Related Work

2.1 Problem Definition

Let $I = \{i_1, i_2, \dots, i_n\}$ be a unique set of items. A transaction database, TDB, is a set of transactions in which each transaction, denoted as a tuple $\langle \text{tid}, X \rangle$, contains a unique tid and a set of items. A pattern is called a k-pattern if it contains k items. A pattern $\{x_1, x_2, \dots, x_n\}$ is also represented as x_1, x_2, \dots, x_n . The support of a pattern is the number of transactions containing the pattern in the database. A weight of an item is a non-negative real number that shows the importance of the item. We can use the term, weighted itemset to represent a set of weighted items. A simple way to obtain a weighted itemset is to calculate the average value of the weights of the items in the itemset. The weight of each item is assigned to reflect the importance of each item in the transaction database. A weight is given to an item within a weight range, $W_{\min} \leq WR \leq W_{\max}$. Table 1 shows the

transaction database TDB. Table 2 shows example sets of items with different weights. If the minimum support threshold (min_sup) is 2, the frequent list is $\langle a:4, b:4, c:4, d:5, f:3, g:3, h:2 \rangle$. As defined by previous studies [2, 7, 8, 10, 11], the problem of weighted frequent pattern mining is to find the complete set of patterns satisfying a support constraint and a weight constraint in the database.

TID	Set of items
100	a, b, c, d, g
200	a, b, d, f
300	a, b, c, d, g, h
400	c, f, g
500	a, b, c, d
600	d, f, h

Table 1: transaction database TDB

2.2 Related Work

Previous weighted frequent pattern mining algorithms [2, 7, 8] are based on the Apriori algorithm which uses a candidate set generation and test mechanism. WFIM [10] is the weighted frequent itemset mining algorithm to use a pattern growth algorithm. WFIM focused on the downward closure property while maintaining algorithm efficiency. Patterns generated by WFIM have weak support and/or weight affinity patterns. WFIM uses a weight range to adjust the number of patterns. However, WFIM does not provide ways to remove patterns that include items with different support and/or weight levels. It would be better if the weak affinity patterns could be pruned first, resulting in fewer patterns after mining.

To mine correlated patterns, interesting measures [4, 6] have been suggested. In Hyperclique Miner [9], a hyperclique pattern was defined that generates patterns involving items with a similar level of support. The Hyperclique Miner has the following limitations. First, Hyperclique Miner has adopted an Apriori algorithm which uses candidate set generation and test approaches. It is very costly to generate and test all the candidates. Second, our performance tests show that many cases exist in which the number of hyperclique patterns is not increased or decreased, even though the minimum support is changed. Additionally, the minimum h-confidence must be set very high, otherwise, the number of hyperclique patterns becomes very large or is unchanged even if the minimum h-confidence is increased. The h-confidence can be an effective measure, but the effect is not always present in real datasets.

Item	a	b	c	d	f	g	h
Support	4	4	4	5	3	3	2
Weight ($0.7 \leq WR_1 \leq 1.1$)	1.1	1.0	0.8	1.0	0.7	0.9	0.8
Weight ($0.4 \leq WR_2 \leq 0.8$)	0.5	0.8	0.6	0.4	0.7	0.6	0.6
Weight ($0.2 \leq WR_3 \leq 0.6$)	0.6	0.5	0.3	0.3	0.2	0.5	0.4

Table 2: example sets of items with different WRs

3. WIP (Weighted Interesting Pattern mining)

In WIP, the main approach of WIP is to push weight confidence and/or h-confidence into the weighted frequent pattern mining algorithm based on the pattern growth approach and prune uninteresting patterns. A level of weight and/or support is considered to reflect the overall weight and/or support affinity among items within the pattern. In WIP, a new measure of weight confidence is defined and related properties are described. We present our algorithm in detail (Section 3.4) and give examples to illustrate the effect of weight confidence and weighted hyperclique patterns.

3.1 Weighted hyperclique pattern

In this section, we define the weight confidence measure and explain the concept of weighted hyperclique patterns.

Definition 3.1 Weight confidence (w-confidence)

Weight confidence of a pattern $P = \{i_1, i_2, \dots, i_m\}$, denoted as $wconf(P)$, is a measure that reflects the overall weight affinity among items within the pattern. It is the ratio of the minimum weight of items within the pattern to the maximum weight of items within the pattern. That is, this measure is defined as

$$wconf(P) = \frac{\text{Min}_{1 \leq j \leq m} \{\text{weight}(\{i_j\})\}}{\text{Max}_{1 \leq k \leq m} \{\text{weight}(\{i_k\})\}}$$

Definition 3.2 Weighted Hyperclique Pattern

A pattern is a weighted hyperclique pattern if the weight confidence of a pattern is greater than or equal to a minimum weight confidence. In other words, given a set of items $I = \{i_1, i_2, \dots, i_m\}$, a pattern P is a hyperclique pattern if and only if $|P| > 0$ and $wconf(P) \geq \text{min_wconf}$, where min_wconf is a minimum weight confidence.

Example 1: consider a pattern $P = \{A, B, C\}$ and $P' = \{D, E, F\}$. Assume that a minimum weight confidence

is 0.5, $\text{weight}(\{A\}) = 0.2$, $\text{weight}(\{B\}) = 0.5$, $\text{weight}(\{C\}) = 0.8$, $\text{weight}(\{D\}) = 0.4$, $\text{weight}(\{E\}) = 0.5$, and $\text{weight}(\{F\}) = 0.6$, where $\text{weight}(X)$ is the weight value of a pattern X . Then, the average weight of pattern P and pattern P' are both 0.5, $\text{wconf}(P) = 0.25$ and $\text{wconf}(P') = 0.67$. Therefore, pattern P is not a weighted hyperclique pattern but pattern P' is a weighted hyperclique pattern.

3.3 Mining weighted interesting patterns

In this section, we define weighted interesting pattern mining and show the mining process.

Definition 3.3 Weight Range (WR)

The weight of each item is assigned to reflect the importance of each item in the transaction database. A weight is given to an item with a weight range, $W_{\min} \leq W \leq W_{\max}$.

Definition 3.4 Weighted Interesting Pattern (WIP)

A pattern is a weighted interesting pattern if the following pruning conditions

Pruning condition 1: (support * MaxW (MinW) \geq min_sup)

In a transaction database, the value of multiplying the support of itemsets with a maximum weight (MaxW) among items in the transaction database is greater than or equal to a minimum support. In conditional databases, the value of multiplying the support of an itemset with a minimum weight (MinW) of a conditional pattern in the FP-trees is greater than or equal to a minimum support.

Pruning condition 2: (w-confidence \geq min_wconf)

The pattern is a weighted hyperclique pattern if the w-confidence, as given in definition 3.1, of a pattern is no less than a minimum w-confidence.

Pruning condition 3: (h-confidence \geq min_hconf)

The pattern is a hyperclique pattern if the h-confidence of a pattern is no less than a minimum h-confidence. Given a pattern $P = \{i_1, i_2, \dots, i_m\}$, h-confidence of the pattern P is defined as $\text{support}(\{i_1, i_2, \dots, i_m\}) / \text{Max}_{1 \leq k \leq m} \{\text{support}(\{i_k\})\}$. The h-confidence for a pattern P has the following upper bound: $\text{upper}(\text{hconf}(P)) = \text{Min}_{1 \leq j \leq m} \{\text{support}(\{i_j\})\} / \text{Max}_{1 \leq k \leq m} \{\text{support}(\{i_k\})\}$.

3.3.1 FP-tree structure and Bottom up Divide and Conquer

We use the transaction database TDB in Table 1 and use $0.2 \leq \text{WR}_3 \leq 0.6$ as the weight range from Table

2. Assume that min_sup is 2, min_wconf is 0.6 and min_hconf is 0.8. After the first scan of the transaction database, we know that the frequent list is $\langle a:4, b:4, c:4, d:5, f:3, g:3, h:2 \rangle$, the weight list is $\langle a:0.6, b:0.5, c:0.3, d:0.3, f:0.2, g:0.5, h:0.4 \rangle$, and the MaxW is 0.6.

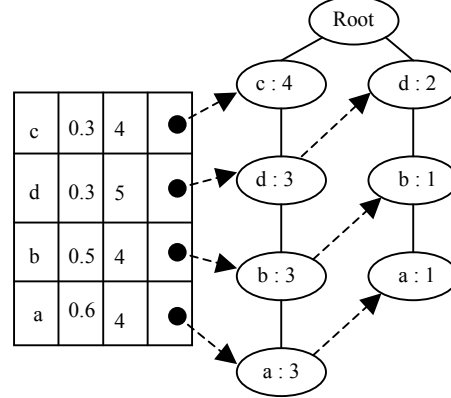


Fig. 1. The global FP tree

When an item is inserted in the FP-tree, weighted infrequent items are removed and the rest are sorted by weight ascending order. From pruning by weighted support constraint, items “f”, “g” and “h” are pruned because the values of multiplying the supports of items “f”, “g” and “h” with a MaxW (0.6) is less than a minimum support (2). Fig. 1 presents the global FP-tree and the related header table. After the global FP-tree is generated from the transaction database, WIP mines weighted interesting patterns from the FP tree. The weighted interesting patterns are generated by adding items one by one. WIP adapts the divide and conquer approach for the mining process.

WIP divides mining the FP-tree into mining smaller FP trees. From the global FP-tree, WIP mines (1) the patterns containing item “a” which have the highest weight, (2) the patterns including “b” but not “a”, (3) the patterns containing “d” but no “a” or “b”, and finally (4) the patterns containing “c” but no “a”, “b” or “d”. For node “a”, we generate a conditional database by starting from a’s head and following a’s node link. The conditional database for prefix “a:3” contains two transactions: $\langle bdc:3 \rangle$ and $\langle bd:1 \rangle$. In WIP, item “c:3” is pruned by the weighted support constraint of condition 1 in definition 3.4 because the value (1.8) of multiplying item c’s support (3) with a MinW (0.6) of the conditional pattern is less than the minimum support (2). In addition, a local item “d:4” is pruned by weight confidence. The candidate pattern, from a local item “d:4” and a conditional pattern “a” is “ad:4” and the weight confidence (0.5) of the candidate pattern “ad:4” is less than the minimum weight confidence (0.6). Note that there is only one item “a” in the conditional pattern, so the MinW of the

conditional pattern is 0.6. After that, Fig. 2 (a) shows a FP-tree with prefix “a”. For node “b”, WIP derives a conditional pattern (b:3) and two paths in the FP-tree: $\langle dc:3 \rangle$, and $\langle d:1 \rangle$. A local item “c:3” is pruned by the weighted support constraint of condition 1 in definition 3.4 because the value (1.8) of multiplying the support (3) of an item “c” with MinW (0.6) of a conditional pattern “b” is less than the minimum support (2). After pruning weighted uninteresting patterns in the conditional database, the projected FP-tree for the prefix “b:3” is constructed. Fig 2 (b) shows a conditional FP-tree for prefix b:3. For node “d”, the conditional database for prefix “d:3” contains a transaction: $\langle c:3 \rangle$. In this way, we can build local projected FP-trees from the global FP-tree and mine weighted interesting patterns from them recursively.

3.4 WIP ALGORITHM

In WIP, an ascending weight order method and a bottom-up traversal strategy are used in mining weighted interesting patterns. WIP defines a w-confidence measure and generates weighted hyperclique patterns. Items are given different weights within the weight range. We now show the weighted interesting pattern mining process and present the mining algorithm.

WIP algorithm: mining Weighted Interesting Patterns with a weight confidence and/or an h-confidence

- Input:** (1) A transaction database: TDB,
 (2) Minimum support: min_sup,
 (3) Weights of the items within weight range: w_i ,
 (4) Minimum w-confidence: min_wconf
 (5) Minimum h-confidence: min_hconf

Output: The complete set of weighted hyperclique patterns.

Begin

1. Let WIP be the set of weighted interesting patterns that satisfy the constraints. Initialize $WIP \leftarrow \{\}$;
2. Scan TDB once to find the global weighted frequent items satisfying the following definition: A pattern is a weighted frequent pattern if the following pruning condition is not satisfied.

Pruning condition: $(\text{support} * \text{MaxW} < \text{min_support})$

In a transaction database, the value of multiplying the support of a pattern with a MaxW of each item in the transaction database is less than a minimum support.

3. Sort items of WIP in weight ascending order. The sorted weighted frequent item list forms the weighted frequent list.
4. Scan the TDB again and build a global FP-tree using weight_order.
5. Call WIP (FP-tree, $\{\}$, WIP)

Procedure WIP (Tree, α , WIP)

- 1: For each a_i in the header of Tree do
 - 2: set $\beta = \alpha \cup a_i$;
 - 3: Get a set I_β of items to be included in β conditional database, CDB_β ;
 - 4: For each item in I_β ,
 Compute its count in β conditional database;
 - 5: For each b_j in I_β do
 - 6: If $(\text{sub}(\beta b_j) * \text{MinW} < \text{min_support})$ delete b_j from I_β ;
 - 7: If $(\text{wconf}(\beta b_j) < \text{min_wconf})$ delete b_j from I_β ;
 - 8: If $(\text{hconf}(\beta b_j) < \text{min_hconf})$ delete b_j from I_β ;
 - 9: End for
 - 10: $\text{Tree}_\beta \leftarrow \text{FP_Tree_Construction}(I_\beta, CDB_\beta)$
 - 11: If $\text{Tree}_\beta \neq 0$ then
 - 12: Call WIP (Tree_β , β , WIP)
 - 13: End if
 - 14: End for
-

4. PERFORMANCE EVALUATION

In this section, we report our experimental results on the performance of WIP in comparison with two recently developed algorithms, WFIM [10] and Hyperclique Miner [9]. The main purpose of this experiment is not only to show the effectiveness of w-confidence in generating weighted hyperclique patterns but also to demonstrate how effectively the weighted interesting patterns can be generated by using a w-confidence and/or an h-confidence. The real dataset used is Connect. WIP was written in C++. Experiments were performed on a sparcv9 processor operating at 1062 MHz, with 2048MB of memory. All experiments were performed on a Unix machine. In our experiments, a random generation function was used to generate weights for each item.

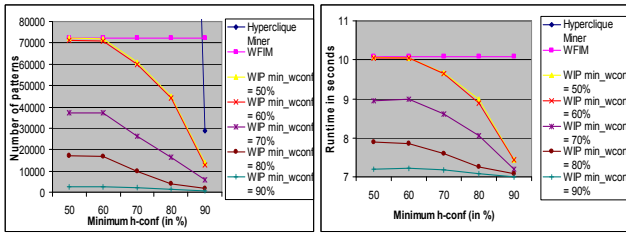


Fig. 3. Num of WIP

Fig. 4. Runtime

(Connect, Min_sup = 10%)

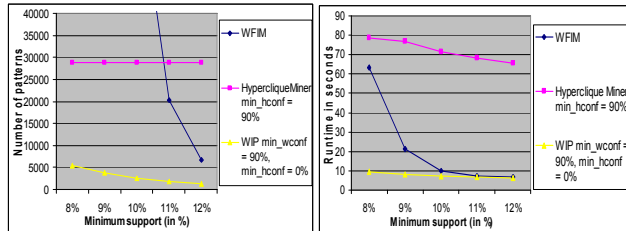


Fig. 5. Num of WIP(Connect) Fig. 6. Runtime (Connect)

In this test, we focused on the efficiency of w-confidence. First, we evaluated the performance on the Connect dataset. We set up a weight range from 0.1 to 0.2. Fig. 3 and Fig. 5 show that WIP generates fewer patterns than WFIM and Hyperclique Miner. Specifically, fewer patterns are generated as the w-confidence is increased. In Fig. 4 and Fig. 6, we can see that WIP is faster than WFIM and Hyperclique Miner. In Fig. 5 and Fig. 6, WIP did not use an h-confidence but only a w-confidence for the performance test of the w-confidence. The number of patterns discovered by WIP is several orders of magnitude fewer than the number of patterns found by Hyperclique Miner. Note that, in Fig. 3, Hyperclique Miner generates a huge number of patterns with h-confidence of less than 90%. From Fig. 5, we see that the number of patterns generated by Hyperclique Miner is unchanged although the minimum support is reduced. Hyperclique Miner uses two thresholds, a minimum h-confidence and a minimum support. However, from Fig. 3, we see that the number of patterns increases quickly for the Hyperclique Miner as a minimum h-confidence is decreased. In Fig. 4, we could not show the runtime of Hyperclique Miner because the runtime becomes so much larger as the minimum h-confidence is decreased. For example, the runtimes of Hyperclique Miner are 71 seconds with a minimum h-confidence of 90%, 146.27 seconds with a minimum h-confidence of 80% and 814.15 seconds with a minimum h-confidence of 70%. As a reverse case, the number of patterns in Hyperclique Miner is not reduced although a minimum support is increased in Fig. 5. Meanwhile, runtime is increased when the minimum support becomes lower showing that the h-confidence is not always effective with different

parameter settings in real datasets. In Fig. 3 and Fig. 4, the number of patterns and runtime is not changed in WFIM because WFIM does not use an h-confidence measure.

5. CONCLUSION

In this paper, we defined the problem of mining weighted interesting patterns. We introduced a w-confidence measure and the concept of weighted hyperclique patterns using the w-confidence. Our main goal in this framework is to push a w-confidence and an h-confidence into a weighted frequent pattern mining algorithm based on the pattern growth method and to prune weak affinity patterns. In WIP, the w-confidence and/or the h-confidence are used to avoid generating patterns that involve items with different weight and/or support levels. The extensive performance analysis shows that WIP is efficient and scalable in weighted frequent itemset mining.

6. ACKNOWLEDGEMENT

We would like to thank Hui Xiong not only for providing executable code of Hyperclique Miner but also for giving valuable comments.

7. REFERENCES

- [1] Rakesh Agrawal, Tomasz Imieliński, Arun Swami, *Mining association rules between sets of items in large databases*, ACM SIGMOD, May 1993.
- [2] C. H. Cai, Ada Wai-Chee Fu, C. H. Cheng, and W. W. Kwong, *Mining association rules with weighted items*. IDEAS'98, July 1998.
- [3] Jiawei Han, Jian Pei, Yiwen Yin, *Mining frequent patterns without candidate generation*, ACM SIGMOD, May 2000.
- [4] Young-Koo Lee, Won-Young Kim, Y. Dora Cai, and Jiawei Han, *CoMine: Efficient Mining of Correlated Patterns*, IEEE ICDM, 2003.
- [5] Guimei Liu, Hongjun Lu, Yabo Xu, Jeffrey Xu Yu: *Ascending Frequency Ordered Prefix-tree: Efficient Mining of Frequent Patterns*. DASFAA 2003.
- [6] Edward R. Omiecinski. *Alternative Interest Measures for Mining Associations in Databases*. IEEE Transaction on Knowledge and Data Engineering, 2003.
- [7] Feng Tao, *Weighted Association Rule Mining using Weighted Support and Significant framework*. ACM SIGKDD, Aug 2003.
- [8] Wei Wang, Jiong Yang, Philip S. Yu, *Efficient mining of weighted association rules (WAR)*, ACM SIGKDD, Aug 2000.
- [9] Hui Xiong, Pang-Ning Tan and Vipin Kumar, *Mining Strong Affinity Association Patterns in Data Sets with Skewed Support Distribution*, IEEE ICDM, 2003.
- [10] Unil Yun, John J. Leggett, *WFIM: Weighted Frequent Itemset Mining with a weight range and a minimum weight*, SDM'05, April 2005.
- [11] Unil Yun, John J. Leggett, *WLPMiner: Weighted Frequent pattern Mining with Length decreasing support constraints*, PAKDD'05, May 2005.