ISSN 2395-1621

Incremental Mining Algorithm For Efficient Mining Of Frequent Item Sets On Large Uncertain Databases

[#1]Rahul Arun Misal, [#2]Sadashiv Prabhakar Shinde

[1]rahulmisal26@gmail.com
[2]sada.shinde83@gmail.com

[#1]B.E. Student,
[#2]Ass.Professor(Guide)

Rajiv Gandhi College of Engineering, A'nagar, Maharashtra.

ABSTRACT

The data handled in emerging applications like location-based services, sensor monitoring systems, and data integration, are often inexact in nature. In this paper, we study the important problem of extracting frequent itemsets from a large uncertain database, interpreted under the Possible World Semantics. This issue is technically challenging, since an uncertain database contains an exponential number of possible worlds. By observing that the mining process can be modelled as a Poisson binomial distribution, we develop an approximate algorithm, which can efficiently and accurately discover frequent itemsets in a large uncertain database. We also study the important issue of maintaining the mining result for a database that is evolving (e.g., by inserting a tuple). Specifically, we propose incremental mining algorithms, which enable probabilistic frequent itemset results to be refreshed. This reduces the need of re-executing the whole mining algorithm on the new database, which is often more expensive and unnecessary. We examine how an existing algorithm that extracts exact itemsets, as well as our approximate algorithm, can support incremental mining. All our approaches support both tuple and attribute uncertainty, which are two common uncertain database models. We also perform extensive evaluation on real and synthetic datasets to validate our approaches.

General Terms: Frequent itemsets, uncertain dataset, approximate algorithm, incremental mining.
Keywords: PFI, PWS, S-PMF, CDF.

ARTICLE INFO

Article History
Received: 21st March 2018
Received in revised form :
21st March 2018
Accepted: 23rd March 2018
Published online :
24th March 2018

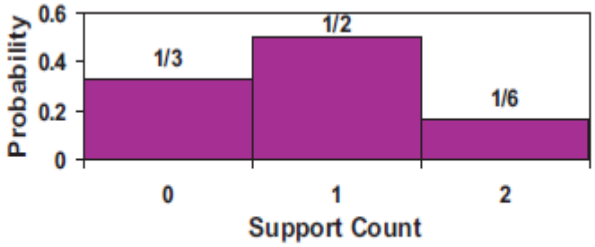
I. INTRODUCTION

The databases used in many important and novel applications are often uncertain. For example, the locations of users obtained through RFID and GPS systems are not precise due to measurement errors. As another example, data collected from sensors in habitat monitoring systems (e.g., temperature and humidity) are noisy. Customer purchase behaviours, as captured in supermarket basket databases, contain statistical information for predicting what a customer will buy in the future [3], [6]. Integration and record linkage tools also associate confidence values to the output tuples according to the quality of matching. In structured information extractors, confidence values are appended to rules for extracting patterns from unstructured data. To meet the increasing application needs of handling a large amount of uncertain data, *uncertain databases* have been recently developed [10]. Figure 1 shows an online marketplace application, which carries probabilistic information. Particularly, the purchase behaviour details of customers Jack and Mary are recorded. The value associated with each item represents the chance that a customer may buy that item in the near future. These probability values may be obtained by analyzing the users' browsing histories. For instance, if Jack visited the marketplace ten times in the previous week, out of which *video* products were clicked five times, the marketplace may conclude that Jack has a 50% chance of buying *videos*. This *attribute-uncertainty* model, which is well-studied in the literature [6], [10] associates confidence values with data attributes. It is also used to model location and sensor uncertainty in GPS and RFID systems.

Customer	Purchase Items
Jack	(*video*:1/2), (*food*:1)
Mary	(*clothing*:1), (*video*:1/3); (*book*:2/3)

Illustrating an uncertain database.

The frequent itemsets discovered from uncertain data are naturally probabilistic, in order to reflect the confidence placed on the mining results. Figure 2 shows a *Probabilistic Frequent Itemset* (or *PFI*) extracted from Figure 1.



s-pmf of PFI {*video*} from Figure 1.

A *PFI* is a set of attribute values that occurs frequently with a sufficiently-high probability. In Figure 2, the *support probability mass function* (or *s-pmf* in short) for the *PFI {video}* is shown. This is the pmf for the number of tuples (or *support count*) that contain an itemset. Under PWS, a database induces a set of possible worlds, each giving a (different) support count for a given itemset. Hence, the support of a frequent itemset is described by a pmf. In Figure 2, if we consider all possible worlds where itemset *{video}* occurs twice, the corresponding probability is 1/6 . A simple way of finding PFIs is to mine frequent patterns from every possible world, and then record the probabilities of the occurrences of these patterns. This is impractical, due to the exponential number of possible worlds. To remedy this, some algorithms have been recently developed to successfully retrieve PFIs without instantiating all possible worlds [6]. These algorithms can verify whether an itemset is a PFI in $O(n2)$ time (where n is the number of tuples contained in the database). However, our experimental results reveal that they can require a long time to complete (e.g., with a 300k real dataset, the dynamic programming algorithm in [6] needs 30.1 hours to find all PFIs). We observe that the s-pmf of a PFI can be captured by a Poisson binomial distribution, for both attribute- and tuple-uncertain data. We make use of this intuition to propose a method for approximating a PFI's pmf with a Poisson distribution, which can be efficiently and accurately estimated. This *model-based algorithm* can verify a PFI in $O(n)$ time, and is thus more suitable for large databases. We demonstrate how our algorithm can be used to mine *threshold-based* PFIs, whose probabilities of being true frequent itemsets are larger than some user-defined threshold [6]. Our algorithm only needs 9.2 seconds to find all PFIs , which is four orders of magnitudes faster than the method in [6].

II. RELATED WORK

Mining frequent itemsets is an important problem in data mining, and is also the first step of deriving association rules [4]. Hence, many efficient itemset mining algorithms (e.g., Apriori [4] and FP-growth) have been proposed. While these algorithms work well for databases with precise values, it is not clear how they can be used to mine probabilistic data. Here we develop algorithms for extracting frequent itemsets from uncertain databases. Although our algorithms are developed based on the Apriori framework, they can be considered for supporting other algorithms (e.g., FP-growth) for handling uncertain data. For uncertain databases, [2] developed efficient frequent pattern mining algorithms based on the expected support counts of the patterns. However, [6] found that the use of expected support may render important patterns missing. Hence, they proposed to compute the probability that a pattern is frequent, and introduced the notion of PFI. In [6], dynamic-programming-based solutions were developed to retrieve PFIs from attribute uncertain databases. However, their algorithms compute exact probabilities, and verify that an itemset is a PFI in $O(n2)$ time. Our model-based algorithms avoid the use of dynamic programming, and are able to verify a PFI much faster (in $O(n)$ time). Approximate algorithms for deriving threshold-based PFIs from tuple- uncertain data streams were developed. While only considered the extraction of singletons (i.e., sets of single items), our solution discovers patterns with more than one item. Recently, developed an exact threshold-based PFI mining algorithm. However, it does not support attribute-uncertain data considered in this paper. In a preliminary version of this paper, we examined a model-based approach for mining PFIs. Here, we study how this algorithm can be extended to support the mining of evolving data. Other works on the retrieval of frequent patterns from imprecise data include: [9], which studied approximate frequent patterns on noisy data, which examined association rules on fuzzy sets; and, which proposed the notion of a "vague association rule". However, none of these solutions are developed on the uncertainty models studied here. For evolving databases, a few incremental mining algorithms that work for exact data have been developed. For example, in , the Fast Update algorithm (FUP) was proposed to efficiently maintain frequent itemsets, for a database to which new tuples are inserted. Our incremental mining framework is inspired by FUP. The FUP2 algorithm was developed to handle both addition and deletion of tuples. ZIGZAG [1] also examines the efficient maintenance of maximal frequent itemsets for databases that are constantly changing. A data structure, called CATS Tree, was introduced to maintain frequent itemsets in evolving databases. Another structure, called CanTree, arranges tree nodes in an order that is not affected by changes in item frequency. The data structure is used to support mining on a changing database. To our best knowledge, maintaining frequent itemsets in evolving uncertain databases has not been examined before. We propose novel incremental mining algorithms for both exact and approximate PFI discovery. Our algorithms can also support attribute and tuple uncertainty models. Table 1 summarizes the major work done in PFI mining. Here, "Static Algorithms" refer to algorithms that do not handle database changes. Hence, any change in the database necessitates a complete execution of these algorithms.

TABLE 1
Our Contributions (marked [√]).

Uncertainty Model	Static Algorithms	Incremental Algorithms
Attribute	Exact [6] Approx. [√]	Exact [√] Approx. [√]
Tuple	Exact [30] Approx. (singleton) [35] Approx. (multiple items) [√]	Exact [√] Approx. [√]

III. PROPOSED SYSTEM

A Model-based algorithm is proposed for mining manifold itemset from large ambiguous dataset illustrated under possible world semantics . An approximated algorithm is established to ascertain frequent itemset from large ambiguous database. An Incremental mining algorithm is adduced to maintain the result of mining algorithm. It supports both tuple and attribute ambiguity. Validated by interpreting both real and synthetic dataset.

- Frequent itemset is ascertained efficiently and accurately.
- O(n) time is needed to authenticate an itemset as PFI
- Low computational cost.
- High performance in detecting PFI.
- Support incremental mining by refreshing the mining result than reevaluating whole algorithm.
- Both tuple and attribute uncertainty is supported.

INCREMENTAL MINING

Now the method of how to efficiently maintain a set of PFIs in an evolving database is presented here, where new tuples, or transactions, are constantly appended to it. We assume that every tuple has a timestamp attribute, which indicates the time that it is created. This timestamp is not used for mining; it is only used to differentiate new tuples from existing ones. Let D be the "old" database that contains n tuples, and d be a delta database of nI tuples, whose timestamps are larger than those of tuples in D. Let D+ be a "new" database, which is a concatenation of the tuples in D and d, and has a size of N+ = n + nI. Given the set of PFIs and their s− pmfs in D, the goal is to discover PFIs on D+, under the same *minsup* and *minprob* values used to mine the PFIs of D. The algorithm uses sDB(I) and PrDBfreq(I) to respectively denote the support count and the frequentness probability of item set I in some database DB, where DB is in{D, d, D+ }. The mining problem described above can be treated as a special case of *stream mining*, which refers to the maintenance of mining results for stream data. Particularly, the database d as the arrival of |d| data units from a stream source is assumed. Moreover, assume that the sliding window initially contains D, which then expands to incorporate new stream units. Mining D+ is then equivalent to updating the mining results for the arrival of |d| stream units. In Section 8.3, an adaptation of a stream algorithm in [35] for use in mining is observed. A simple way of obtaining PFIs from D+ is to simply rerun a PFI− mining algorithm on it. However, this approach is not very economical, since 1) running a PFI algorithm on a large database is not trivial; and 2) the same algorithm has to be frequently executed if a lot of update activities occur. In fact, if only a few tuples in d are appended to D, it may not be necessary to compute all PFIs on D+ from scratch. This is because the PFIs found in D+ should not be very different from those discovered in D. Based on this intuition, a mining algorithm that finds PFIs in D+, without rerunning a complete PFI algorithm was designed. This algorithm works the best when the size of d is very small compared with that of D; nevertheless, it works with any size of d. The framework of the solution, discovers PFIs in D+, based on the PFIs found in D. this solution is extended to discover PFIs.

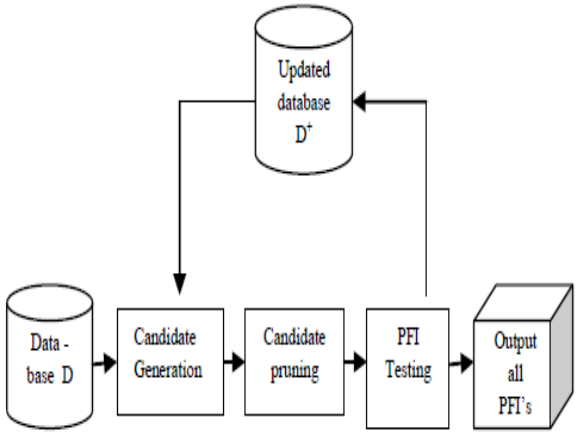


Fig.1: PFI mining from the uncertain database

Algorithm:

Algorithm for PFI mining

Input: Data base D, updated delta database d, minsup, minprob .
Output: PFI"s of D^+, $F_k^+ = \{ F_1^+ , F_2^+ , \ldots , F_n^+ \}$ // F_k^+ is the set of PFI"s .
Method:

```
if (deltaDb is empty) then
{
    C(1).Generate( oldDb);
     k=1;
    while(C(k)!=0) do
    {
        C(k).prune(oldDb, F, μm);
        if(C(k)!=0) then
            F =C(k).Test( oldDb, d, F, μm);
        else
            break;
        C(k+1).Generate( oldDb ) ;
        k=k+1;
    }
}
else  //deltaDb is not empty
```



```
{
    C(1).Generate(deltaDb);
    k=1;
    while(C(k)!=0)
    {
        C(k).prune( deltaDb, F, μ^L_m`);
            F=C(k).Test(oldDb, d, F,μ^L_m`);
        else
            break;
        C(k+1).Generate(d);
        k=k+1;
    }
}
return F_k^+ = {F_1^+, F_2^+, ..., F_n^+}
```

IV. RESULTS

Now the experimental results on the data set, called accidents, comes from the Frequent Item set Mining (FIMI) Data Set Repository. This data set is obtained from the National Institute of Statistics (NIS) for the region of Flanders (Belgium), for the period of 1991-2000. The data are obtained from the "Belgian Analysis Form for Traffic Accidents," which are filled out by a police officer for each traffic accident occurring on a public road in Belgium. The data set contains 3,40,184 accident records, with a total of 572 attribute values. On average, each record has 45 attributes. The algorithm uses the first 10k tuples as the default data set as its input. The default value of minsup is 20 percent. To test the mining algorithm, it uses the first 10k tuples as the old database D, and the subsequent tuples as the delta database d. The default size of d is 5 percent of D. , it considers both attribute and tuple uncertainty models. For attribute uncertainty, the existential probability of each attribute is drawn from a Gaussian distribution with mean 0.5 and standard deviation 0.125. This same distribution is also used to characterize the existential probability of each tuple, for the tuple uncertainty model. The default value of minprob is 0.4. In the results presented, minsup is shown as a percentage of the data set size n. Notice that when the values of minsup or minprob are large, no PFIs can be returned; it do not show the results for these values. The experiments were carried out on the Windows XP operating system, on a machine with a 2.66 GHz Intel Core 2 Duo processor and 2 GB memory. The programs were written in Java and compiled with J2SE runtime environment 1.6.0. The proposed algorithm is implemented under the java runtime environment. The algorithm extracts the probabilistic frequent item sets within a fraction of seconds.Now the comparison of the performance of the PFI mining algorithms mentioned in this paper:1) APM, the Apriori algorithm used in [6] that employs the PFI testing method and 2) PA (Proposed algorithm), the proposed algorithm that uses the improved version of the PFI testing method is discussed here.

Since APM approximates s-pmf by a Poisson distribution, first examine that its accuracy with respect to AP, which yields PFIs based on exact frequentness probabilities. Here, the standard recall and precision measures [7] is used, which quantify the number of negatives and false positives. Specifically, let $FAPM$ be the set of PFIs generated by APM, and FPA be the set of PFIs produced by PA. To compare the existing algorithm with the proposed algorithm, The recall and precision were used. Both recall and precision have values between 0 and 1. The recall and the precision of APM, relative to PA, are defined as

$$recall = \frac{|F_{APM} \cap F_{AP}|}{F_{APM}}$$

$$precision = \frac{|F_{APM} \cap F_{AP}|}{F_{AP}}$$

In these formulas, Also, a higher value reflects a better accuracy. Table 2 shows the recall and the precision of APM (Apriori algorithm), for a wide range of minsup, n, and minprob values. As it can observed that the precision and recall values are always higher than 98 percent. Since The proposed algorithm PA returns the same PFIs as APM, it is also highly accurate result. Both precision and recall are used to compare the performance of mining algorithms.

Table 2. Recall and Precision of APM

minsup	0.1	0.2	0.3	0.4	0.5
Recall	1	1	1	1	1
Precision	0.997	1	1	1	1

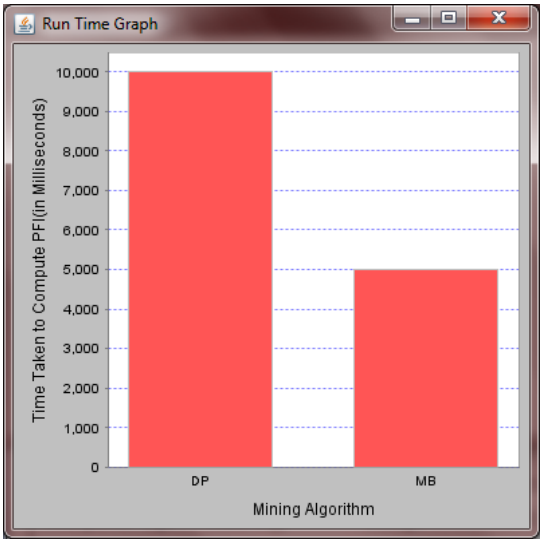
(a) Recall and Precision vs. minsup

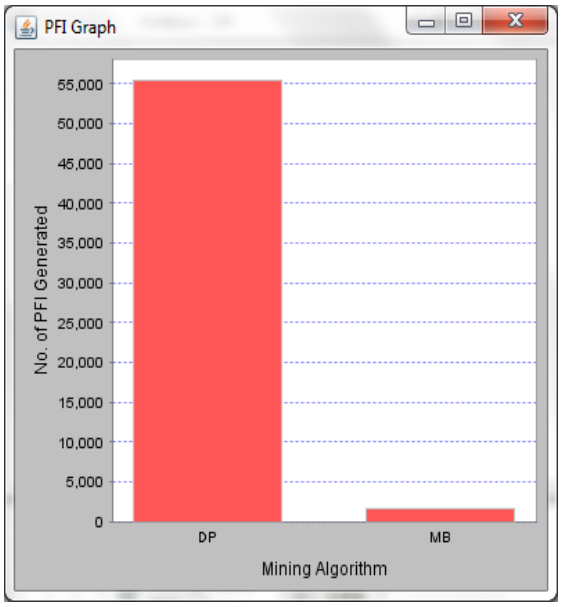
minprob	0.1	0.3	0.5	0.7	0.9
Recall	1	1	1	1	1
Precision	0.986	1	0.985	1	1

(b) Recall and Precision vs. minprob

n	1k	4k	10k	50k	100k
Recall	1	1	1	1	1
Precision	0.987	0.988	1	1	1

(c) Recall and Precision vs. n





V. FUTURE ENHANCEMENT

Rule Mining Algorithm can be used to extract frequent item set from large uncertain database. This mining model solves the update and deletes operations based itemset mining problems. The probability neighborhood is used for the frequent itemset selection process. Dynamic database update operations are managed with rule refreshment process. The system uses the dynamic threshold values for candidate set pruning process.

VI. CONCLUSION

In this paper, we propose a approach to extract threshold-based PFIs from large uncertain databases. Its main idea is to the s-pmf of a PFI by some common probability model, so that a PFI can be verified quickly. This approach supports retrieving PFIs from evolving databases. The experimental results show that the proposed algorithm is highly efficient and accurate. It supports both attribute- and tuple uncertain data approach to develop other mining algorithm (e.g., clustering and classification) on uncertain data. It is also interesting to study efficient mining algorithm for handling tuple. Another interesting work is to investigate PFI mining algorithm for probability models that capture correlation among attributes and tuples.

REFERENCES

[1] A. Veloso, W. Meira Jr., M. de Carvalho, B. Poˆ ssas, S. Parthasarathy, and M.J. Zaki, “Mining Frequent Itemsets in Evolving Databases,” Proc. Second SIAM Int‟l Conf. Data Mining (SDM), 2002.

[2] C. Aggarwal, Y. Li, J. Wang, and J. Wang, “Frequent Pattern Mining with Uncertain Data, “, Proc. 15th ACM SIGKDD Int‟l Conf. Knowledge Discovery and Data Mining (KDD), 2009.

[3] C. Aggarwal and P. Yu, “A Survey of Uncertain Data Algorithm and Applications,” IEEE Trans Knowledge and Data Eng., vol. 21, no. 5, pp. 609-623, May 2009.

[4] R. Agrawal, T. Imieliμnski, and A. Swami, “Mining Association Rules between Sets of Items in Large Databases,” Proc.ACM SIGMOD Int‟l Conf. Management of Data, 1993.

[5] O. Benjelloun, A.D. Sarma, A. Halevy, and J. Widom, “ULDBs: Databases with Uncertainty and Lineage,” Proc. 32ndInt‟l Conf. Very Large Data Bases (VLDB), 2006.

[6] T. Bernecker, H. Kriegel, M. Renz, F. Verhein, and A. Zuefle,“Probabilistic Frequent Itemset Mining in Uncertain Databases,” Proc. 15th ACM SIGKDD Int‟lConf. Knowledge Discovery and Data Mining (KDD), 2009.

[7] C.J. van Rijsbergen, Information Retrieval. Butterworth, 1979.

[8] L.L. Cam, “An Approximation Theorem for the Poisson Binomial Distribution,” Pacific J. Math., vol. 10, pp. 1181-1197,1960.

[9] H. Cheng, P. Yu, and J. Han, “ Frequent Itemset Mining in the Presence of Random Noise,” Proc. Soft Computing for Knowledge Discovery and Data Mining, pp. 363-389, 2008.

[10] R. Cheng, D. Kalashnikov, and S. Prabhakar, “Evaluating Probabilistic Queries over Imprecise Data,” Proc. ACM SIGMOD Int‟l Conf. Management of Data, 2003.

