IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 2, April 2014.

www.ijiset.com

ISSN 2348 - 7968

MINING FREQUENT PATTERN USING PROBABILITY BASED INCREMENTAL DATABASE DISCOVERY

Shaliniselvi

ME Software Engineering in College Of Engineering Guindy, Anna University, Chennai.

Abstract – There has been a recent surge in work with mining Frequent Pattern from the Dynamic Database. In Dynamic Database where new transaction are inserted frequently. Mining Frequent pattern from the dynamic database requires rescanning the Database which increases the transaction cost and its time. This paper is concerned with extracting useful information from Dynamic Database with the help of probability based Incremental Database Discovery. This paper uses the probability to find out expected frequent Itemsets which reduces the time to scan the Original Database. This paper discovers the frequent patterns and association rulesfrom probabilistic data. This is technically challenging, since a probabilistic database can have an exponential number of possible worlds. Apriori Algorithm finds Support Probability mass function which discover frequent patterns in bottom-upmanner without expand the database into exponential number of possible worlds . These algorithm which inherit the frequent pattern using approach namely Divide And Conquer Approach partition the Database. Thus the extracted frequent pattern are used in the generation of probability association rules.

Index Terms – Frequent Item sets, Uncertain dataset, Association Rules.

1.INTRODUCTION:

The mining of association rules on transactional database is usually an offline process since it is costly to find the association rules in large databases.Incremental association rule discovery is one ofthose issues which maintain rules when new transactions are appended to an original database. In fact, data hasgrown rapidly; thus, when a new set of transactions,called increment database, are inserted into the originaldatabase, some rules from the previous mining may beinvalid.With usual market-basket applications, new transactions are generated and old transactions may be obsolete as time advances. As a result, incremental updating techniques should be developed for maintenance of the discovered association rules to avoid redoing mining on the whole updated database.

A Database may allow frequent updates which changes the database with new Information which make the existing Association rule invalid and also new rules has been generated. This became tedious task in Larger Database. The Simple method for handling this problem is to rescan the Database with Apriori. These algorithm cannot be applied directly without taking the incremental characteristics into consideration. However these algorithm is time consuming and inefficiency. To handle the Incremental Mining Effectively the frequent Itemsets are mined with the Probability gurantees. Several attempts has

IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 2, April 2014.
www.ijiset.com
ISSN 2348 - 7968

been made to predict the probability using Bernoulli trials and the normal approximation. But numerical difficulty incomputing probabilities occurs when the algorithm dealswith a large database. To manipulate with this problem,the improved probability-based incremental associationrule discovery using Support count and the Expected mean prediction to estimate
the probability of occurrence of expected frequent itemset

1.1 OVERVIEW:

Table 1.1 shows the illustrative Transactional Database which increments iteratively. Let D be the Original database which contains the list of transactional items. As the time advances Δ- be the set of items removed from the database and Δ+ be the new items added to the Database. The above addition and deletion in a dynamic database where new transactions are inserted frequently,association rules discovered in the previous database possibly no longer valid andinteresting rules in the updated database. As a result, new business information suchas changing customer preferences or new seasonal trends may not be discovered. Tocreate an intelligent environment such that new business information can bediscovered in a dynamic database, association rules algorithms should be capable ofmining a dynamic database incrementally.

A basic and simple method for solving this problem isto rescan entire databases with Apriori algorithm toget new itemsets. However, this method is timeconsuming and inefficiency. By reducing a number oftimes to scan databases, several algorithms are proposedsuch as Sliding Windows Filtering (SWF), Negative Border (NBd), probability-based incremental association rule discovery and so on.

Table1.1: Illustrative Transactional Database Incremental Mining Example

D	Δ-	T_1	A B C D E F	
		T_2	A F	
		T_3	A B C E	
	D-	T_4	A B D F	D+
		T_5	C E	
		T_6	B D E F	
		T_7	A D F	
		T_8	A B C	
	Δ+	T_9	C D E	
		T_{10}	A F	
		T_{11}	A B C	

For the probability-based incremental association rulediscovery algorithm, it needs only one time to scan thewhole original database and works by using the principlesof of Bernoulli trials to predict the expected frequentitemsets, i.e., the infrequent itemset which can possibly bea frequent itemset. However, numerical difficulty incomputing probabilities occurs when the algorithm dealswith a large database. To manipulate with this problem,the improved probability-based incremental associationrule discovery using normal approximation to estimatethe probability of occurrence of expected frequent itemset.

A **probabilistic database** is an database in which the possible worlds have associated probabilities. In a probabilistic database, each data item - relation, tuple and value that an attribute can take - is associated with a probability ∈ (0,1], with 0 representing that the data is certainly incorrect, and 1 representing that it is certainly correct. There are essentially two kinds of uncertainties that could exist in a probabilistic database namely Tuple-level uncertainty and Attribute-level uncertainty. Due to its simplicity in database design and query semantics, the *tuple-uncertainty* model is commonly used in probabilistic databases. Conceptually, each tuple carries an *existential probability* attribute, which denotes the confidence that the tuple exists.

To interpret tuple uncertainty, the *Possible World Semantics* (or PWS in short) is often used. Conceptually, a database is viewed as a set of deterministic instances (called *possible worlds*), each of which contains a set of zero or more tuples. A possible world for Figure 1 consists of the tuples (*t*2,*t*3,*t*5) existing with a probability of (1 – 0.1) *1.0 * 0.5 *(1 -0.2) *1:0 = 0.036. Any query evaluation algorithm for a probabilistic database has to be correct under PWS. That is, the results produced by the algorithm should be the same as if the query is evaluated on every possible world.

Although PWS is intuitive and useful, evaluating queries under this notion is costly. This is because a probabilistic database has an exponential number of possible worlds. For example, the table in Figure 1 has $2^3 = 8$ possible worlds. Performing query evaluation or data mining under PWS can thus be technically challenging. In fact, the mining of uncertain or probabilistic data has recently attracted research attention. In efficient clustering algorithms were developed to group uncertain objects that are close to each other.

The frequent item sets discovered from uncertain dataare naturally probabilistic, in order to reflect the confidence placed on the mining results. Fig. 2 shows a Probabilistic Frequent Item set (PFI) extracted from Fig. 1. A PFI is a set of attribute values that occurs frequently with a sufficiently high probability. In Fig. 2, the support probability mass function (s-pmf) for the PFI {location=x} is shown. This is the pmf for the number of tuples (or support count) that contain an item set. Under PWS, a database induces a set of possible worlds, each giving a (different) support count for a given item set. Hence, the support of a frequent item set is described by a pmf. In Fig. 2, if we consider all possible worlds where item set {location=x} occurs twice, the corresponding probability is 0.44.

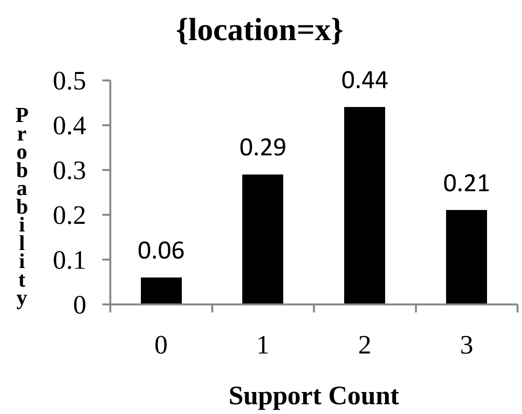


Fig 1.1: Sample Probability Frequent Pattern derived from Fig 1

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, proposed algorithms have been recently developed to successfully retrieve PFIs without instantiating all possible worlds. The proposed System present simple and effective methods to prune infrequent patterns and adopt divide and conquer (DC) to compute the support pmf of a pattern. This method, also used by to derive probability Frequent Patterns for other probabilistic models, has a complexity of achieves $O(n \log^2 n)$ time. Based on these methods, the *p-Apriori*algorithm has been developed to retrieve probability Frequent Patterns in a *bottom-up* manner. And also extend p-Apriori to generate rule from frequent pattern.

In exact databases, deriving association rules from frequent patterns is not difficult. Given two frequent patterns X and XY , the confidence of $X \Rightarrow Y$ can be calculated with an arithmetic division on their supports. This is no longer true for probabilistic data. Here, the support of X and XY become *correlated* random variables. It is not clear how to define and compute the confidence of $X \Rightarrow Y$ .Hence the concept of probability Association Rule has been proposed which naturally extends the association rule semantics.

To summarize we develop the algorithm which efficiently identifies the frequent pattern in the Dynamic Data Mining.And also effective rule generation for probabilistic database.

The rest of the paper is organized as follows: in Section 2, we review the related works. Section 3 introduces the notions of p-FPs and p-ARs. Sections 4 and 5 present two algorithms for mining p-FPs for the Dynamic Data Mining . In Section 6, we develop an algorithm for generating p-ARs. Section 7 presents our experimental results. We conclude in Section 8.

2 RELATED WORK

Mining frequent item sets is an important problem in data mining, and is also the first step of deriving association rules. Hence, many efficient item set mining algorithms(e.g., Apriori 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 item sets 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.

The probabilistic database paradigm was proposed early in 1980s. In probabilistic databases, uncertainty is treated as first-class citizen, where probabilities are stored along with data records to reflect the uncertainty. A probabilistic database can be viewed as a succinct summary of a set of possible database instances, or possible worlds. Two types of models, namely, attribute- and tuple-level uncertainty, were used to represent uncertain data. The attribute-uncertainty models represent the impreciseness of data values, in which an attribute can take a range of values governed by a probability distribution, rather than a fixed one. The tuple uncertainty, on the other hand, doubts the existence of data records, where each tuple is annotated by a probability that it really belongs the database. Uncertainty models also vary in aspects of independence assumptions. The existence of tuples is assumed as independent with each other in [11]. The x-tuple model [8] additionally considers the mutual-exclusiveness of tuples, and it was generalized by the And-Xor tree model [15] which can represent the relationship of mutual existence and mutual exclusivity in a hierarchy. Arbitrary tuple correlations are considered. Query processing techniques on uncertain data have been extensively studied [7,12].

The problem of discovering frequent patterns and association rules was first proposed in [9, 8]. Over more than a decade, there is a large research body on this topic with various emphasis. Since this task naturally incurs high computational costs, many works devoted to developing efficient algorithms for it. For example, the Apriori algorithm, Partition , FP-tree, and Tree Projection [3]. Due to the anti-monotonicity property of frequent patterns, there is a great interest in developing efficient algorithms to discover only maximal frequent patterns, which is a small fraction of frequent patterns but can represent the complete pattern set. Notable algorithms include MAX-MINER [11], Depth-Project [3], MAFIA [14], GenMAX. The notion of closed frequent patterns was also proposed in [16], which not only represent the whole frequent pattern set but also preserve their exact support counts, and some efficient algorithms are developed, e.g., CLOSET and CHARM. Alternative interestingness measures of association rules, despite of the classical support and confidence, were also considered [11,5,6]. Other interesting topics include constraint based mining, generalized association rules and so on. There is indeed some works that touched the issues of uncertainty or errors in association rule mining. The authors of [15] proposed approximate frequent patterns on the data with random noises; in [1], the notion of vague association rules is developed. However, none of these solutions are developed based on probabilistic data models.

Fast UPdate algorithm (FUP) [6] was proposed tomaintain association rules in dynamic databases. It worksby using frequent itemsets from previous mining in theoriginal database compares with frequent itemsets in theincrement database. For each iteration, a frequent itemsetin the increment database which is not a frequent itemsetin the original database will be rescanned in the originaldatabase and updates its support count. From the FUPexperiment result, even though it can save thecomputational time but it still needs to rescan an originaldatabase k times when new frequent k-itemsets are found.

Sliding Windows Filtering (SWF) [3] was proposed toreduce a number of rescanning times of an originaldatabase by dividing both original database and incrementdatabase into several partitions, and processing from thefirst partition to the last partition. There are 2 majorprocedures: preprocessing procedure and incrementalprocedure. Two new ideas are proposed in SWF: all size 1-iemsets are assumed to frequent itemsets and candidatek_3 itemsets are obtained from Ck-1 * Ck-1, these ideas candecrease a number

of candidate itemsets. This algorithmis a good work for both deleted and inserted database. In
addition, SWF requires only one time to rescan anoriginal database.

Negative Border algorithm (NBd) [4] was proposed toreduce the number of rescanning times of an originaldatabase by collecting both frequent itemsets and borderitemsets (itemset which is not frequent itemsets but itsproper subsets are frequent itemset). This algorithm issuccessful for reducing the number of rescanning timesbut a large number of border itemsets have to collect.Thus this Negative Border consumes a large amount ofmemory. Moreover, in the worst case, Negative Borderalgorithm needs to rescan an original database severaltimes when new frequent itemsets are discovered in anupdated database.

Priori studies approximate frequent patterns on noisy data, and also examined association rules on fuzzy sets. The notion of a vague association rule was developed. These solutions were not developed on probabilistic data models. For probabilistic databases, derived patterns based on their expected support counts found that the use of expected support may render important patterns missing. They discussed the computation of the probability that a pattern is frequent. While handled the mining of *single* items, the proposed solution can discover patterns with more than one item. The data model used in assumes that for each tuple, each attribute value has a probability of being correct. This is different from the tuple-uncertainty model, which describes the joint probability of attribute values within a tuple. The *supportprobability mass function* (or *support pmf*) for each PFIis calculated which gives the number of tuples (or *supportcount*) that contains a pattern.A simple way of finding PFI is to extract frequent patterns from every possible world. This is practically infeasible, since the number of possible worlds is exponentially large. The proposed system contains the simple and effective methods to prune infrequent patterns and also adopt a divide-and-conquer (DC) approach, which achieves in $O(n \log 2\, n)$ time.

2.2.1 Divide and Conquer Approach:

DC Approach finds the frequent pattern by dividing the database into horizontal partition where each partition updates its support count for Probability mass function calculation.

3 PROBLEM DEFINITION

3.1 Tuple Uncertainty

In the tuple uncertainty model, each tuple or transaction is associated with a probability value. We assume each transaction $tj \in D$ is associated with a set of items and an existential probability $Pr(t_j) \in (0,1]$ which indicates that tj exists in D with probability $Pr(t_j)$. Table 1 summarizes the list of symbols used in this paper.

3.2 Frequent patterns and Association Rule

A transaction is a set of *items* (e.g., goods bought by a customer in a supermarket). A set of items is also called an *itemset*or a *pattern*.

Notation	Meaning
Probabilistic Database	

ISSN 2348 - 7968

PDB	A probabilistic database of size *n*
T_i	ID of a tuple in *PDB*, where $i=1 \ldots n$
$T_i.S$	Set of items contained in T_i
$T_i.P$	Existential probability of T_i
W	Set of possible worlds expanded from *PDB*
W_i	A possible world, where $W_i \in W$
P(E)	Probability of Event *E*
Probabilistic Frequent patterns	
minsup	support threshold
minprob	probability threshold
sup(X)	Support count of pattern *X*
$f_x(k)$	Support pmf of *X* in *PDB*
cnt(X)	No of tuples for which $p_i^X>0$
esup(X)	Expected support of pattern *X*
p_i^X	Probability that *X* occurs in T_i
L^X	Inverted probability list of *X*
X.exItem	Exclusive item of *X*
X.cnt	Length of L^X
Probabilistic Association Rule	
minconf	Confidence threshold
Conf(X=>Y)	Confidence of *X=>Y*

Table 1: Summary of Notations

Given a transaction database of size *n* and a pattern *X*, we use *sup*(*X*) to denote the *support* of *X*, i.e., the number of times that *X* appears in the database. A pattern *X* is *frequent* if:

$$sup(X) \geq minsup \quad (1)$$

where $minsup \in N \cap [1,n]$ is the *support threshold*. Given patterns *X* and *Y* (with $X \cap Y = \phi$), if pattern XY is frequent, then X is also frequent (called the antimonotonicity property). Also, *X* =>*Y* is an association rule if the following conditions hold:

$$sup(XY) \geq minsup \quad (2)$$

$$\frac{sup(XY)}{sup(X)} \geq minconf \quad (3)$$

where $\frac{sup(XY)}{sup(X)}$, denoted by *conf*(*X* =>*Y*), is the *confidence* of *X* =>*Y* , and $minconf \in R \cap (0,1]$ is the *confidence threshold*. To verify Equation 3, the values of *sup*(*XY*) and *sup*(*X*) have to be found first.

4 THE APRIORI ALGORITHM

Apriori uses the *bottom-up* framework such that each item is tested to see whether it is a Frequent pattern.All probabilistic frequent singletons then have their support probability mass function computed, and are used to generate size-2 patterns called candidate patterns. These patterns are examined to see which are frequent patterns. The size-2 p-FPs again have their support pmfs evaluated, and are used to create size-3 candidate patterns. The process is repeated until no more frequent patterns are found.

The database is exact and scanned once to find the support count of item and test it with Equation

sup(X) >= minsup

sup(X) = Support of X

minsup = minimum support threshold (in %), range (0,100]

4.1 Initial Infrequent pattern purning

To Prune the infrequent item set the following two lemma are satisfied.

1. If cnt(X) <minsup, then X is not a p-FP
2. Let μ= esup(X), and σ= (minsup- μ-1)/ μ then X is not a p-FP if:

σ >=2e -1 and2^-(σ μ) <minprob,

or

0 < σ< 2e -1 and e^- (σ^2 μ)/4 <minprob

cnt(X) = number of tuples that contain pattern X esup(X) = the expected support of X, which can be found by summing up all the tuples.

4.2 Divide and Conquer Algorithm

1.Given a pattern X, compute the pmf for the case where PDB has only one tuple.

2. Otherwise PDB is horizontally partitioned into two databases (D1 and D2). Then, DC is recursively invoked on D1 and D2 to obtain X's pmf for each database. The two pmfs are used to generate the support pmf of X.

Complexity:

Time Complexity is $O(n \log2 n)$. Thus, DC is more efficient and scalable than DP for large datasets. The space complexity is $O(n)$. After calculating SupportPmf check against the equation

P(sup(X) >=minsup) >=minprob

Thus the Apriori uses the Bottom Up Framework for Calculating the Support Probability Mass Function.

5.UPDATING FREQUENT PATTERNS IN INCREMENTAL MINING

Frequent Itemsets from the Original Database becomes invalid while new transcations are added to the database. To handle this problem Expected Frequent Itemsets are calculated which is infrequent now in the mining but later while new transcations are added it may became frequent. According to the algorithm, the size 1-candidate itemsets of an updated database canbe found by combining the size 1-candidate itemsets of an original databasewiththe 1-candidate itemsets of an increment database. Then, the support count ofSize-1 Candidate Itemsets of an Updated databasecan be updated by scanning only an increment database. Then, the size 1-frequent and expected itemsets of an updated database can be found.

Algorithm 5.1:Updating Singleton Frequent Patterns

Step 1: Scan db and find C(X,db),μ(X),P(X) for all X ϵ C_1^{DB} U C_1^{db}

Step 2: For all X ϵC_1^{DB} U C_1^{db} do

Step 3: C(X,UP)=C(X,DB)+C(X,db)

Step 4: end do

Step 5: $F_1^{UP}=\{X\epsilon C_1^{UP}|C(X,UP)>=\sigma^{UP}\}$

Step 6: $EF_1^{UP}=\{X\epsilon C_1^{UP}|\rho^{UP}<=C(X,UP)>=\sigma^{UP}\}$

The Algorithm describes found the Updated Database with the count values of Original Database and the Incremental Database.Size-1 frequent Itemset can be find out if the item in the updated database is greater than minimum support.Expected frequent itemset can be find out if the item’s count is in between minimum support and expected minimum frequent.

Algorithm 5.2: Generating and Updating Incremental Dataset size-k Frequent Itemset

Step 1:$C_k^{db}=F_{k-1}^{db}*F_{k-1}^{db}$

Step 2: For all X ϵC_k^{db}do

Step 3:$C_k^{new}=\{X\epsilon C_2^{db}|X\epsilon(F_2^{DB}UEF_2^{DB})\}$

Step 4: end do

This algorithm finds the k-candidate itemsets of an increment database C_k^{db} by joiningF_{k-1}^{db}withF_{k-1}^{db}.C_k^{new} will keep only the k- candidate itemsets of an increment database whose subsets of the k- candidate itemsets are in the (k-1)-updated frequent itemsets.

Algorithm 5.3: Updating Incremental Dataset with Original Database

Step 1: Scan DB and obtain count C(X,DB)for all $Temp\_scanDB_k$

Step 2: For all $Temp\_scanDB_k$ do

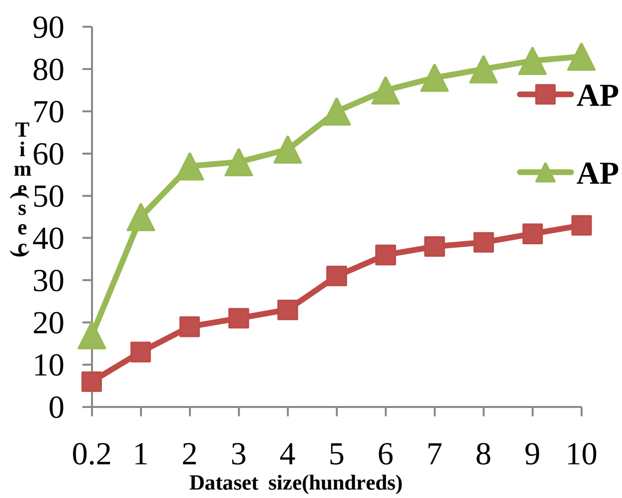
Step 3: C(X,UP)=C(X,DB)+C(X,db)

Step 4: end do

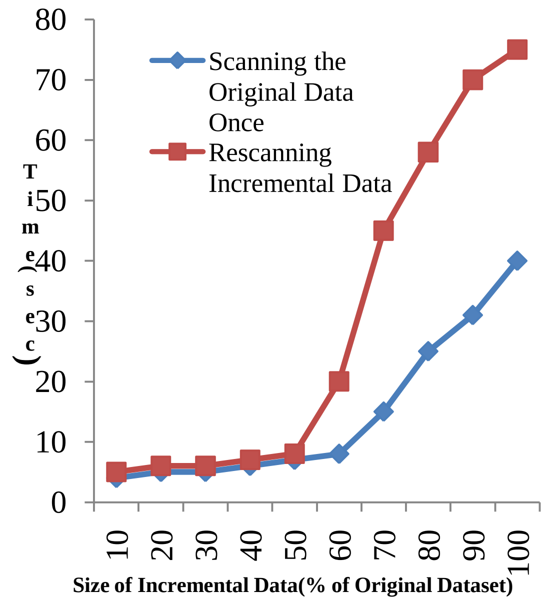
To reduce the number of itemset for scanning original database, if sum of any new candidate's support count and support of $prob_{pl}$ minus 1 is greater than minimum support of updated database, then it will be moved to Temp_scanDB.Then the items in the temp_scanDB updated with the Original Database.

6 RESULT

Original Database Evaluation:



Comparison for the Original Dataset between Apriori and the Probability Based Apriori which shows that AP DC yields better Performance.



Size of Incremental Data as the percentage of Original Dataset is executed with the Average Execution time to show the Performance of Incremental Dataset rescanning the Original Data.

7 CONCLUSION

There is the problem of maintaining mining results for changing, or evolving, databases. The type of evolving data address here is about the appending, or insertion of a batch of tuples to the database. Tuple insertion is common in most of online and location based applications and hence we need to derive the PFIs for the new database to manage them . A straightforward way of refreshing the mining results is to re-evaluate the whole mining algorithm on the new database. This can be costly, however, when new tuples are appended to the database at different time instants. In fact, if the new database D_{new} is similar to its older version, D, it is likely that most of the PFIs extracted from D remain valid for D_{new}. Based on this intuition, developed incremental mining algorithms, which use the PFIs of D to derive the PFIs of D_{new}, instead of finding them from scratch.An incremental mining algorithm which discovers exact PFIs. As the experiments show, when the change of the database is small, running our incremental mining algorithms on D_{new} is much faster than finding PFIs on D_{new} from scratch.

8 REFERENCE

[1] A. Deshpande et al. Model-driven data acquisition in sensor networks. In VLDB '07: Proceedings of the 33rd International Conference on Very Large Databases, 2004.

[2] Ramesh C. Agarwal, Charu C. Aggarwal, and V. V. V. Prasad.Depth first generation of long patterns. In KDD '00: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 108{118, New York, NY, USA, 2000. ACM.

[3] Ramesh C. Agarwal, Charu C. Aggarwal, and V. V. V. Prasad. A tree projection algorithm for generation of frequent item sets. Journal of Parallel and Distributed Computing, 61(3):350-371, 2001.

[4] C. Aggarwal. On density based transforms for uncertain data mining. In ICDE '07: Proceedings of the 23rd International Conference on Data Engineering, 2007.

[5] C.C. Aggarwal and P. Yu.A survey of uncertain data algorithms and applications.IEEE Transactions on Knowledge and Data Engineering, 21(5), 2009.

[6] CharuAggarwal, Yan Li, Jianyong Wang, and Jing Wang.Frequent pattern mining with uncertain data. In KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009.

[7] L. Antova, T. Jansen, C. Koch, and D. Olteanu.Fast and simple relational processing of uncertain data. In ICDE '08: Proceedings of the 24th International Conference on Data Engineering, 2008.

[8] Omar Benjelloun, Anish Das Sarma, Alon Halevy, and Jennifer Widom.Uldbs: databases with uncertainty and lineage. In VLDB '06: Proceedings of the 32nd International Conference on Very Large Databases, 2006.

[9] Hong Cheng, Philip Yu, and Jiawei Han. Approximate frequent itemset mining in the presence of random noise. Soft Computing for Knowledge Discovery and Data Mining, 2008.

[10] Chun Kit Chui, Ben Kao, and Edward Hung. Mining frequent itemsets from uncertain data. In PAKDD '07: Proceedings of the 11th Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2007.

[11] G. Cormode and M. Garofalakis.Sketching probabilistic data streams. In SIGMOD '07: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, 2007.

[12] Ming Hua, Jian Pei, Wenjie Zhang, and Xuemin Lin. Ranking queries on uncertain data: a probabilistic threshold approach. In SIGMOD '08: Proceedings of the 2008 ACM SIGMOD International Conference on Managementof Data, 2008.

[13] J. Huang et al. MayBMS: A Probabilistic Database Management System. In SIGMOD '09: Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data, 2009.

[14] Ben Kao, Sau Lee, David Cheung, WaiShing Ho, and K. Chan. Clustering uncertain data using voronoi diagrams. In ICDM '08: Proceedings of the 8th International Conference on Data Mining, 2008.

[15] Jian Li and AmolDeshpande. Consensus answers for queries over probabilisticdatabases. In PODS, 2009.

[16] M. Yiu et al. E_cient evaluation of probabilistic advanced spatial queries on existentially uncertain data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 2009.