Review of Algorithm for Mining Frequent Patterns from Uncertain Data

Liwen Yue

University of Yanshan, College of Information Science and Engineering, Qinhuangdao, China

Summary
Mining frequent patterns from traditional database is an important research topic in data mining and researchers achieved tremendous progress in this field. However, with high volumes of uncertain data generated in distributed environments in many of biological, medical and life science application in the past ten years, researchers have proposed different solutions in extending the conventional techniques into uncertainty environment. In this paper, we review the classic mining algorithms: Apriori algorithm and FP-growth algorithm, and then analyses the improved algorithm for mining frequent patterns form uncertain data and uncertain data streams. Last, some further research directions on mining frequent patterns form uncertain data are given.

Key words:
Uncertain Data, Frequent Patterns, Data Stream, Data Mining.

1. Introduction

As one of the popular data mining tasks, frequent pattern mining [1-3] aims to discover implicit, previously unknown and potentially useful knowledge in the form of frequent patterns and so on, sets of frequently co-occurring items, objects, or events. Since its introduction, frequent pattern mining has been the subject of numerous studies, in which it has also played an important role in the mining of other patterns [6-8].

Many of the early frequent pattern mining algorithms were Apriori-based [4], which depend on a generate-and-test paradigm to mine frequent patterns from transaction databases of precise data by first generating candidates and then checking their actual support (i.e., occurrences) against the database. To improve algorithmic efficiency, tree-based frequent pattern mining algorithms [5] construct an extended prefix-tree structure (FP-tree) to capture the contents of the database and perform the mining process using a restricted test-only approach. These tree-based algorithms do not generate candidates, only test for support. With the popularization of wireless sensor networks, many real-life biological, medical or life science applications, huge volumes of data which riddled with uncertainty were collected. The presence or absence of items in a dataset in these applications is uncertain partially due to inherent measurement inaccuracies and sampling errors (e.g., in sensors or laboratory equipment), human reaction time, and intentional blurring of data to preserve anonymity (e.g., preserve patient's privacy). Hence, mining uncertain data is in demand.

The rest of paper is organized as follows. The next section introduces the frequent patterns mining algorithms from certain data briefly. In section 3 and 4, the related definition about uncertain data and improved algorithms for mining frequent patterns from uncertain data are introduced. Section 5 we check efficiency of algorithms using IBM datasets and draw the conclusion finally.

2. Frequent Pattern Mining Algorithms from Certain Data

In the past twenty years, researchers have made tremendous achievements in developing efficient and scalable algorithms for frequent patterns mining in precise and certain databases. The most classic algorithms are Apriori [1], FP-growth [2].

Apriori algorithm was proposed by Agarwal and Srikant in 1994.Apriori is bottom up approach. It is more popular algorithm to find all the frequent patterns. Apriori is designed to operate on databases containing transactions. Each transaction is seen as a set of items (a pattern). Given a threshold, the Apriori algorithm identifies the item sets which are subsets of at least C transactions in the database.

Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time, and groups of candidates are tested against the database. Firstly, algorithm generate candidate 1 length item sets C_1, compute the support of all candidate set in C_1 to determine frequent sets and save these candidate frequent 1-sets in L_1.Secondly ,it will generate candidate 2 length sets C_2 from L_1, by computing the support of candidate sets in C_2 determine the frequent 2-sets, which saved in L_2. And so on, it generates candidate item sets of length k from item sets C_{k-1} of length k-1. Then it prunes the candidates which have an infrequent sub item set. According to the downward closure lemma, the candidate set contains all frequent k- 1 length item sets. The algorithm terminates when C_k is empty and no further successful extensions are found. Algorithm scans the transaction database every time

Manuscript received June 5, 2015
Manuscript revised June 20, 2015

to determine frequent item sets among the candidates, when the database is very large, candidate sets will generate large numbers of subsets (the algorithm attempts to load up the candidate set with as many as possible before each scan). Bottom-up subset exploration (essentially a breadth-first traversal of the subset lattice) finds any maximal subset S only after all 2^{|S|}-1 of its proper subsets. Apriori has low efficiency especially when the database very big and a large number of item sets exist.
The FP-Growth Algorithm, proposed by Han in [2], is an efficient and scalable method for mining the complete set of frequent patterns by pattern fragment growth, using an extended prefix-tree structure for storing compressed and crucial information about frequent patterns named frequent-pattern tree (FP-tree).
The FP-growth algorithm works as follows: first scan the transaction database DB once, collect F, the set of frequent items and the support of each frequent item. Sort F by support descending order as FList, the list of frequent items. Second create the root of an FP-tree, T, and label it as "null". For each transaction Trans in DB do the following:
Select the frequent items in Trans and sort them according to the order of FList. Let the sorted frequent-item list in Trans be [p | P], where p is the first element and P is the remaining list. Call insert tree ([p | P], T).
The function insert tree ([p | P], T) is performed as follows. If T has a child N such that N.item-name = p.item-name, then increment N 's count by 1; else create a new node N , with its count initialized to 1, its parent link linked to T , and its node-link linked to the nodes with the same item-name via the node-link structure. If P is nonempty, call insert tree (P, N) recursively. FP-Growth Algorithm does not generate candidate patterns, only test for support, the popularity and efficiency of FP-Growth Algorithm contributes with many studies that propose variations to improve his performance.

3. Methods of Mining Frequent Patterns form Uncertain Data

3.1 Uncertain Data Mining

When handling probabilistic databases of uncertain data, the presence or absence of items in the databases is uncertain. The degree of uncertainty can be expressed in terms of existential probability, which is associated with each item in transactions in the probabilistic databases of uncertain data. The existential probability P(x, ti) of an item x in a transaction ti indicates the likelihood of x being present in ti.

Given an item x and a transaction ti, there are two possible worlds when using the "possible world" interpretation of uncertain data [9,10]: (i) a possible world W1 where x∈ti and (ii) another possible world W2 where x ∉ ti. Although it is uncertain which of these two worlds is the true world, the probability of W1 being the true world is P(x, ti) and that of W2 is 1 − P(x, ti). Then, in a probabilistic database of uncertain data with n transactions, a pattern X is frequent if it's expected support ≥ the user-defined threshold minsup. The expected support of X in the database can be computed by using the following equation:

$$\mathbf{expSup}(X) = \sum_j \left[\sup(X, W_j) * prob(W_j)\right] \quad (1)$$

where (i) sup(X,W_j) denotes the support of X in a possible world W_j which can be computed by counting the number of transactions that contain X in the possible world W_j and (ii) prob(W_j) denotes the probability of W_j to be the true world, it can be computed by following equation:

$$\prod_{i=1}^{n}\left(\prod_{x\in ti\ in\ Wj} P(x, ti) \times \prod_{y\in ti\ in\ Wf} [1 - P(y, ti)]\right) \quad (2)$$

For items x and y within X when items in X are independent, Equation (1) can be simplified [18] to become the following equation:

$$\mathbf{expSup}(X) = \sum_{i=1}^{n}\left(\prod_{x\in X} P(x, ti)\right) \quad (3)$$

When the support of items X expSup(X)>=ρs (user specified threshold), items X is frequent, or else is not.

3.2 Algorithm for Uncertain Data Mining

3.2.1 Apriori-base Algorithm

Apriori algorithm has a good application in frequent patterns mining form traditional database. Chui et al proposed U-Aprioir[9] algorithm for using in uncertain database, which is improvement of the Apriori. In Apriori algorithm, support for the candidate set is computed by accumulating. If the candidate set occurres in transaction, the support degree plus 1; but in the uncertain database, support of each item is represented by probability, from formula (3), support of each set X is by accumulating product of the probability of all x appears in the ti, which all x belong to set X.
Similar to the Apriori, method of U-Apriori is not good at large database, especially when the probability of candidate items is very small. According formula (3), accumulation of probabilities multiplication is meaningless, instead it increases execution time and system consumption. In order to solve this problem, Chui et al proposed a pruning strategy, first remove small probability events from the original database, produce a new database DT, then apply the U-Apriori algorithm in DT. The strategy mainly includes three parts: trimming module, pruning module and patch up module.

First trimming module, find all frequent items by scanning original database, set a pruning threshold ρt,(ρt is less than user specified threshold ρs).When second scan form a new database DT, some items are removed which threshold is less ρt . According to different strategies, pruning threshold ρt can be global or local, usually the most probability of pruned partition in items is its error estimation E(x). So all frequent patterns in DT were found by using U-Apriori algorithm, which must be frequent in original database. Contrary, unfrequent patterns in DT are not always unfrequent in original database. For a pattern, if its expected support expSup(X) is less than threshold ρs and its expected support add error estimation(E(X)) is greated than threshold ρs, which maybe be frequent, but if its expected support add error estimation is still less than threshold ρs, which must be unfrequent. The second step is pruning module, it will confirm error estimation based on statistical information getting form the trimming module and prune the certain unfrequent patterns. The last step check the support of frequent pattern and probable frequent in original database, confirm the accuracy.

When small probability items have large proportion(R) in original database, the size of new database DT, which generated from pruning strategy, is much smaller than the original database, this can greatly improve the efficiency. But when R is small, the size of new database and the original database is almost similar, conform the new database DT will waste time and reduce the efficiency, so the algorithm is sensitive to R. Otherwise the setting of pruning threshold ρt is important, if too high will lose frequent patterns, but too little will make the pruned items very small and new database DT big, the algorithm will not reach the result of pruning. In reality, the distribution of every item support is consistent, so the selection of threshold is very difficult.

In order to solve the above problems, Chui et al proposed an alternative method, called Decremental Pruning [11]. For candidate item, set estimated support upper bound when every transaction is processed in database, it is unfrequent when its support upper bound less than ρs, so it can be removed. One diminishing counter including the candidate items X, one of nonempty subsets X' of X, and number of transactions K is set when determined the support upper bound. When beginning, the probability of X-X' include item x is 1, it can be proved that he value of the counter is constantly greater or equal to the number of support of candidate set X, so the counter is support upper bound of the candidate item set X. Before handling, the counter is initialized X' support and in the processing the counter is decremented according to the actual support of $x \subset X\text{-}X'$, when it lower than given threshold ρs, the candidate item sets is filtered out. The number of detrimental counter is very large because the number of nonempty subsets of X (it has 2|X|-2 nonempty subsets), it made consumption of resources and low efficiency.

To improve the efficiency more methods were proposed including AS and CP, they can generate more little candidate items, but algorithms are Aprioir-based, they must process a large number of candidate items.

3.2.2 Tree-based Algorithm

Using FP-growth algorithm in mining frequent pattern from traditional database is efficient, but is not good at mining uncertain data, Leung et al proposed an improved algorithm based on FP-growth, named UF-growth [12-14], it was used in frequent patterns mining form uncertain data. Similar to FP-growth, UF-growth also has two steps :(i)construction of UF-tree;(ii)frequent pattern mining in UF-growth.

The most important is how to store the information of each item, and is conducive to later mining when the UF-tree is constructed. Therefore, each node including three parts in UFtree : (1) name; (2) the expected support; (3) and the frequency count of same expected to support. The process as follows: first scan of the database, obtain every item expected support by accumulating support degree of them, when expected support is greater than the user specified threshold ρs, the pattern is frequent, and remove unfrequent pattern, and sort frequent item by accumulating expected support on descending. The second scan, insert each transaction into UF-Tree, the insertion process similar to FP-Tree, only when all items name of a transaction and its expected support completely consist with one branch, the transaction can be merged with the branch. Using this method, the accumulated count of farther's is definitely more than the children's nodes.

When using algorithms, UF-tree is different from FP-growth in following aspects: (i)the support of X need to be recorded when construct the UF-tree of sub-database of item X;(ii) support of items bigger than X(like $X \cup \{y\}$) obtain from multiplying the support of x and support of y in this branch.

The tree is merged only when the name and corresponding support of item are completely consistent, the worst case, the number of nodes in the tree is consistent with the number of items of all transactions in the database, which will occupy lots of memory space. In order to solving the problem, Leung et al proposed two improved methods.

Support of some items may is an infinite decimal in the uncertain database, save them in the tree nodes will occupy a lot of memory space. Therefore, setting decimal digits before construction of tree, it can reduce occupation of memory space and increase the probability of the same items which have same support, at same time reduces the number of nodes in the tree and save the memory space also.

For second method, it does not construct UF-tree for item X, UF-tree only for original database and single item x, the path of x is extracted from UF-tree of itself, and then calculate the support of all the subsets belonging to this path and find frequent pattern.

4. Frequent Pattern Mining from Uncertain Data Streaming

Data stream is continuous, infinite and distribution of data usually change with time, so frequent pattern mining form data stream is big challenges. In the past few years, researches proposed a variety of algorithms for frequent patterns from data stream. For example, Giannella et al proposed FP-streaming algorithm based on static data stream. Therefore, in order to achieve frequent patterns from uncertainty data streams, it needs to put forward a new algorithm or improve existing algorithms. Two kinds of algorithms for mining frequent pattern from data streams will be introduced as follows, they are based on tree structure [15].

The first method is an improved algorithm of FP-streaming, a slightly smaller threshold than the user selected was reserved firstly, when a group of data stream reaches, calls the FP-growth method to find out the data items which support is greater than the reserve threshold, called for ”maybe frequent”. The second step, calculates the support of the “maybe frequent” patterns, and saves them in tree structure of FP-stream. Chose a preparatory threshold is necessary because the data flow is constant, at this moment the data is not frequent and may become frequent at next time. In order to prevent the frequent data was pruned early, slightly smaller threshold than the user specified threshold is set up. The difference between FP-tree and FP-stream as follow:(i) each path in FP-tree represents a transaction and in FP-stream, it represents a”maybe frequent” pattern;(ii)every nodes information include its corresponding support in FP-tree ,but node information in FP-stream include name of the item and a window table, the window table save the support of item of recent w group of data stream, when window move, the support of each node will change.

UF-stream algorithm is similar with FP-stream, “maybe frequent “ patterns were found when a batch of data stream arriving, then computed the support of these patterns, these patterns were saved in UF-stream, each node in UF-stream includes name of pattern and a window table.

UF-stream have some questions: first “maybe frequent” is not equal with frequent, it need some extra work, and some frequent pattern will be missed when set the preparatory threshold, at same time it is difficult to select the prepared threshold. Second, the item sets need to save in an extra structure (UF-stream). At last, the algorithm is immediate mining, it needs to save every data stream, some of them maybe useless.

For solving above questions, a new SUF-growth algorithm based on tree-structure is proposed. A SUF-tree is built to save the item in data stream, but the situation of nodes will change with data stream, so use a standard sort method to avoid change of situation of nodes when construct the SUF-tree. Name and support of item, and record table were included in nodes; record table saved the occurrence number of support of item in every group data stream appeared at present window, so it is easy to update information when new data stream arrived.

It can extract information from SUF-tree when it is completed and the more and more smaller database will be constructed recursively for mining frequent patterns. According to the standard sort of SUF-tree, SUF-growth algorithm adopt “bottom to up” method, it can find all frequent patterns avoid Omission or repetition. SUF-growth is delayed mining, it work when the data is needed. Once the SUF-tree is constructed, the nodes save the newest when window moved, it can effectively avoid useless data flow mining, and save a lot of resources.

Compared to UF-streaming, SUF-growth has following advantages: first it returns real frequent patterns; second the algorithm does not need extra tree-structure to save mined pattern and last it can avoid lots of waste by adopting delayed mining.

To check the algorithms, Leung et al designed contrastive experiment based on IBM datasets, FIMI datasets and UCI datasets, the conclusion is similar. We give the conclusion by emulational experiment, using IBM datasets. The datasets include 1M record, every record have 10 data items, totally have 1000 items. In experiment, set every data stream including the size of transaction is 0.1M, window size is 5, the expected support of item in every transaction is between 0 and 1. Experiment result will take the average of many times and show as following figures.

In figure 1, we checked the effect of threshold, with the threshold increasing, support of item larger than it reduce, so frequent patterns and corresponding run time reduce also. Extendibility of algorithms also was checked by increasing the number of transaction.

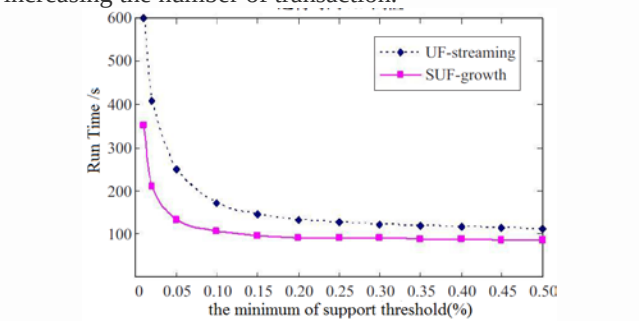

Fig. 1 Effect of Change of Threshold

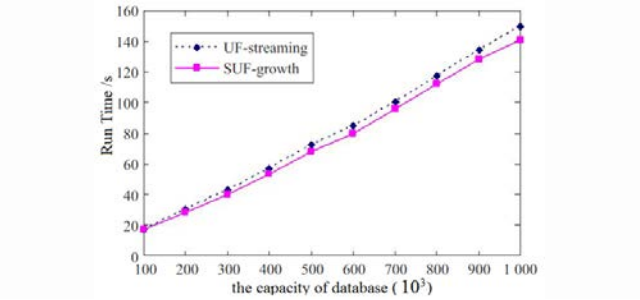

Fig. 2 Effect of Capacity of Database

From the figure 2, the running time of the algorithm with incensement of number of transactions has a linear growth. Leung et al also compare the two algorithms from the distribution of expected support, nodes number in tree and accuracy or efficiency of algorithm. The experimental results show that two methods have their own advantages and disadvantages, can find frequent patterns from the uncertain data stream, SUF-growth is better than UF-streaming in running time and high accuracy.

5. Conclusion

In reality，there are an amount of uncertain data, how to find valuable data and mine frequent pattern from these data is a big question, it need efficient algorithm. In this paper introduced some algorithms for frequent pattern mining, all of them were based on U-Aprior or tree structure UF-growth or improvement of them. Especially, we introduced the US-streaming algorithm and SUF-growth algorithm which were used of mining frequent pattern form uncertain data. The efficiency of US-streaming and SUF-growth algorithm was checked by setting the Minimum support threshold and size of database respectively. Experiment result shows that the improved algorithms for uncertain data have good efficiency in reducing memory and run time, especially used in complete frequent patterns. But the other type of frequent patterns include frequent patterns based on constraint, maximal frequent patterns, and frequent closed patterns so on， every type need their own suitable algorithm, these algorithms are one of directions of next research work. The efficiency and the application area in reality of algorithm is problem to be solved.

Acknowledgments

This paper is partially supported by Science and technology Instruction Program of Qinhuangdao, No：201302A029.

References

[1] R. Agrawal, T. Imielinski & A. Swami, Mining association rules between sets of items in large databases, in ACM SIGMOD 1993, pp. 207–216.
[2] B. Chikhaoui, S. Wang, & H. Pigot, A frequent pattern mining approach for ADLs ecognition in smart environments, in IEEE AINA 2011, pp. 248–255.
[3] C.K.-S. Leung & C.L. Carmichael, Exploring social networks: A frequent pattern visualization approach, in IEEE SocialCom 2010, pp. 419–424
[4] R. Agrawal & R. Srikant, Fast algorithms for mining association rules, in VLDB 1994, pp. 487–499.
[5] J. Han, J. Pei & Y. Yin, Mining frequent patterns without candidate generation, in ACM SIGMOD 2000, pp. 1–12
[6] J.J. Cameron, C.K.-S. Leung & S.K. Tanbeer, Finding strong groups of friends among friends in social networks, in IEEE DASC/SCA 2011,pp. 824–831
[7] Y. Chen, A. Narayanan, S. Pang & B. Tao, Malicioius software detection using multiple sequence alignment and data mining, in IEEE AINA 2012, pp. 8–14.
[8] H. Takei & H. Yamana, IC-BIDE: intensity constraint-based closed sequential pattern mining for coding pattern extraction, in IEEE AINA 2013, pp. 976–983.
[9] C.-K. Chui, B. Kao & E. Hung, Mining frequent itemsets from uncertain data, in PAKDD 2007, pp. 47–58
[10] C.K.-S. Leung & B. Hao, Mining of frequent itemsets from streams of uncertain data, in IEEE ICDE 2009, pp. 1663–1670.
[11] Chui C K，Kao B.A decremental approach for mining frequent itemsets from uncertain data[C] LNAI 5012：PAKDD，2008：64-75.
[12] Leung C K S，Mateo M A F，Brajczuk D A.A tree-based approach for frequent pattern mining from uncertain data[C] LNAI 5012：PAKDD，2008：653-661.
[13] Leung C K S，Carmichael C L，Hao B.Efficient mining of frequent patterns from uncertain data[C] Proc IEEE ICDM Workshops,2007：489-494.
[14] Leung C K S，Brajczuk A D.Efficient mining of frequent itemsets from data streams[C] LNCS 5071：BNCOD，2008：2-14.
[15] Leung C K S，Hao B.Mining of frequent items from streams of uncertain data[C] Proc IEEE Computer Society，2009：1663-1670

Liwen Yue received the B.S. degrees in Computer Science from Jilin Normal Univ in 2000 and M.S. degree in Computer Application from Yanshan Unvi in 2007. During 2007-now, she words as a teacher in Yanshan Univ. Her main research interest includes XML Data Model, Data Mining and their applications.