Mining High Utility Itemsets using Transactional Databases

Megha V, Priyanka L, Sowmya P S, Sunitha H S
Department of Computer Science and Engineering
T. John Institute of Technology,
Bangalore, India

Roopashree S
Assistant Professor,
Department of Computer Science and Engineering,
T.John Institute of Technology
Bangalore, India

Abstract**—**Mining high utility itemsets from database refers to finding the itemsets with high utility like profits. There are many relevant algorithms that are proposed in the recent years. In this paper, we propose an algorithm namely, *utility pattern growth* (*UP-Growth*)for discovering high utility itemsets and a compact tree-based structure called *utility pattern tree* (*UP-tree*) for maintaining important information related to utility patterns within databases proposed. High-utility itemsets can be generated from UP-tree efficiently with only two scans of original database. Several strategies are proposed for facilitating the mining processes of UP-Growth by maintaining only essential information in Up-tree. By these strategies, overestimated utilities of candidates can be well reduced by discarding utilities of the items that cannot be high utility and are not involved in the search space. Different types of both real and synthetic data sets are used in a series of experiments to compare the performance of the proposed algorithms with the state-of-the-art utility mining algorithms.

Experimental results show that UP-Growth outperform other algorithms substantially in terms of execution time, especially when databases contain lots of long transactions or low minimum utility thresholds are set.

Keywords**—**Candidate pruning, frequent itemset, high utility itemset, utility mining, data mining.

I. Introduction

DATA mining, an interdisciplinary subfield of computer science is the process of revealing nontrivial, previously unknown and potentially useful pattern or information from large databases. The overall goal of the data mining process is to extract information from a large dataset and transfer it into an understandable structure. Discovering of the useful patterns hidden in a database plays an important role in several data mining tasks, such as frequent pattern mining, weighted frequent pattern mining, and high utility pattern mining. Among them, frequent pattern mining is a fundamental research topic which is used in different kinds of databases, such as transactional databases [1], [8], [13] and streaming databases
[11], [18].

In frequent pattern mining, the relative importance of each item is not considered. The weighted association rule mining was proposed [4], [17], [19], [22], [26], [27], [28] to solve the problem of frequent pattern mining.

In this method the items which have high weights can be found even if they do not appear frequently. This method cannot satisfy the requirements of the users who want to find the itemsets which have high profits, since profits are concerned with unit profits in weights and purchase quantities but not with the quantities of items.

Utility mining is an important topic in data mining field. Mining high utility itemsets from databases refers to finding the itemsets with high profits. Here, itemset utility means the interestingness, importance and profitability of an item to users. Utility of items in a transaction database consists of two aspects: 1) the importance of distinct items, which is called *external utility*, and 2) the importance of items in transactions, which is called *internal utility*. The product of its external utility and its internal utility is the Utility of an itemset. An itemset is called a *high utility itemset* if its utility is no less than a user-specified minimum utility threshold; otherwise, it is called a *low-utility itemset*. The various applications of mining high utility itemsets from databases are website click stream analysis [9], [16], [20], online e-commerce management, cross marketing in retail stores [3], [6], [12], [21], [23], [24] and so on.

In existing studies [3], [6], [9], [10], [12], [15], [20], [21], overestimated methods are applied to facilitate the performance of utility mining. In this, the potential high utility itemsets (PHUIs) are found first, and later the database scan is performed to identify their utilities. But this method generates a large number of PHUIs and it degrades the mining performance when there are many long transactions. There are more number of PHUIs generated here and it leads to the consumption of higher processing time and thus decreases the mining performance.

To solve this problem, two algorithms and a compact data structure has been proposed to efficiently discover the high utility itemsets from transactional databases. Major contributions of this work is as follows:

1. An algorithm, named *utility pattern growth* (*UP-Growth*), and a compact tree structure, called *utility pattern tree* (*UP-Tree*) are proposed. High-utility itemsets can be generated from UP-Tree efficiently with only two scans of original databases.
2. In the mining processes of UP-Growth only the important and essential information is maintained in the UP-Tree. Thus by doing this, the utilities of the items that cannot be high utility or that are not involved in the search space are discarded. This proposed algorithms in addition to decreasing the overestimated utilities of PHUIs, but they also greatly reduce the number of candidates.
3. Both real and synthetic data sets are used to compare the performance of the proposed algorithms. The outcome of the experiments show that the UP-Growth and UP-Growth+ algorithms outperform the other algorithms in terms of execution time, especially when databases contain lots of long

transactions or low minimum utility thresholds are set.

II. Background

Among the issues of frequent pattern mining, the most famous are association rule mining [1], [7], [8], [13], [25], [29] and sequential pattern mining [2], [14]. One of the well-known algorithms for mining association rules is Apriori [1]. It is used for efficiently mining association rules from large databases.

Later, to achieve a better performance than Apriori based algorithm, FP-Growth algorithm [8] was proposed. FP-Growth algorithm finds frequent itemsets without generating any candidate itemset and scans database just twice.

In the framework of frequent itemset mining, the importance of items to users is not considered. Thus, the topic called weighted association rule mining was proposed [4], [17], [19], [22], [26], [27], [28]. However, since the framework of weighted association rules does not have downward closure property, mining performance cannot be improved.

Hence the the concept of weighted downward closure property [19] was proposed. Since weighted association rule mining considers only the importance of items, but not the quantities in transactions. Thus, the issue of high utility itemset mining is raised.

An algorithm named Two- Phase [12] was proposed to overcome this problem which is mainly composed of two mining phases. Two-phase algorithm reduces search space, but it generates too many candidates and requires multiple database scans. Thus isolated items discarding strategy (IIDS) [10] was proposed to reduce the number of candidates. However, this algorithm still scans database for several times to find high utility itemsets.

To efficiently generate High Transaction Weighted UtilityItemsets (IHUPs) and avoid scanning database too many times, a tree-based algorithm, named IHUP [3] was proposed. A tree- based structure called IHUP-Tree is used to maintain the information about itemsets and their utilities. Each node of an IHUP-Tree consists of an item name, a total weighted utility value and a support count.

Although IHUP achieves a better performance than IIDS and Two-Phase, it still produces too many HTWUIs. Such a large number of HTWUIs will degrade the mining performance in terms of execution time and memory consumption.

Let us consider an example database,

Given a finite set of items I = { $i_{1,}$ i_2,....,i_m }, each item i_p (1≤ p≤ m) has a unit profit pr(i_p). An itemset X is a set of k distinct items { i_1, i_2, ..., i_k }, where i_j € I, 1≤ j≤ k. k is the length of X. An itemset with length k is called a k-itemset. A transaction database D = {T_1, T_2, ..., T_n } contains a set of transactions, and each transaction T_d(1≤ d ≤ n) has a unique identifier d, called TID. Each item i_p in transaction T_d is associated with a quantity q(i_p, T_d), that is, the purchased quantity of i_p in T_d.

Now, in Tables 1 and 2,
u({A},T_1) = 5×1 = 5;
u({AD}, T_1) = u({A},T_1) + u({D}, T_1) = 5 + 2=7;
u({AD}) = u({AD}, T_1) + u({AD}, T_3) + u({AD}, T_6) = 7 + 22 + 7 = 36.
If min_util is set to 30, {AD} is a high utility itemset.

TABLE 1
An Example Database

TID	Transaction	TU
T_1	(A,1)(C,10)(D,1)	17
T2	(A,2) (C,6) (E,2) (G,5)	27
T3	(A,2) (B,2) (D,6) (E,2) (F,1)	37
T4	(B,4) (C,13) (D,3) (E,1)	30
T5	(B,2) (C,4) (E,1) (G,2)	13
T6	(A,1) (B,1) (C,1) (D,1) (H,2)	12

TABLE 2
Profit Table

Item	A	B	C	D	E	F	G	H
Profit	5	2	1	2	3	5	1	1

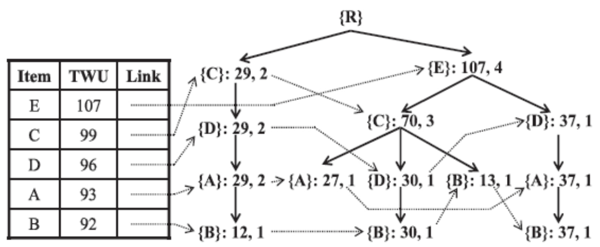

Fig. 1. An IHUP-Tree when min_util=40

Fig. 1 shows the global IHUP- Tree for the database in Table 1, in which items are arranged in the descending order of TWU. For each node in Fig. 1, the first number beside item name is its TWU and the second one is its support count.

To reduce itemsets' overestimated utilities several strategies are proposed. By applying those proposed strategies, the number of generated candidates can be highly reduced.

III. Mining Using Up-Growth

The framework of the proposed methods consists of three steps: 1) Scan the database twice to construct a global UP- Tree with the first two strategies; 2) recursively generate PHUIs from global UP-Tree and local UP-Trees by UP-Growth with the third and fourth strategies; and 3) identify actual high utility itemsets from the set of PHUIs. We use a new term "potential high utility itemsets" to distinguish the patterns found by our methods from HTWUIs. By our effective strategies, the set of PHUIs will become much smaller than the set of HTWUIs.

A. The Proposed Data Structure: UP-Tree

To improve the mining performance and avoid scanning original database repeatedly, we use a compact tree structure, named UP-Tree, to maintain the information of transactions and high utility itemsets. Two strategies are applied to minimize the overestimated utilities stored in the nodes of global UP-Tree, DGU (Discarding Global Unpromising Itemsets) and DGN (Decreasing Global Node Utilities). Here, the elements of UP-

Tree are first defined. Next, the two strategies are introduced.

In an UP-Tree, each node N consists of N.name, N.count, N.nu, N.parent, N.hlink and a set of child nodes. N.name is the node's item name. N.count is the node's support count. N.nu is the node's node utility, i.e., overestimated utility of the node. N.parent records the parent node of N. N.hlink is a node link which points to a node whose item name is the same as N.name. A table named header table is employed to facilitate the traversal of UP-Tree. In header table, each entry records an item name, an overestimated utility, and a link.

Strategy 1. DGU: Discarding Global Unpromising items and their actual utilities from transactions and transaction utilities of the database.

The construction of a global UP-Tree can be performed with two scans of the original database. In the first scan, transaction utility(TU) of each transaction is computed. At the same time, transaction weighted utility(TWU) of each single item is also accumulated. By transaction weighted downward closure(TWDC) property, an item and its supersets are unpromising to be high utility itemsets if its TWU is less than the minimum utility threshold. Such an item is called an unpromising item.

During the second scan of database, transactions are inserted into a UP-Tree. When a transaction is retrieved, the unpromising items should be removed from the transaction and their utilities should also be eliminated from the transaction's TU.

Strategy 2. DGN: Decreasing Global Node utilities for the nodes of global UP-Tree by actual utilities of descendant nodes during the construction of global UP-Tree.

Our second proposed strategy for decreasing overestimated utilities is to remove the utilities of descendant nodes from their node utilities in global UP-Tree. The process is performed during the construction of the global UP-Tree.

By applying strategy DGN, the utilities of the nodes that are closer to the root of a global UP-Tree are further reduced. DGN is especially suitable for the databases containing lots of long transactions. In other words, the more items a transaction contains, the more utilities can be discarded by DGN.

Construction of a Global UP-Tree by Applying DGU and DGN

The construction of a global UP-Tree is performed with two database scans. In the first scan, each transaction's TU is computed; at the same time, each 1- item's TWU is also accumulated. Thus, we can get promising items and unpromising items. After getting all promising items, DGU is applied. The transactions are reorganized by pruning the unpromising items and sorting the remaining promising items in a fixed order. Each transaction after the above reorganization is called a reorganized transaction.

Then a function Insert_Reorganized_Transaction is called to apply DGN during constructing a global UP-Tree.

Consider the transaction database in Table 1 and the profit table in Table 2. Suppose min_util is 50. In the first scan of database, TUs of all transactions and TWUs of distinct items are computed. Five promising items, i.e., {A}: 93, {B}: 92, {C}: 99, {D}: 96 and {E}: 107, are sorted in the header table by the descending order of TWU, that is, {E}, {C}, {D}, {A},

TABLE 3
Reorganized Transactions and their RTUs

TID	Reorganized transaction	RTU
T_1'	(C,10)(D,1)(A,1)	17
T_2'	(E,2)(C,6)(A,2)	22
T_3'	(E,2)(D,6)(A,2)(B,2)	32
T_4'	(E,1)(C,13)(D,3)(B,4)	30
T_5'	(E,1)(C,4)(B,2)	11
T_6'	(C,1)(D,1)(A,1)(B,1)	10

and {B}. Then the transactions are reorganized by sorting promising items and subtracting utilities of unpromising items from their TUs. The reorganized transactions and their RTUs are shown in Table 3. Comparing Tables 3 and 1, the RTUs of T2, T3, and T5 in Table 3 are less than the TUs in Table 1 since the utilities of {F}, {G}, and {H} have been removed by DGU.

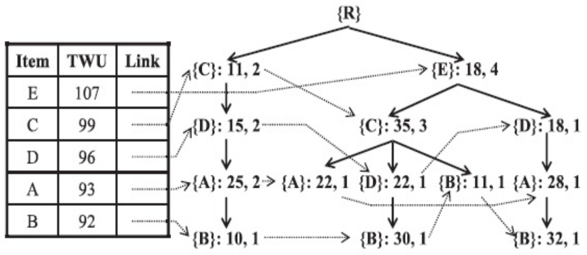

Fig. 2.A UP-Tree by applying strategies DGU and DGN

After inserting all reorganized transactions by the same way, the global UP-Tree shown in Fig. 2 is constructed. Comparing with the IHUP-Tree in Fig. 1, node utilities of the nodes in UP-Tree are less than those in IHUP-Tree since the node utilities are effectively decreased by the two strategies DGU and DGN.

B. The Proposed Mining Method: UP-Growth

After constructing a global UP-Tree using DGU and DGN, a basic method for generating PHUIs is to mine UP-Tree by FP-Growth [8]. However too many candidates will be generated. Thus, we propose an algorithm UP-Growth. By the strategies applied in UP-Growth Algorithm, the overestimated utilities of itemsets can be decreased and thus the number of PHUIs can be reduced. We propose the two strategies and then describe the process of UP-Growth.

Strategy 3. DLU: Discarding Local Unpromising Items during Constructing a Local UP-Tree.

The common method for generating patterns in tree-based algorithms [3], [8] contains three steps: 1) Generate conditional pattern bases by tracing the paths in the original tree; 2) construct conditional trees (also called local trees in this paper) by the information in conditional pattern bases; and 3) mine patterns from the conditional trees. In this, the actual utilities of unpromising items that need to be discarded in conditional pattern bases is not known unless an additional database scan is performed.

To overcome this problem, a naive solution is to maintain items' actual utilities in each transaction into each node of global UP-Tree. But this is impractical since it needs lots of memory space. So, the two strategies, named DLU (Discarding Local Unpromising Items) and DLN (Decreasing Local Node Utilities) are proposed which are applied in the first two mining steps.

DLN is used for decreasing local node utilities for the nodes of local UP-Tree by estimated utilities of descendant nodes. The same as DLU, DLN can be recognized as local version of DGN. By the two strategies, overestimated utilities for itemsets can be locally reduced in a certain degree without losing any actual high utility itemset.

TABLE 4
Minimum Item Utility Table

Item	A	B	C	D	E
Minimum item utility	5	2	1	2	3

Strategy 4. DLN: Decreasing Local Node Utilities during Constructing a Local UP-Tree

Since $\{i_m\}$-Tree must not contain the information about the items below i_m in the original UP-Tree, we can discard the utilities of descendant nodes related to i_m in the original UP-Tree while building $\{i_m\}$-Tree. (Here, original UP-Tree means the UP-Tree which is used to generate $\{i_m\}$-Tree.) Because we cannot know actual utilities of the descendant nodes, we use minimum item utilities to estimate the discarded utilities.

UP-Growth: Mining a UP-Tree by Applying DLU and DLN

The process of mining PHUIs by UP-Growth is described as follows: First, the node links in UP-Tree corresponding to the item im, which is the bottom entry in header table, are traced. Found nodes are traced to root of the UP-Tree to get paths related to im. All retrieved paths, their path utilities and support counts are collected into im's conditional pattern base.

A conditional UP-Tree can be constructed by two scans of a conditional pattern base. For the first scan, local promising and unpromising items are learned by summing the path utility for each item in the conditional pattern base. Then, DLU is applied to reduce overestimated utilities during the second scan of the conditional pattern base. When a path is retrieved, unpromising items and their estimated utilities are eliminated from the path and its path utility. Then the path is reorganized by the descending order of path utility of the items in the conditional pattern base.

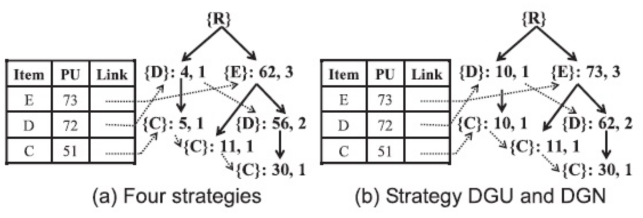

Fig. 3. {B}-Trees with different strategies

Fig. 3a with that in Fig. 3b, it can be observed that node utilities of the nodes are further reduced by the strategies DLU and DLN.

C. Efficiently Identify High Utility Itemsets

After finding all PHUIs, the third step is to identify high utility itemsets and their utilities from the set of PHUIs by scanning original database once. In our framework, overestimated utilities of PHUIs are smaller than or equal to TWUs of HTWUIs since they are reduced by the proposed strategies. Thus, the number of PHUIs is much smaller than that of HTWUIs. Therefore, the proposed method is much efficient than the previous methods and also high utility itemsets can be identified by scanning reorganized transactions. Since there is no un-promising item in the reorganized transactions, I/O cost and execution time can be further reduced. This technique works well especially when the original database contains lots of unpromising items.

IV. CONCLUSION

In this paper, an efficient algorithm named UP-Growth are proposed for mining high utility itemsets from transaction databases. A data structure named UP-Tree was proposed for maintaining the information of high utility itemsets. PHUIs can be efficiently generated from UP-Tree with only two database scans. Moreover, several strategies were developed to decrease overestimated utility and enhance the performance of utility mining. In the experiments, both real and synthetic data sets were used to perform a thorough performance evaluation. Results show that the strategies considerably improved performance by reducing both the search space and the number of candidates. Moreover, the proposed algorithm, outperforms the state-of-the-art algorithms substantially especially when databases contain lots of long transactions or a low minimum utility threshold is used.

REFERENCES

[1] R. Agrawal and R. Srikant, “Fast Algorithms for Mining Association Rules,” Proc. 20th Int’l Conf. Very Large Data Bases (VLDB), pp. 487-499, 1994.
[2] R. Agrawal and R. Srikant, “Mining Sequential Patterns,” Proc. 11th Int’l Conf. Data Eng., pp. 3-14, Mar. 1995.
[3] C.F. Ahmed, S.K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, “Efficient Tree Structures for High Utility Pattern Mining in Incremental Databases,” IEEE Trans. Knowledge and Data Eng., vol. 21, no. 12, pp. 1708-1721, Dec. 2009.
[4] C.H. Cai, A.W.C. Fu, C.H. Cheng, and W.W. Kwong, “Mining Association Rules with Weighted Items,” Proc. Int’l Database Eng. and Applications Symp. (IDEAS ’98), pp. 68-77, 1998.
[5] R. Chan, Q. Yang, and Y. Shen, “Mining High Utility Itemsets,” Proc. IEEE Third Int’l Conf. Data Mining, pp. 19-26, Nov. 2003.
[6] A. Erwin, R.P. Gopalan, and N.R. Achuthan, “Efficient Mining of High Utility Itemsets from Large Data Sets,” Proc. 12th Pacific-Asia Conf. Advances in Knowledge Discovery and Data Mining (PAKDD), pp. 554-561, 2008.
[7] J. Han and Y. Fu, “Discovery of Multiple-Level Association Rules from Large Databases,” Proc. 21th Int’l Conf. Very Large Data Bases, pp. 420-431, Sept. 1995.
[8] J. Han, J. Pei, and Y. Yin, “Mining Frequent Patterns without Candidate Generation,” Proc. ACM-SIGMOD Int’l Conf. Manage- ment of Data, pp. 1-12, 2000.

[9] H.F. Li, H.Y. Huang, Y.C. Chen, Y.J. Liu, and S.Y. Lee, "Fast and Memory Efficient Mining of High Utility Itemsets in Data Streams," Proc. IEEE Eighth Int'l Conf. on Data Mining, pp. 881- 886, 2008.

[10] Y.-C. Li, J.-S. Yeh, and C.-C. Chang, "Isolated Items Discarding Strategy for Discovering High Utility Itemsets," Data and Knowl- edge Eng., vol. 64, no. 1, pp. 198-217, Jan. 2008.

[11] C.H. Lin, D.Y. Chiu, Y.H. Wu, and A.L.P. Chen, "Mining Frequent Itemsets from Data Streams with a Time-Sensitive Sliding Window," Proc. SIAM Int'l Conf. Data Mining (SDM '05), 2005.

[12] Y. Liu, W. Liao, and A. Choudhary, "A Fast High Utility Itemsets Mining Algorithm," Proc. Utility-Based Data Mining Workshop, 2005.

[13] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang, and D. Yang, "H-Mine: Fast and Space-Preserving Frequent Pattern Mining in Large Databases," IIE Trans. Inst. of Industrial Engineers, vol. 39, no. 6, pp. 593-605, June 2007.

[14] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Moal, and M.C. Hsu, "Mining Sequential Patterns by Pattern-Growth: The Prefixspan Approach," IEEE Trans. Knowledge and Data Eng., vol.16, no.10, pp. 1424-1440, Oct. 2004.

[15] B.-E. Shie, H.-F. Hsiao, V., S. Tseng, and P.S. Yu, "Mining High Utility Mobile Sequential Patterns in Mobile Commerce Environ- ments," Proc. 16th Int'l Conf. DAtabase Systems for Advanced Applications (DASFAA '11), vol. 6587/2011, pp. 224-238, 2011.

[16] B.-E. Shie, V.S. Tseng, and P.S. Yu, "Online Mining of Temporal Maximal Utility Itemsets from Data Streams," Proc. 25th Ann. ACM Symp. Applied Computing, Mar. 2010.

[17] K. Sun and F. Bai, "Mining Weighted Association Rules without Preassigned Weights," IEEE Trans. Knowledge and Data Eng., vol. 20, no. 4, pp. 489-495, Apr. 2008.

[18] S.K. Tanbeer, C.F. Ahmed, B.-S. Jeong, and Y.-K. Lee, "Efficient Frequent Pattern Mining over Data Streams," Proc. ACM 17th Conf. Information and Knowledge Management, 2008.

[19] F. Tao, F. Murtagh, and M. Farid, "Weighted Association Rule Mining Using Weighted Support and Significance Framework," Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD '03), pp. 661-666, 2003.

[20] V.S. Tseng, C.J. Chu, and T. Liang, "Efficient Mining of Temporal High Utility Itemsets from Data Streams," Proc. ACM KDD Workshop Utility-Based Data Mining Workshop (UBDM '06), Aug. 2006.

[21] V.S. Tseng, C.-W. Wu, B.-E. Shie, and P.S. Yu, "UP-Growth: An Efficient Algorithm for High Utility Itemsets Mining," Proc. 16th ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD '10), pp. 253-262, 2010.

[22] W. Wang, J. Yang, and P. Yu, "Efficient Mining of Weighted Association Rules (WAR)," Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD '00), pp. 270-274, 200

[23] H. Yao, H.J. Hamilton, and L. Geng, "A Unified Framework for Utility-Based Measures for Mining Itemsets," Proc. ACM SIGKDD Second Workshop Utility-Based Data Mining, pp. 28-37, Aug. 2006.

[24] S.J. Yen and Y.S. Lee, "Mining High Utility Quantitative Association Rules." Proc. Ninth Int'l Conf. Data Warehousing and Knowledge Discovery (DaWaK), pp. 283-292, Sept. 2007.

[25] S.J. Yen, Y.S. Lee, C.K. Wang, C.W. Wu, and L.-Y. Ouyang, "The Studies of Mining Frequent Patterns Based on Frequent Pattern Tree," Proc. 13th Pacific-Asia Conf. Advances in Knowledge Discovery and Data Mining (PAKDD), vol. 5476, pp. 232-241, 2009.

[26] U. Yun, "An Efficient Mining of Weighted Frequent Patterns with Length Decreasing Support Constraints," Knowledge-Based Sys- tems, vol. 21, no. 8, pp. 741-752, Dec. 2008.

[27] U. Yun and J.J. Leggett, "WFIM: Weighted Frequent Itemset Mining with a Weight Range and a Minimum Weight," Proc. SIAM Int'l Conf. Data Mining (SDM '05), pp. 636-640, 2005.

[28] U. Yun and J.J. Leggett, "WIP: Mining Weighted Interesting Patterns with a Strong Weight and/or Support Affinity," Proc. SIAM Int'l Conf. Data Mining (SDM '06), pp. 623-627, Apr. 2006.

[29] M.J. Zaki, "Scalable Algorithms for Association Mining," IEEE Trans. Knowledge and Data Eng., vol. 12, no. 3, pp. 372-390, May 2000.