Volume 4, Issue 2, February 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

UP-Growth Algorithms for Knowledge Discovery from Transactional Databases

Shoban Babu Sriramoju*
Associate Professor
Department of Computer Science and Engineering
Varadha Reddy Engineering College
Warangal, India

Madan Kumar.Chandran
Assistant Professor
Department of Computer Science and Engineering
Varadha Reddy Engineering College
Warangal , India

Abstract— Data mining has been around and all enterprises in the real world need it in order to make well informed decisions. The reason behind this is that analyzing huge data is not possible manually. For mining high utility item sets from databases many techniques came into existence. However, they tend to produce huge number of itemsets thus degrading the performance of the system in terms of space and time. Recently Tseng et al. proposed UP-Growth and UP-Growth+ algorithms for mining high utility item sets from transactional databases. Moreover the generated itemsets are maintained in a tree structure named UP-tree that will reduce the number of scans of database. This has improved performance considerably. In this paper we study on the problem of generating high utility item sets and implement the methods proposed by Tseng et al. We built a prototype application that demonstrates the proof of concept. The empirical results reveal that the methods are useful in the real world applications.

Keywords— Data mining, high utility item sets, candidate pruning, frequent item sets

I) INTRODUCTION

Data mining is used to process huge number of records in database and discover interesting facts that can help in making well informed decisions. Many algorithms came into existence for data mining. Algorithms such as Apriori [1] are used to mine the association rules. Later on weighted frequent pattern mining came into existence as the previous methods do not consider the importance of generated item sets. Transactional [2], [3], [1] and streaming databases [4], [5] were also used to experiment on data mining. Other databases considered are time series databases [6], [7]. As data mining is essential for enterprises, there is increasing research into the item set mining. The results of this mining are very useful to enterprises in making policy decisions. The businesses such as insurance and banking for instance use data mining algorithms in order to generate business intelligence that is used to make expert decisions that lead to profits. Section II provides more details on the literature which the names of the techniques are described here. In the process of generating frequent item sets, Apriori [1], FP-Growth [3] played an important role. These algorithms, however, produce large number of items. Recently Tseng et al. [8] overcame this drawback by introducing two algorithms namely UP-Growth and UP-Growth+ besides introducing a novel data structure by name UP-Tree. In this paper we implement the methods proposed in [8] using Java programming language. We built a prototype application that demonstrates the performance of these methods. The results revealed that they are better than existing methods in generating high utility item sets. The remainder of this paper is structured as follows. Section II provides review of literature that is related to data mining and the various methods used for frequent item set mining. Section III provides details of the algorithms implemented in this paper. Section IV presents experimental results while section V concludes the paper.

II) RELATED WORKS

Frequent data mining has been around and there were many studies on that as explored in [9], [10], [11], [12], [13], and [2]. All these studies reveal that data mining can be used to retrieve frequent item sets that can help in making some important decisions in the enterprise world. In the same fashion association rule mining has witnessed much research as explored in [11], [3], [2], and [1]. In [9] and [13] sequential pattern mining was presented. A widely used algorithm for association rule mining is Apriori [1] which is most efficient mining of rules from data sources. Later on pattern growth based algorithms came into existence. For instance GP-Growth [3] is one such algorithm for association rule mining. When compared to the Apriori, the FP-Growth algorithm achieved better performance. It is also best in scanning database as it does it only twice. There are many algorithms on frequent item set mining. However, they do not consider the importance of generated items. To consider this fact weighted association rule mining came into existence [14], [15], [16], [17], [18] and [19]. As there was no downward closure property in these methods, the performance of the mining can’t be improved. In order to address this problem in [17] Tao et al. proposed weighted downward closure property. This has improved the performance by using transaction weight and other inputs. Studies were found in the similar lines as explored in [15], [19] and [20]. These studies have taken the importance of items in the item sets into consideration. However, they do not consider the quantities in the transactions. In order to overcome this issue high utility item set mining came into existence [21], [22], [23], [24], [25], [26], and [27]. Recently Tseng et al. proposed two algorithms such as UP-Growth and UP-Growth+ that helped in achieving better performance and reducing the number of items in the high utility item sets.

III) UP-Tree Data Structure and Methods

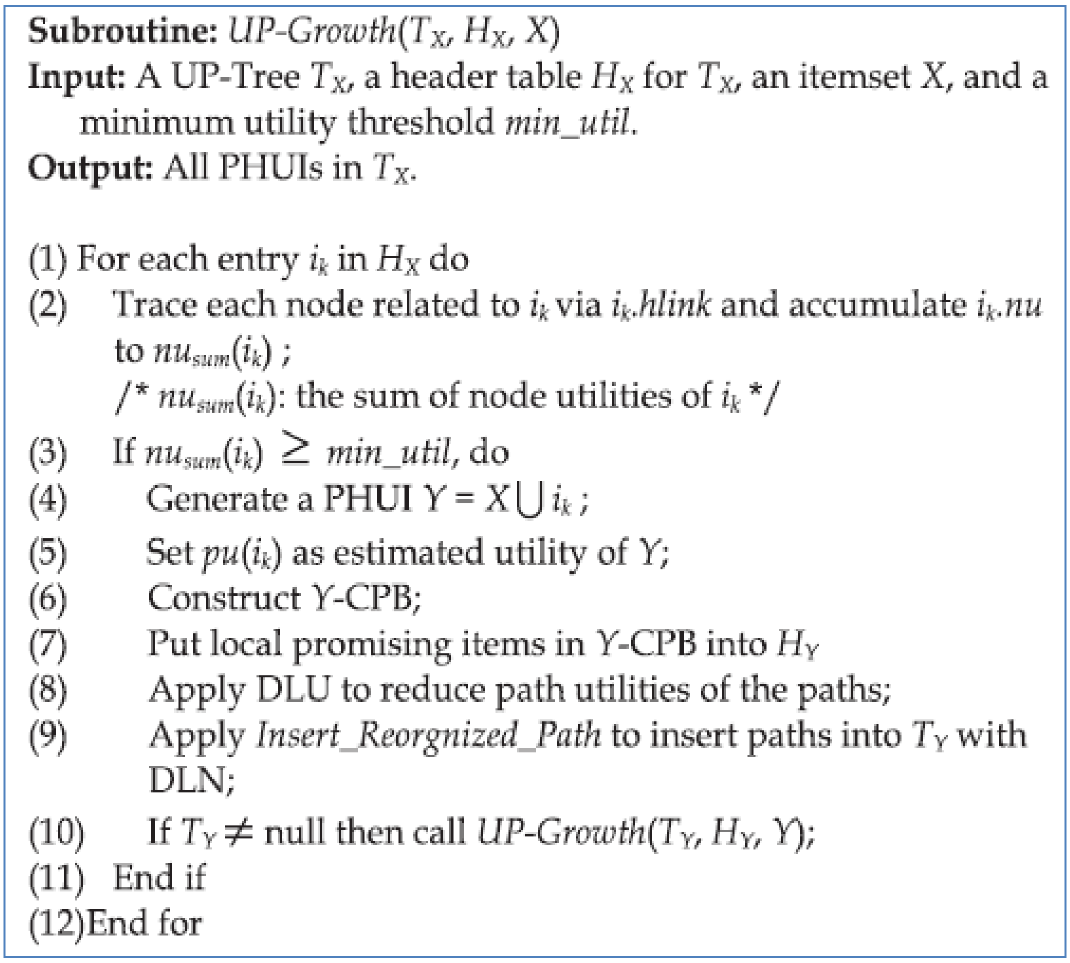
UP-Tree data structure and the data mining methods like UP-Growth and UP-Growth+ are originally proposed by Tseng et al. For more information about them the reader is suggested to read the reference paper [8]. However, we present the data structure and methods as briefly as possible. We implemented those methods using Java programming language and presented the results in the ensuing section.

A. UP-Tree Data Structure

This is a data structure which is used to hold high utility item sets that will help in reducing the number of round trips to database while improving mining performance. While constructing this dataset two strategies are followed. First, while constructing the tree global unpromising items are discarded. Second, while constructing the tree the global node utilities are decreased. The following sub-sections briefly describe the methods such as UP-Growth and UP-Growth+.

B. UP-Growth

The basic method to generate high utility item sets is the FP-Growth [3] algorithm. However, it produces huge number of item sets. In order to reduce the number of item sets and produce only high utility item sets UP-Growth algorithm [8] is used. The algorithm is as presented in Figure1.



```
Subroutine: UP-Growth($T_X$, $H_X$, $X$)
Input: A UP-Tree $T_X$, a header table $H_X$ for $T_X$, an itemset $X$, and a
    minimum utility threshold $min\_util$.
Output: All PHUIs in $T_X$.

(1) For each entry $i_k$ in $H_X$ do
(2)     Trace each node related to $i_k$ via $i_k.hlink$ and accumulate $i_k.nu$
        to $nu_{sum}(i_k)$ ;
        /* $nu_{sum}(i_k)$: the sum of node utilities of $i_k$ */
(3)     If $nu_{sum}(i_k) \geq min\_util$, do
(4)         Generate a PHUI $Y = X \bigcup i_k$ ;
(5)         Set $pu(i_k)$ as estimated utility of $Y$;
(6)         Construct $Y$-CPB;
(7)         Put local promising items in $Y$-CPB into $H_Y$
(8)         Apply DLU to reduce path utilities of the paths;
(9)         Apply Insert_Reorgnized_Path to insert paths into $T_Y$ with
            DLN;
(10)        If $T_Y \neq$ null then call UP-Growth($T_Y$, $H_Y$, $Y$);
(11)    End if
(12)End for
```

Figure 1 – UP-Growth Algorithm

As can be seen in Figure 1, it is evident that the UP-Growth algorithm takes UP-Tree and other parameters as input and generates high utility item sets from given data sources. It also applies pruning in the process. Due to the pruning its performance will be improved while reducing the number of high utility item sets.

C. UP-Growth+ Algorithm

When compared with FP-Growth UP-Growth algorithm has better performance. However, it has some limitation that is its performance is suboptimal as it does not fully use the fact that the overestimated utilities can be closer to their actual utilities. To overcome this problem UP-Growth+ [8] which is an improved form of UP-Growth was introduced. The algorithm presented in Figure 2 along with the algorithm in Figure 1 is used in order to improve the performance.

```
Subroutine : Insert_Reorganized_Path(N, $i_x$)
Line 1 : If N has a child $N_{i_x}$ such that $N_{i_x}.item = i_x$, increment $N_{i_x}.count$ by
        $p_j.count$. Otherwise, create a new child node $N_{i_x}$ with $N_{i_x}.item = i_x$,
        $N_{i_x}.count = p_j.count$, $N_{i_x}.parent$ = N and $N_{i_x}.nu = 0$.
Line 2 : Increase $N_{i_x}.nu$ by Eq (3).
Line 3 : If there exists a node $N_{i_x}$ in $p_j$ where $x + 1 < m'$,
        call Insert_Reorganized_Path($N_{i_x}$, $i_{x+1}$)
```

Figure 2 – A Subroutine Used By UP-Growth+

As seen in figure 2, this subroutine is invoked by UP-Growth+ in order to improve the performance. Thus the number of high utility item sets is decreased significantly. The results reveal the truly high utility item sets which can be used to make well informed and expert decision making.

IV) EXPERIMENTAL RESULTS

With our prototype application experiments are made to test the efficiency algorithms which were implemented. The environment used to execute the application includes a PC with 1 GB RAM, 2.80GHz Intel Pentium processor. The application was built in Java platform and executed in Windows XP operating system. Experiments were made with real and synthetic data sources. The synthetic data source was generated by writing a program in Java language while the real data sets were taken from Internet sources. The features of real datasets are as presented in Table 1.

Dataset	|D|	T	|I|	Type
Accidents	340,183	33.8	468	Dense
Chain-store	1,112,949	7.2	46,086	Sparse
Chess	3,196	37.0	75	Dense
Foodmart	4,141	4.4	1,559	Sparse

Table 1 – Features of real data sets

The methods implemented in this paper are compared with previous methods. The results presented in the following figures reveal that the performance of the methods of this paper show better performance when compared to that of previous ones.

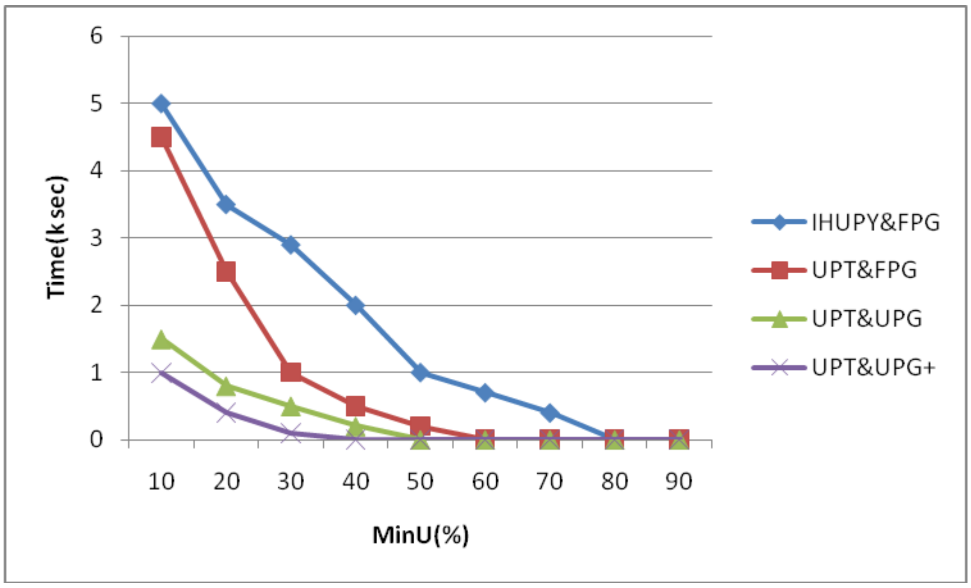


Figure 3– Performance comparison with respect to dense data set (Chess)

As can be seen in Figure 3, it is evident that the horizontal axis represents number of candidates while the vertical axis represents the runtime. Out of all the methods the UPT & UPG+ method has higher performance. This is due to the reason that the number of generated candidates is taken into consideration.

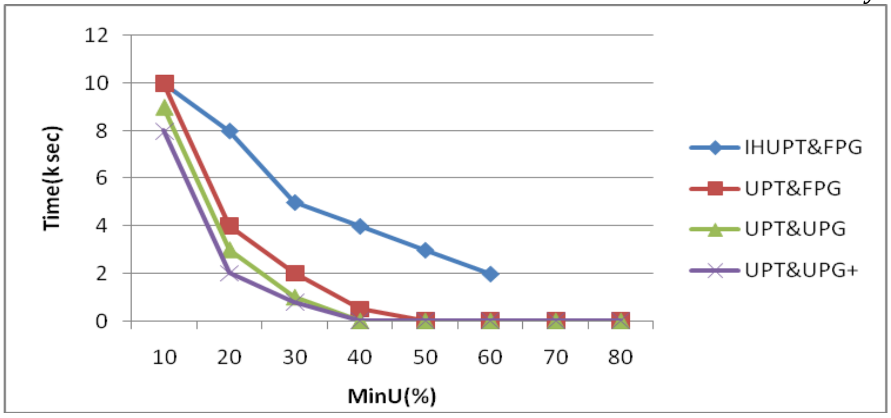


Figure 4 – Performance comparison of runtime for phase II (Foodmart)

As can be seen in Figure 4, it is evident that the horizontal axis represents number of candidates while the vertical axis represents the runtime. Out of all the methods the UPT & UPG+ method has higher performance. This is due to the reason that the number of generated candidates is taken into consideration.

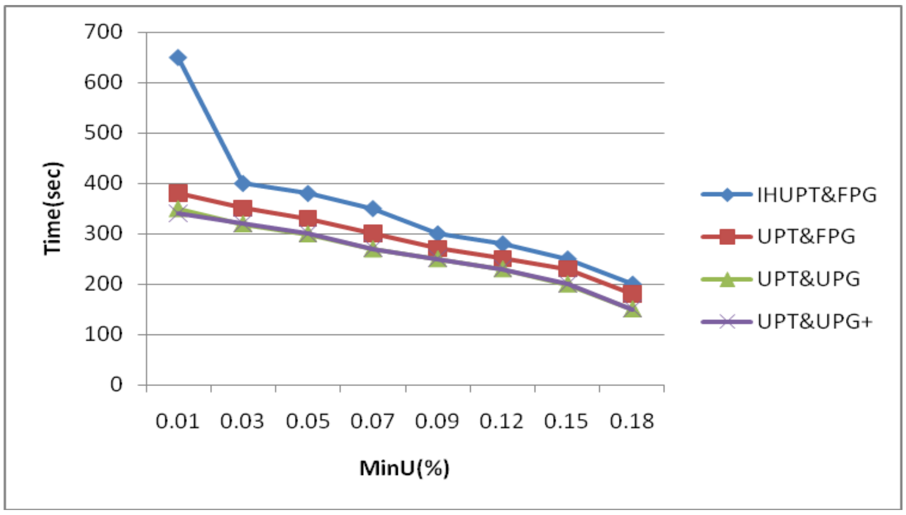


Figure 5 – Performance comparison on sparse data set (Foodmart)

As can be seen in Figure 5, it is evident that the horizontal axis represents number of candidates while the vertical axis represents the runtime. Out of all the methods the UPT & UPG+ method has higher performance. The runtime is proportional to number of candidates as expected. The reason behind is that the overhead of scanning is very high.

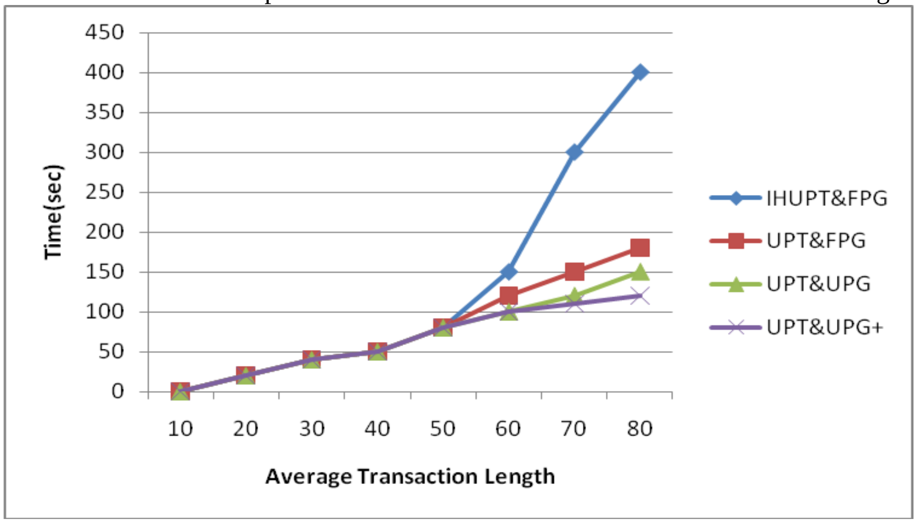


Figure 6 – Average transaction length

As can be seen in Figure 6, it is evident that the horizontal axis represents the average transaction length while the vertical axis represents time in seconds. The results reveal that the runtime is increased as there is increase in the average transaction length. The UPT&UPG+ shows higher performance when compared to other methods.

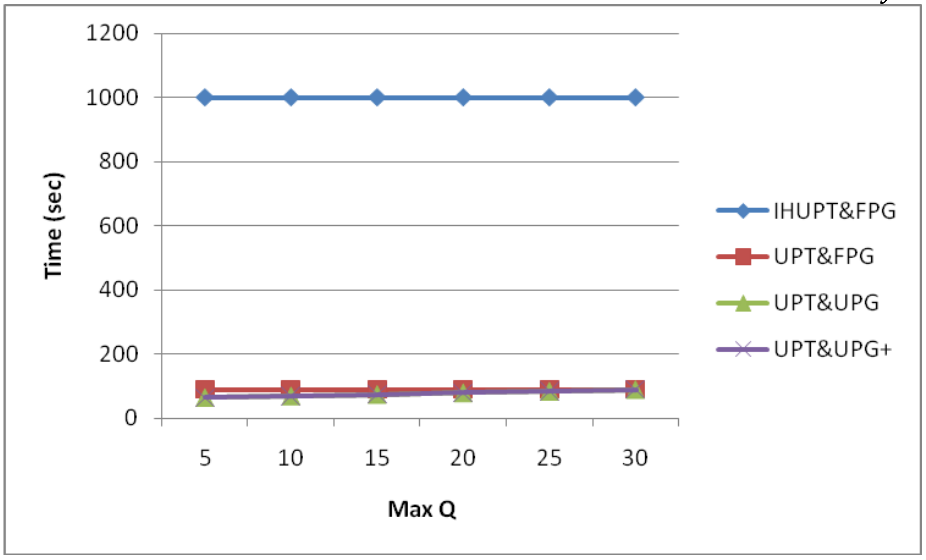


Figure 7 – Maximum number of purchases items (Accidents dataset)

As can be seen in Figure 7, it is evident that the horizontal axis represents number of candidates while the vertical axis represents the time in seconds. The time is increased when the number of candidates is increased. However, the runtime of the IHUPT&FPG is worst when compared with other methods.

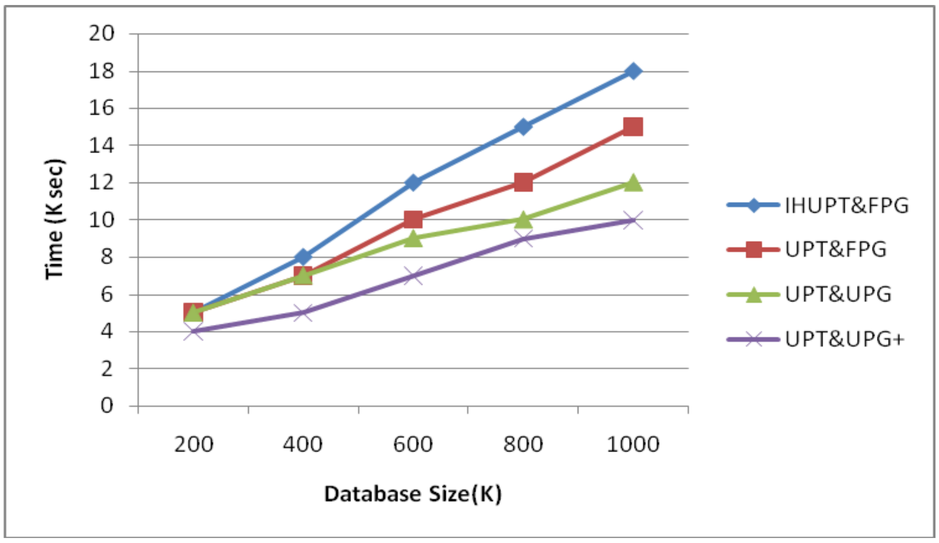


Figure 8 – Runtime comparison of methods

As can be seen in Figure 8, it is evident that the horizontal axis represents database size while the vertical axis represents time in seconds. The scalability of all the methods is reasonably good. Naturally as the database size grows, the time taken increases.

V) CONCLUSION

In this paper, we studied the many algorithms that are used to mining data sources. Especially we focused on the mining of high utility item sets from given data sources. We made experiments on the algorithms as presented in [8] such as UP-Growth and UP-Growth and UP Growth+. In the experiments, a data structure is constructed by name UP-Tree for holding the high utility item sets. Such item sets can be generated effectively from UP-Tree with less number of database scans. The overestimated utility has been decreased using many strategies as part of utility mining. The experiments are made using both real time and synthetic datasets. The performance of the implemented algorithms are presented and compared with previous methods. The results reveal that the methods implemented in this paper outperform the methods of previous ones found in the literature. From this it is understood that the prototype application can be used in the real world applications.

REFERENCES

[1] sR. 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] 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.

[3] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," Proc. ACM-SIGMOD Int'l Conf. Management of Data, pp. 1-12, 2000.

[4] 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.

[5] 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.

[6] J. Han, G. Dong, and Y. Yin, "Efficient Mining of Partial Periodic Patterns in Time Series Database," Proc. Int'l Conf. on Data Eng., pp. 106-115, 1999.

[7] M.Y. Eltabakh, M. Ouzzani, M.A. Khalil, W.G. Aref, and A.K. Elmagarmid, "Incremental Mining for Frequent Patterns in Evolving Time Series Databases," Technical Report CSD TR#08-02, Purdue Univ., 2008.

[8] Vincent S. Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S. Yu, Fellow, IEEE (Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases). IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 25, NO. 8, AUGUST 2013.

[9] 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.

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

[11] 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.

[12] 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.

[13] R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc. 11th Int'l Conf. Data Eng., pp. 3-14, Mar. 1995.

[14] 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.

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

[16] 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, 2000.

[17] C.H. Cai, A.W.C. Fu, C.H. Cheng, and W.W. Kwong, "Mining Association Rules with eighted Items," Proc. Int'l Database Eng. and Applications Symp. (IDEAS '98), pp. 68-77, 1998.

[18] 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.

[19] 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.

[20] J.H. Chang, "Mining Weighted Sequential Patterns in a Sequence Database with a Time-Interval Weight," Knowledge-Based Systems, vol. 24, no. 1, pp. 1-9, 2011.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

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

[26] 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.

[27] R. Chan, Q. Yang, and Y. Shen, "Mining High Utility Itemsets," Proc. IEEE Third Int'l Conf. Data Mining, pp. 19-26, Nov. 2003.