2017 5th Intl Conf on Applied Computing and Information Technology/4th Intl Conf on Computational Science/Intelligence
and Applied Informatics/2nd Intl Conf on Big Data, Cloud Computing, Data Science & Engineering

A Rapid Incremental Frequent Pattern Mining Algorithm for Uncertain Data

Tu-Liang Lin
Department of Management
Information System
National Chiayi University
Chiayi, Taiwan
tuliang@mail.ncyu.edu.tw

Bo-Wei Wen
Department of Management
Information System
National Chiayi University
Chiayi, Taiwan
s1051439@mail.ncyu.edu.tw

Hong-Yi Chang*
Department of Management
Information System
National Chiayi University
Chiayi, Taiwan
hychang@mail.ncyu.edu.tw

Wan-Kun Chang
Information and Resource
Technology Department
Industrial Technology Research Institute
Taichung City, Taiwan
wayne65@ itri.org.tw

Shih-Che Hsu
Information and Resource
Technology Department
Industrial Technology Research Institute
Taichung City, Taiwan
itri990508@ itri.org.tw

Abstract—**Association rule analysis is an important topic in data mining. Basket analysis is one of the most well-known applications. Store or retailer can get better sales through the analysis of goods combination. For example, placing beer and diapers at the same place can bring greater sales for the store. However, due to the rapid increase in the amount of data in this big data era, how to mine frequent patterns from big data has become an important issue. Many approaches were proposed to solve the incremental problem of certain data, but these approaches did not address uncertain data. The CUF-Growth algorithm preventing branches improves the performance of the traditional UF-Growth. In this paper, we propose an incremental association algorithm based on CUF-Growth to solve the problem of incremental updating of uncertain frequent items. This method retains the advantages of the original CUF-Growth, and significantly reduces the complexity of adding new transactions. The experimental results show that the proposed method reduces the execution time and perform better than the traditional UF-Growth.**

Keywords—Uncertain Data, Incremental Data Mining Algorithm; Big Data; CUF-Growth.

I. INTRODUCTION

Association rule mining is one of the most important and well researched techniques in data mining [1] and it can help humans to obtain useful and valuable information[2]. Early proposed algorithms for traditional association rules are Apriori-like algorithm[3, 4]. Apriori-like algorithm generates candidate item sets, and them compares the candidate item sets with the transactions in the database to find out all frequent itemsets, and then based on the identified frequent itemsets, useful association rules are generated. Frequent Pattern Growth (FP-Growth) algorithm[5] improves the Apriori algorithm and greatly reduce the memory space requirement. FP-Growth algorithm uses a tree structure to store the entire transaction. Therefore, FP-Growth does not need to generate candidate set and the tree data structure is effective in compressing the database. The Apriori algorithm, when dealing with huge amounts of data, has caused a sharp drop in the speed of each candidate set, and the frequent itemsets of each class must be scanned again. FP-Growth has improved the issue of candidate set. However, in some situations, data might be uncertain. In recent years, more and more uncertainty data appeared in our daily life. Many mining frequent pattern algorithms for uncertain data have been proposed. Leung et al. propose UF-Growth method [6] for uncertain data in 2008, but this method create lots of nodes in the tree construction when encountering items with different probabilities so it results in a lot of resource consumption. Leung et al. proposed a new CUF-Growth[7] algorithm in 2012 for uncertain data. This method can effectively reduce the number of nodes for items with different probabilities. This method adopts the transaction cap which is based on the highest probability of two different items in each transaction record and the tree structure is constructed using the transaction caps. Currently, to the best of our knowledge, no incremental based updating algorithms for uncertain data can be found. Therefore, in this paper we proposed an incremental association rule mining algorithm based on CUF-Growth to solve the problem of incremental updating in uncertain datasets.

II. RELATED WORK

A. Apriori algorithm

In the database, it is assumed that if the set of transaction items of the Transaction Identifier (TID) is larger than the original threshold, we will refer to the collection of items as a frequent item set. The threshold value is defined by users.

978-1-5386-3302-1/17 $31.00 © 2017 IEEE
DOI 10.1109/ACIT-CSII-BCD.2017.61

Agrawal et al. proposed Apriori algorithm[3], which candidate sets are generated and are compared with the data in the database. The followings are the steps of the Apriori algorithm:

- The first step: Scan the entire transaction database and calculate the supports of all generated candidate itemsets. Then remove the itemsets less than threshold and obtain the n-candidate frequent itemset.
- The second step: Combine the previous n-candidate frequent itemset with each other and generate n+1-candicate frequent itemset.
- Repeating first step and second step until no new n-candidate frequent itemset can be generated.

The biggest drawback of the Apriori algorithm is that it is time consuming to recursively scan the database. Assuming that the amount of data becomes larger, it takes a lot of time to read the database repeatedly, and each stage has a large number of frequent itemsets, resulting in bad efficiency.

B. *FP-growth algorithm*

Since Apriori algorithm need to rescan the database in each stage. It causes lots of I/O time consumption. Many scholars made improvements of the Apriori algorithm [8, 9], but results are still not very effectiveness.

FP-growth algorithm[10] used tree structure to improve efficiency. It is not like Apriori algorithm which large number of candidate itemsets are generated. The FP-Growth algorithm can be divided into two parts. The first part is to create the tree, and the second part is to explore. In the exploration of frequent itemsets, the FP-Tree structure is used to recursively find out all the frequent itemsets.

- Build FP-tree: The first step: Scan the database, through the item of occurrence to explore more than or less than of the threshold. Then built item occurrence counts in Head Table. The second step: Scan the database again and build FP-Tree based on the Head Table.
- Mining FP-tree: The first step: For each branch node of FP-Tree, create a Conditional Pattern Base. The second step: Base on Conditional Pattern Base create Conditional FP-Tree. The third step: The leaf node to root Bottom up mining by recursive. The fourth step: Conditional FP-Tree contains a complete association path that lists all patterns.

C. *UF-growth algorithm*

In uncertain data, each transaction contains different items and these items come with probabilities. However, same probability can appear in different items. Leung et al. propose UF-Growth method [6] in 2008. UF-Growth method address the uncertain data problem. This method is similar to the FP-Growth build tree process and can be applied to uncertain data. However, UF-growth uses tree structure to store the possibilities of different items. In UF-tree, items with same probability in different transactions can be merged into a single node. In uncertain data, the item probabilities usually are different, so items with different probabilities become different nodes.

D. *CUF-growth algotithm*

In the past, UF-Growth algorithm solve the tree growth of uncertain data, but this method produces different branches when encountering different probabilities. This will results in excess resource consumption. When large amounts of data are available, the UF-Growth constructed tree structure will become too large and the construction process is inefficient.

Leung et al. propose CUF-Growth[7] in 2012. This method can effectively reduce the number of nodes in different probability problem of uncertain data. Not only it can improve the efficiency of tree structure construction, but also it has increase the speed in the mining step. Therefore, compared to UF-Growth, CUF-Growth performs better the traditional UF-Growth. This method adopts the concept of transaction cap. The transaction cap is the highest probability of different items in the transaction record. The tree structure is constructed based on the transaction caps. The probabilities of the descendent nodes are calculated from the cumulative transaction cap probabilities when the tree structure is constructed.

E. *IFPM-BS algorithm*

Dong et al. proposed IFPM-BS(Incremental Algorithm For Frequent Pattern Mining Based On Bit-Sequence) algorithm[11] in 2011. The algorithm uses the concept of pre-frequent items in the mining association. The total transaction is divided into the upper reaches of the Support Value of 50%, while the lower bound of 30%, assuming there are total of 10 transactions, the counts of threshold ≥ 5 means frequent items, if the counts of threshold <3 means infrequent item, between 3-5counts it should be called the pre-frequent items set. IFPM-BS algorithms that the bit sequence table indicates that the bit index value in the transaction entry to check if the transaction had been purchased.

F. *RUFP algorithm*

Mundra et al. propose Rapid Update In Frequent Pattern(RUFP) algorithm[12]in 2013. This method is based on Apriori algorithm for improvement. The RUFP algorithm is divided into two parts, the first part is the traditional data format. Each transaction data records the counts of each field and gets the threshold of each item, then get the first sort of frequent items, frequent items with F (Frequent) that infrequent items are expressed in IF (Infrequent). The second stage uses the intersection to obtain frequent itemsets. In contrast, the traditional Apriori algorithm must scan the entire database before it can get the transaction, so the RUFP algorithm effectively reduces the time of each frequent project set to scan the database to achieve a more efficient algorithm.

III. PROPOSE METHOD

A. Problem definition

Previous research on the association mining algorithm actually consume a lot of resources in the database scan process. When new data are added to the original data, the tree is reconstructed once again.

When dealing with the big data, data are constantly added and modified, if the tree is constantly reconstructed in such a traditional manner, it will decrease the efficiency a lot, so we proposed a method to retain the tree structure in uncertain data. To maintain the flexibility of the original data, the proposed algorithm does not need to re-build the tree again. First, we give some definitions.

- Definition 1. Each item in a transaction has two columns <Item, Probability>. Item is the object purchased. Probability is the purchased probability.

Definition 2. Through the association rules stored in the tree structure, the root is a null node. Tid T1~T5 are transaction identifications. Items a~e are different items and the items form different orders in different transactions. Calculate the transaction cap using the probabilities in each transaction, as Table 1. The transaction cap will be used to build the uncertainty CUF-growth.

TABLE I. TRANSACTION CAP

TID	Item	Transaction cap
T1	a:0.33,d:0.12,e:0.15	0.0495
T2	b:0.23,c:0.24,d:0.18,e:0.01	0.0552
T3	b:0.26,a:0.37,d:0.16,e:0.13	0.0962
T4	b:0.29,c:0.26,d:0.14	0.0754
T5	b:0.28,a:0.24,c:0.23,e:0.11	0.0672

And then calculate the sum of the probability of occurrence of each item. Construct accumulating table for each item in descending order such as Table 2.

TABLE II. ACCUMULATING TABLE

Accumulating Table	
Item	**Accumulation probability**
b	1.06
a	0.94
c	0.73
D	0.6
E	0.4

If the order of the new items is different with the order in the accumulating table, adjust the order and the probabilities in the accumulating table. The next step is to build the CUF-tree as Figure 1.

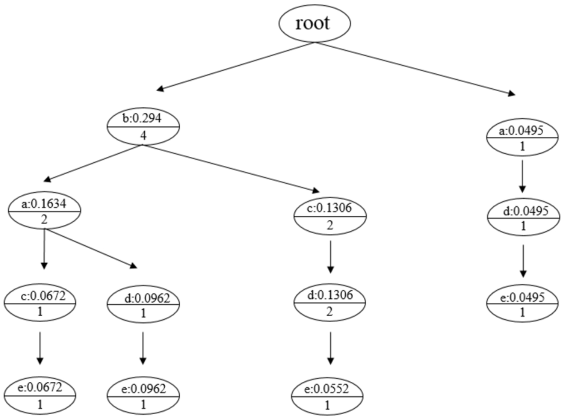

Fig. 1. CUF-Tree

B. Method description

Our proposed method can be divided into three steps, (1) database scan, (2) adding new transactions to the accumulating table and conducting probability updates, and (3) add new transactions into the original tree structure. The following describes the three steps in detail:

- The first step: Scan the new transaction items in Table 3, and then add the items into accumulating table, and then check the items sorted in Table 4.

TABLE III. NEW TRANSACTION

Transaction	Item
T1	b:0.22,a:0.21,e:0.1
T2	a:0.21,d:0.12
T3	a:0.21,c:0.2

TABLE IV. ACCUMULATING TABLE UPDATE

Update Accumulating Table	
Item	**Accumulation probability**
A	1.57
B	1.28
C	0.93
D	0.72

E	0.5

- The second step: This step can be divided into three parts, splitting, sorting and merging. New items are sorted according to the accumulating table. Traverse the parent node, and perform the splitting process without affecting the probabilities of other child nodes. Figure 2 shows how the splitting process is performed.

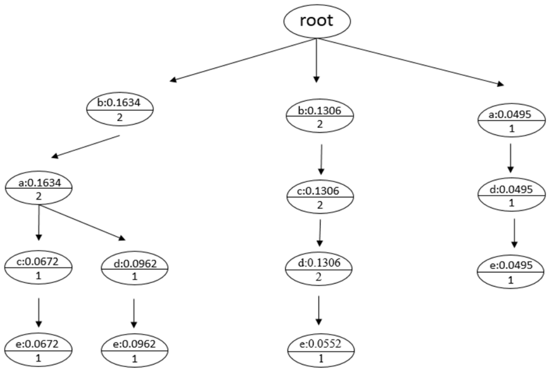

Fig. 2. Splitting

- Adjusting: New item order can be obtained from the previous step. The nodes are exchanged with each other in Figure 3.

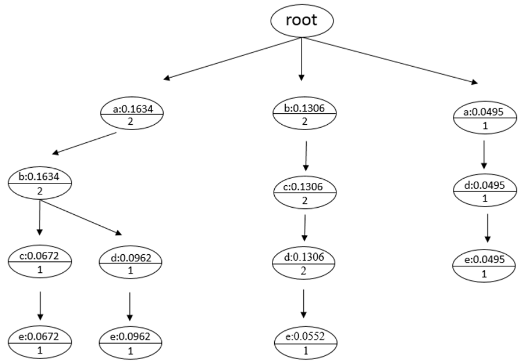

Fig. 3. Adjusting

- Merging: This step merges the items with same level of uncertainties to a single node in Figure 4.

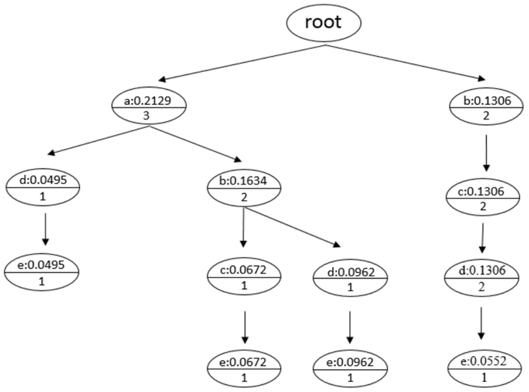

Fig. 4. Merging

C. *Tree updating*

This step will add items of the new transactions into the tree structure to achieve rapid incremental data update based on CUF-Growth in uncertain data. The items are added sequentially until the new data table is empty, as in Figure 5.

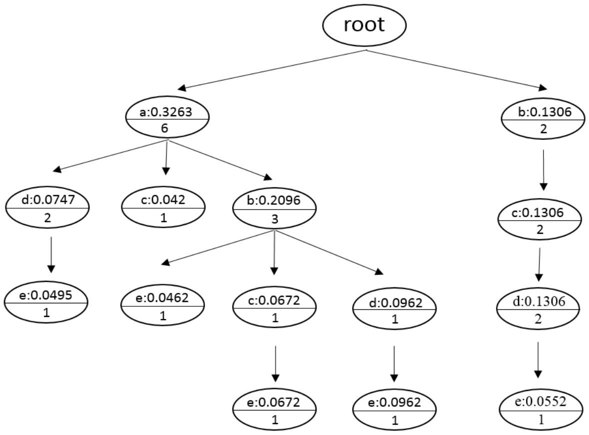

Fig. 5. Updating

In this way, the proposed method can achieve a faster update of incremental data. Unlike the traditional algorithm, the proposed algorithm does not need to rebuild the tree, so lot of data processing time is saved. The purpose algorithm can efficiently and quickly add new uncertain transaction data into the original tree because only affected nodes are considered for update. The proposed method still maintain the same tree structure as CUF-Growth.

IV. Experiment

In this section, we compared our algorithm with UF-Growth algorithm. In this study, we use the T10I4D100K dataset from IBM to run the experiment. Table 5 shows the experimental environment of this study.

TABLE V. ACCUMULATING TABLE UPDATE

CPU	Intel Core i5-4570(3.2GHz,4 cores)
Memory	8GB
HD	1TB HDD
OS	Win 7 Ultimate

A. Experiment result

The T10I4D100K dataset contains one hundred thousand transactions. The traditional UF-Growth algorithm use original database to construct a tree structure, and if additional transactions are added to the original database, a tree is reconstructed. Our method improves the reconstruction of the tree. The proposed method directly adjusts the nodes and this method effectively improves the speed the execution time. This paper split 3%, 5% and 10% data from the T10I4D100K dataset. The proposed method is compared with the traditional UF-Growth and the average execution time of 10 runs vs percentage of increment are plotted in Figure 6. In the 3% data increment, 70 ms difference can be observed, and 80 ms difference can be observed in the 5% data increment. Since UF-Growth is not an incremental approach, so reconstruction of the tree is required and lots of time is wasted. The experimental results show that our method is better than the traditional UF-Growth.

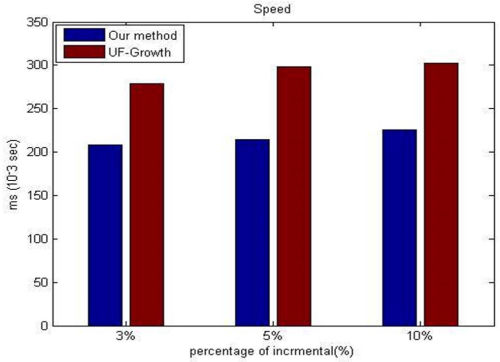

Figure 6. Experiment result

V. CONCLUSION

This paper presents an efficient algorithm for the incremental uncertain frequent pattern mining and the algorithm is efficient when new data are added. Not like the original CUF-Growth which requires reconstruction of the tree, the proposed algorithm construct the new tree structure based on the adjustment the tree nodes so the execution time is greatly reduced. The proposed method does not need to re-scan the database to build the new tree structure, so less time is used when new data are added to the database.

ACKNOWLEDGEMENT

This paper is supported by the Ministry of Science and Technology (MOST) of Taiwan. MOST provides the research funding and devices. The related project number of this work is MOST-105-2221-E-415-016.

REFERENCES

[1] E. Dumbill, "A Revolution That Will Transform How We Live, Work, and Think: An Interview with the Authors of Big Data," *Big Data,* vol. 1, pp. 73-77, 2013.

[2] L. Hetland, "Listening to music enhances spatial-temporal reasoning: Evidence for the" Mozart Effect"," *Journal of Aesthetic Education,* vol. 34, pp. 105-148, 2000.

[3] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, 1994, pp. 487-499.

[4] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Acm sigmod record*, 1993, pp. 207-216.

[5] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *ACM Sigmod Record*, 2000, pp. 1-12.

[6] C. Leung, M. Mateo, and D. Brajczuk, "A tree-based approach for frequent pattern mining from uncertain data," *Advances in Knowledge Discovery and Data Mining,* pp. 653-661, 2008.

[7] C. K.-S. Leung and S. K. Tanbeer, "Fast tree-based mining of frequent itemsets from uncertain data," in *International Conference on Database Systems for Advanced Applications*, 2012, pp. 272-287.

[8] J. S. Park, M.-S. Chen, and P. S. Yu, "Using a hash-based method with transaction trimming for mining association rules," *IEEE transactions on knowledge and data engineering,* vol. 9, pp. 813-825, 1997.

[9] H. Toivonen, "Sampling large databases for association rules," in *VLDB*, 1996, pp. 134-145.

[10] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data mining and knowledge discovery,* vol. 8, pp. 53-87, 2004.

[11] W. Dong, J. Yi, H. He, and J. Ren, "An incremental algorithm for frequent pattern mining based on bit-sequence," *I/ACT: International Journal of Advancements in Computing Technology,* vol. 3, p. 1313, 2011.

[12] A. Mundra, P. Tomar, and D. Kulhare, "Rapid Update in Frequent Pattern form Large Dynamic Database to Increase Scalability," *International Journal of Soft Computing and Engineering (IJSCE) ISSN,* pp. 2231-2307.