Discovering Threshold-based Frequent Closed Itemsets over Probabilistic Data

Yongxin Tong[#1], Lei Chen[#2], Bolin Ding[*3]

[#]*Department of Computer Science and Engineering, Hong Kong Univeristy of Science and Engineering*

[1 2]{yxtong, leichen}@cse.ust.hk

[*]*Department of Computer Science, University of Illinois at Urbana-Champaign*

[3]bding3@uiuc.edu

***Abstract*—In recent years, many new applications, such as sensor network monitoring and moving object search, show a growing amount of importance of uncertain data management and mining. In this paper, we study the problem of discovering threshold-based frequent closed itemsets over probabilistic data. Frequent itemset mining over probabilistic database has attracted much attention recently. However, existing solutions may lead an exponential number of results due to the downward closure property over probabilistic data. Moreover, it is hard to directly extend the successful experiences from mining exact data to a probabilistic environment due to the inherent uncertainty of data. Thus, in order to obtain a reasonable result set with small size, we study discovering frequent closed itemsets over probabilistic data. We prove that even a sub-problem of this problem, computing the frequent closed probability of an itemset, is #P-Hard. Therefore, we develop an efficient mining algorithm based on depth-first search strategy to obtain all probabilistic frequent closed itemsets. To reduce the search space and avoid redundant computation, we further design several probabilistic pruning and bounding techniques. Finally, we verify the effectiveness and efficiency of the proposed methods through extensive experiments.**

I. INTRODUCTION

Recently, due to the wide application of uncertain data, such as sensor network monitoring [16, 17], moving object search [8], object identification [5], etc., mining frequent itemsets over uncertain (or probabilistic) data has attracted much attention from the data mining and database communities. However, like its counterpart, mining frequent itemsets over exact data, mining frequent itemsets over uncertain data [4, 9, 22] can produce an exponential number of frequent itemsets which cause the mining results less useful. A classical solution is proposed to address the similar problem in exact data, called mining frequent closed itemsets [18]. The approach intends to find frequent itemsets whose supports are different from the supports of all their supersets; thus, redundant itemsets will be excluded. Since mining frequent closed itemsets offers high-level compression, in this paper, we would like to investigate this compression strategy on uncertain data in order to reduce the huge number of frequent itemsets generated by the previous mining methods. Unfortunately, the solution for mining frequent closed itemsets over exact data cannot be directly applied to uncertain data as shown by following example.

TABLE I
AN UNCERTAIN TRANSACTION DATABASE

TID	Location	Weather	Time	Speed	Prob.
T1	HKUST	Rain	2:30-3:00	20-30	0.9
T2	HKUST	Rain	2:30-3:00	null	0.6
T3	HKUST	Rain	2:30-3:00	null	0.7
T4	HKUST	Rain	2:30-3:00	20-30	0.9

TABLE II
THE CONCISE FORMAT OF TABLE I

TID	Transaction	Prob.
T1	a b c d	0.9
T2	a b c	0.6
T3	a b c	0.7
T4	a b c d	0.9

Example 1.1 (*Motivation*). Most existing intelligent traffic systems depend on sensors to collect real-time monitoring data. According to logs of vehicles monitoring, some hidden traffic patterns, such as "the gate of HKUST crossroad always had traffic jam at 2-3 p.m.", can be found and be utilized to predict potential traffic problems in the future. However, due to the limitation of sensors, data collected by these sensors are often uncertain. Table I shows synthesized transaction data that records traffic situation of one crossroad. For its commonality in data processing, we also adopt the tuple-uncertainty data model as in this paper [22]. Namely, in Table I, every line represents one tuple which records a sensor reading and its existence probability, where TID is used to identify distinct tuples. For the ease of presentation, we use Table II, which represents each distinct attribute value with a distinct symbol.

In uncertain database, in order to explain the existence of data, *Possible World Semantics* are usually adopted [1, 4, 9, 10, 15, 22, 23]. Specifically, an uncertain transaction database can be viewed as a set of exact transaction databases. Every possible world containing zero or more tuples represents one exact transaction database. Table III shows all possible worlds originated from Table II and their corresponding probabilities. (in the first three columns) For instance, PW5 (the fifth possible world) in Table III illustrates the appearance of T1, T2, T3 and the absence of T4 and the probability of PW5 equals $0.9\times0.6\times0.7\times(1-0.9)=0.0378$.

According to the definition of probabilistic frequent itemsets [4, 22], an itemset is a *probabilistic frequent itemset* if and only if the sum of probabilities of possible worlds

TABLE III
POSSIBLE WORLDS AND FREQUENT CLOSED ITEMSETS IN POSSIBLE WORLDS

PW	Transactions	Prob.	Frequent Closed Itemsets
PW1	T1	0.0108	{}
PW2	T1, T2	0.0162	{abc}
PW3	T1, T3	0.0252	{abc}
PW4	T1, T4	0.0972	{abcd}
PW5	T1, T2, T3	0.0378	{abc}
PW6	T1, T2, T4	0.1458	{abc} {abcd}
PW7	T1, T3, T4	0.2268	{abc} {abcd}
PW8	T1, T2, T3, T4	0.3402	{abc} {abcd}
PW9	T2	0.0018	{}
PW10	T2, T3	0.0042	{abc}
PW11	T2, T4	0.0162	{abc}
PW12	T2, T3, T4	0.0378	{abc}
PW13	T3	0.0028	{}
PW14	T3, T4	0.0252	{abc}
PW15	T4	0.0108	{}
PW16	{}	0.0012	{}

where such itemset is no less than the given *minimum support* (*min_sup*) is larger than the given *probabilistic frequent threshold* (*pft*). In this example, given min_sup=2 and pft = 0.8, we can observe that there are 15 probabilistic frequent itemsets where 7 have the same frequent probability, 0.9726, and the other 8 have the same frequent probability, 0.81. However, we cannot further distinguish their significance in 15 itemsets. ■

In order to address this issue, we study the problem of *mining probabilistic threshold-based frequent closed itemsets* in this paper.

According to the definition of frequent closed itemset in exact databases, an itemset must satisfy two conditions if the itemset is a *frequent closed itemset*. The first one is that the itemset must be *frequent*, namely, the support of such itemset must be no less than the given minimum support. The second one is that the itemset must be *closed*, namely, its support must be larger than supports of any of its supersets. Given an uncertain database, we can check the degree that an itemset is a frequent closed itemset by measuring the *frequent closed probability* which equals the sum of probabilities of possible worlds where such itemset is a frequent closed itemset. Then, an itemset is defined as a probabilistic threshold-based frequent closed itemset (which will be called as probabilistic frequent closed itemset in this paper for simplicity) if and only if its frequent closed probability is larger than the given *probabilistic frequent closed threshold* (*pfct*). We further explain the concept using the following example.

Example 1.2 (*Probabilistic Frequent Closed Itemset*). We can get all frequent closed itemsets in each possible world in the last column of Table III (In Table III, {} means an empty set). For instance, assume min_sup=2 and pfct = 0.8, we can compute the frequent closed probability of {abc}=Pr(PW2)+ Pr(PW3)+Pr(PW5)+Pr(PW6)+Pr(PW7)+Pr(PW8)+Pr(PW10) +Pr(PW11)+Pr(PW12)+Pr(PW14)= 0.8754, and that of {abcd} is equal to 0.81. Moreover, frequent closed probabilities of 13 other probabilistic frequent itemsets are 0. Among 15 probabilistic frequent itemsets, {abc} and {abcd} are only probabilistic frequent closed itemsets. So, {abc} and {abcd} can approximately express whole information of all 15 probabilistic frequent itemsets. Thus, probabilistic frequent closed itemsets can compress the size of all probabilistic frequent itemsets reasonably. ■

With newly defined probabilistic frequent closed itemsets over uncertain data, we target on how to discover these itemsets efficiently over uncertain data. A naïve method first enumerates all possible worlds of a given uncertain database and mines all frequent closed itemsets in each possible world. Then frequent closed probability of each itemset can be computed according to the probabilities of possible worlds. Finally, those itemsets whose frequent closed probabilities are larger than the given threshold (pfct) will be returned as the results. Clearly the naïve solution is very costly due to the exponential number of possible worlds. In fact, the operation of computing frequent closed probability is a #P-Hard problem. Therefore, we need to find efficient solutions to compute the probability.

To summarize, in this paper, we make the following contributions:

1. We propose a new problem, mining probabilistic frequent closed itemsets over uncertain data and prove that even a sub-problem of this problem, computing the frequent closed probability of an itemset, is #P-Hard.
2. We design an efficient algorithm based on depth-first search to discover all probabilistic frequent closed itemsets. In addition, we propose several probabilistic pruning and bounding techniques to reduce search space and avoid redundant computation.
3. Extensive experiments demonstrate that the effectiveness and efficiency of our algorithm.

The rest of the paper is organized as follows. In Section II, we review the mining frequent patterns in exact data and the mining probabilistic frequent patterns over uncertain data. Our problem definition and complexity analysis of the studied problem is introduced in Section III. In Section IV, we propose our efficient algorithm based on depth-first search strategy and several probabilistic pruning techniques. Experimental studies are reported in Section V. Finally, we conclude in Section VI.

II. RELATED WORK

In this section, we will review the related work from two categories, frequent pattern mining in exact and uncertain data.

A. Frequent Pattern Mining over Exact Data

Since Rakesh Agrawal proposed the concept of association rule and the Apriori algorithm respectively [2, 3], many efficient algorithms about mining frequent itemsets have been proposed, such as FP-growth [13], Eclat [28], and so on. However, mining the complete set of frequent itemsets produces a lot of redundant itemsets because of the well-known downward closure property. To solve such a problem, a classical approach of mining frequent closed itemsets was proposed [18]. This approach aims to discover the frequent itemsets whose supports are different from the supports of any their supersets. Since the problem of mining frequent closed itemsets has been proposed, there exist many efficient

algorithms, such as CLOSET [19], Closet+ [24], FPclose [12], Charm [29], and so on, and the inspiration of mining frequent closed itemset is also extended to other complex structural data, for example mining frequent closed sequential patterns [11, 25, 27], mining frequent closed subgraphs [26], etc. Although there are a lot of works on mining frequent closed patterns, all these algorithms are designed for exact data and cannot be extended to uncertain data directly.

B. Frequent Pattern Mining over Uncertain Data

Another set of researches related with our work are mining frequent patterns over uncertain data. With the advanced research on management of uncertain data, a few novel works of mining frequent patterns over uncertain data have emerged in recent years. According to different interpretations of the uncertainty, existing work for mining frequent pattern over uncertain data are grouped into two categories: approaches based on expected support model and approaches based on probabilistic frequent model. In the expected support model, the first work proposed by Chui et al. [9] summed up all the probabilities of an itemset in all transactions which contain such itemset in order to estimate the support of such itemset. Based on the concept of expected support, Chui et al. proposed the U-Apriori algorithm [9] for mining all frequent itemset in uncertain data and an efficient decremental pruning strategy [10]. Later, with the same expected support model, UF-growth algorithm adopted the strategy of FP-growth to mining frequent itemset [15]. Three novel algorithms [1]: UApriori, UH-mine and UFP-tree, respectively, extended from three classical algorithms, Apriori [3], H-mine [20] and FP-growth [13], were designed to find frequent itemsets. A sampling-based algorithm [7] was also proposed to approximately mine frequent itemsets based on the expected support. In addition, the problem of mining frequent subgraph patterns in uncertain graphs was also studied [31].

In addition, under the probabilistic frequent model, the dynamic programming strategy was adopted to obtain frequent probabilities of itemsets [4]. Then, two efficient algorithms, the bottom-up and the top-down, were designed to find all probabilistic frequent itemsets in tuple-uncertainty model [22], respectively. In addition, an efficient algorithm based on the Poisson binomial distribution was proposed to accelerate mining probabilistic frequent itemsets recently [23]. Furthermore, [30] proposed an exact and a sampling-based algorithm to discover likely frequent items in probabilistic data stream. Moreover, mining probabilistic frequent subgraph patterns in uncertain graphs has also been studied [32].

Although the above researches can obtain probabilistic frequent itemsets efficiently, none of them studied mining probabilistic frequent closed itemsets. A natural idea is to discover all probabilistic frequent closed itemsets by simply checking the result set generated from algorithms of [22]. Unfortunately, such solution is inefficient because the checking (namely, computing the frequent closed probability of each itemset) is #P-Hard. Thus, in order to tackle this challenge, our work mainly focuses on designing several efficient pruning techniques together with a new search strategy to get probabilistic frequent closed itemsets.

TABLE IV
THE UPDATED TABLE II

TID	Transaction	Prob.
T1	a b c d	0.9
T2	a b c	0.6
T3	a b c	0.7
T4	a b c d	0.9
T5	a b	0.4
T6	a	0.4

In particular, a close related research with our work, mining probabilistic frequent closed itemsets in uncertain database [34], has been proposed recently. This work defines a new concept, called the probabilistic support. When a probabilistic frequent threshold is given, the probabilistic support of each itemset equals the maximum support which can satisfy the probabilistic frequent threshold. Thus, in [34], an itemset X is a probabilistic frequent closed itemset if and only if the probabilistic support of such itemset is larger than the min_sup and probabilistic supports of all supersets of this itemset are smaller than the probabilistic support of this itemset. The biggest difference between [34] and our work lies in problem definitions: our approach can exactly measure the degree that an itemset is frequent closed itemset in all possible worlds. However, [34] is unable to guarantee that a probabilistic frequent itemset is always a frequent closed itemset in possible worlds which are used to obtain the probabilistic support of such itemset. We will further explain our difference by the following example. To better illustrate, we first add two transactions into Table 2 and build a new uncertain database shown in Table IV. Based on Table IV, We can obtain a contradiction under the definition of [34]. Assume min_sup=2 and pft=0.9 which is same as *Example 1.2*, [34] will return {a} and {abcd} as the result. When min_sup=2 and pft=0.8, {ab} and {abcd} show up as result. The frequent probabilities of {a} and {ab} are 0.99 which already satisfy both the given thresholds, thus, the frequent closed itemset should not vary (from {a} to {ab}) while the threshold decreases from 0.9 to 0.8. In contrast, with the same min_sup=2, no matter how the probabilistic frequent threshold changes, our approach always returns {abc} and {abcd} as results having frequent closed probabilities as 0.88 and 0.99 respectively. Moreover, {a} and {ab}, whose frequent closed probabilities are only 0.4, won't be returned as final results. In short, our result includes more useful information and guarantees the strict probabilistic semantics.

III. PROBLEM DEFINITION

A. Frequent Closed Itemset in Exact Data

Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of distinct items. We call a non-empty subset of I, X, as an itemset or pattern. For brevity, we use $X = x_1x_2\ldots x_n$ to denote itemset $X=\{x_1, x_2, \ldots, x_n\}$. X is called the *l*-itemset if it has *l* items. Given a (exact) transaction database, TDB, each transaction is denoted as a tuple <tid, X> where tid is a transaction identifier. The number of transactions in TDB containing X is called the support of X, denoted as support(X).

TABLE V
SUMMARY OF NOTATIONS

Notation	Meaning
min_sup	minimum support threshold
pft	probabilistic frequent threshold
pfct	probabilistic frequent closed threshold
$Pr_F(X)$	frequent probability of itemset X
$Pr_C(X)$	closed probability of itemset X
$Pr_{FC}(X)$	frequent closed probability of itemset X

In addition, a summary of notations is shown in Table V.

Definition 3.1 (*Frequent Itemset*). Given a minimum support threshold *min_sup*, an itemset X is frequent if and only if (iff) support(X) is equal to or larger than the *min_sup*.

Definition 3.2 (*Closed Itemset*). An itemset X is a closed itemset if and only if there is no proper superset $Y \supset X$ such that support(Y) = support(X). (Y is a non-empty subset of I)

Definition 3.3 (*Frequent Closed Itemset*). An itemset X is a frequent closed itemset if and only if it is frequent and there is no proper superset $Y \supset X$ such that support(Y) = support(X). (Y is a non-empty subset of I)

B. Probabilistic Frequent Closed Itemset

Given an uncertain transaction databases and an itemset X, we can treat support(X) as a random variable and derive its distribution from all possible worlds. Once we have support(X)'s distribution, we can define the frequent probability and the probabilistic frequent itemset as follows [4, 22].

Definition 3.4 (*Frequent Probability*). Given a minimum support *min_sup*, and an itemset X, X's frequent probability, denoted as $Pr_F(X)$, is the probability that X's support is equal to or larger than the *min_sup*.

Definition 3.5 (*Probabilistic Frequent itemset [22]*). Given a minimum support *min_sup*, and a probabilistic frequent threshold *pft*, an itemset X, X is a probabilistic frequent itemset if X's frequent probability is larger than the probabilistic frequent threshold, namely,

$$\Pr\{\text{support}(X) \geq min\_sup\} = Pr_F(X) > pft$$

Now we define the closed probability, frequent closed probability and the probabilistic frequent closed itemset. (Please note that even though an itemset may be a closed itemset, it may not be a frequent one)

Definition 3.6 (*Closed Probability*). Given an itemset X, X's closed probability, denoted as $Pr_C(X)$, is the sum of the probabilities of possible worlds where X is a closed itemset.

Definition 3.7 (*Frequent Closed Probability*). Given an itemset X, X's frequent closed probability, denoted as $Pr_{FC}(X)$, is the sum of the probabilities of possible worlds where X is a frequent closed itemset.

Definition 3.8 (*Probabilistic Frequent Closed Itemset*). Given a probabilistic frequent closed threshold, *pfct*, an itemset X, X is a probabilistic frequent closed itemset if X's frequent closed probability is larger than the probabilistic frequent closed threshold, namely,

$$\Pr\{\text{X is frequent closed itemset}\} = Pr_{FC}(X) > pfct$$

Based on the above definitions, we define the problem that we studied in this paper.

Problem Statement. Given an uncertain transaction database *UTD*; a minimum support threshold *min_sup*; a probabilistic frequent closed threshold *pfct*, we are required to find all probabilistic frequent closed itemsets in *UTD*.

Obviously, the problem of computing the closed probability of an itemset is a special case of the problem of computing frequent closed probability of such itemset when *min_sup*=1. The hardness of the computational complexity of a general problem is at least same as that of its special case. In following Theorems, we first prove the problem of computing the closed probability of an itemset in *UTD* is #P-Hard. Thus, the problem of computing the frequent closed probability of an itemset in *UTD* is also #P-Hard.

Theorem 3.1. It is #P-hard to compute the closed probability of an itemset in an uncertain transaction database.

Proof. In order to prove the problem of computing the closed probability of an itemset is #P-hard, we reduce it from the monotone DNF counting problem (#MDNF), which is known to be #P-complete [33].

Consider an instance of #MDNF: Given a monotone DNF formula $F = C_1 \vee C_2 \ldots \vee C_n$, with n clauses and m Boolean variables $v_1, v_2, \ldots, v_m$. In each clause $C_i = y_1 \wedge y_2 \ldots \wedge y_l$, we have $y_j \in \{v_1, v_2, \ldots, v_m\}$ and each variable can appear at most once. The #MDNF problem is to count the number of satisfying assignments of variables for the formula F.

We map the above instance of #MDNF to an uncertain transaction database $UTD = \{T_1, T_2, \ldots, T_m\}$, where each itemset T_j corresponds to a variable v_j. Create itemset X and another n items $e_1, e_2, \ldots, e_n$, where e_i corresponds to a clause C_i. For every itemset, we have $X \subseteq T_j$; and for each T_j and each e_i, we have $e_i \in T_j$ if and only if v_j does NOT appear in the clause C_i. In this database, each itemset appears with probability ½. For instance, given a formula $F = (v_1 \wedge v_2 \wedge v_3) \vee (v_1 \wedge v_2 \wedge v_4) \vee (v_2 \wedge v_3 \wedge v_4)$, we map it to uncertain transaction database UTD as in Table VI.

TABLE VI
THE UNCERTAIN TRANSACTION DATABASE BASED ON THE GIVEN DNF FORMULA

TID	Transaction	Prob.
T1	X e_3	½
T2	X	½
T3	X e_2	½
T4	X e_1	½

To complete the proof, we claim that in the #MDNF instance, the number of satisfying assignment is N if and only if in *UTD*, itemset X is NOT closed with probability $N/2^m$ (closed with probability $1 - N/2^m$). For some technical reasons, X is also said to be NOT closed if X does not appear in an instance of *UTD* (e.g. in an empty instance). To prove this claim, we map an assignment of F to an instance of *UTD* in the possible world, in such a way that v_j = true if and only if T_j does NOT appear -- we observe that this assignment is satisfying if and only if in the corresponding instance of *UTD*, X is NOT closed. Due to the space limit, details to prove this claim are deferred to the complete version of this paper. ■

Theorem 3.2. It is #P-hard to compute the frequent closed probability of an itemset when the min_sup is given in an uncertain transaction database.

Proof. The problem of computing the closed probability of an itemset is a special case of the problem of computing frequent closed probability of such itemset when min_sup=1. Based on *Theorem 3.1*, we know the problem of computing the closed probability of an itemset is #P-Hard, so the problem of computing the frequent closed probability of an itemset is also #P-Hard. ■

IV. The Depth-First Mining Algorithm

Based on the definition of probabilistic frequent closed itemsets, a naïve solution needs to enumerate all possible worlds and itemsets firstly, and then check the probability of each itemset. However, there are an exponential number of possible worlds and itemsets, so the naïve solution is not scalable when the database size becomes large. Thus, in this paper, we design a Bounding-Pruning-Checking framework to avoid redundant computation.

Fig. 1 illustrates major steps of the general framework, including constructing single item candidate set, bounding & pruning, and checking phases. In remaining parts of this Section, we give the details about each step. We introduce the *Chernoff-Hoeffding Bound-based Pruning* in Section A. Three pruning methods, *Superset Pruning, Subset Pruning*, and *Frequent Closed Probability Bound-based Pruning*, are discussed in Section B. In the end, we present a depth-first search-based mining algorithm.

Procedure MPFCI_Framework {

1: discover all the single probabilistic frequent items, called Cand, using Chernoff-Hoeffding bound and sort all the items in Cand based on the alphabetic order.
//Phase 1: Construct initial probabilistic frequent single items
2: for each item X in Cand, extend X using a depth-first search like strategy to its supersets with X as prefix and perform Chernoff-Hoeffding bound-based pruning, superset pruning, subset pruning, and frequent closed probability bound-based pruning.
//Phase 2: Bounding and Pruning
3: check the frequent closed probability of itemsets which cannot be pruned and return the result set.
//Phase 3: Checking }

Fig. 1 Framework of MPFCI

A. Bounding Frequent Probability

As shown in the framework, in the first phase, we will compute a candidate set of probabilistic frequent single items.

According to definitions of frequent probability, closed probability and frequent closed probability (*Definition 3.4, 3.6, 3.7*), we know that the frequent closed probability of an itemset is not larger than either the frequent probability or the closed probability. Thus, both the frequent probability and closed probability of an itemset are upper bounds of the frequent closed probability of this itemset. In addition, we have also proven that computing closed probability is a #P-Hard problem and we know that there exists the polynomial time algorithm to compute the frequent probability. Thus, it is a natural idea to bound the frequent closed probability by the frequent probability. However, it is meaningless to compute the exact frequent probability if it is very small. Thus, instead of directly computing the frequent probabilities using the existing method, we propose a pruning technique based on the *Chernoff-Hoeffding bound* [35] to filter out probabilistic infrequent items. This pruning is further explained by the following lemma.

Lemma 4.1 (*Chernoff-Hoeffding Bound-based Pruning*). Given an itemset X, an uncertain transaction database *UTD*, X's expected support μ, a minimum support threshold *min_sup*, probabilistic frequent closed threshold *pfct*, an itemset X is a probabilistic infrequent itemset if,

$$\begin{cases} e^{-2n^2\delta^2} < pfct & \delta > \mu \\ e^{-2n\delta^2} < pfct & 0 < \delta < \mu \end{cases}$$

where $\delta = (\text{min\_sup} - \mu - 1)/n$ and n is the number of transactions in UTD.

Proof. According the definition of the frequent probability, $\Pr\{\text{support}(X) \geq min\_sup\}$ equals $\Pr\{\text{support}(X) > min\_sup - 1\}$. Moreover, based on the Chernoff-Hoeffing bound [35], we can get the following inequality:

$$\Pr\{|\sup(X) - \mu| > n\delta\} < \begin{cases} e^{-2n^2\delta^2} < pfct & \delta > \mu \\ e^{-2n\delta^2} < pfct & 0 < \delta < \mu \end{cases}$$ ■

B. Computing Frequent Closed Probability

1. Basic idea

Again, based on definitions of frequent probability, closed probability and frequent closed probability (Definiton 3.4, 3.6, 3.7), we can find that the frequent closed probability of a given itemset, X, equals to the frequent probability of X subtracted by the frequent non-closed probability of X. Formally, we define the frequent non-closed probability of an itemset as follows.

Definition 4.1 (*Frequent Non-Closed Probability*). Given an itemset X, an uncertain transaction database *UTD*, a minimum support threshold *min_sup*, the frequent non-closed probability of X is the probability that there exists at least one superset of X which always appears together with X at least *min_sup* times.

Assume that there are m other items, $e_1, e_2, ..., e_m$, besides items of X in UTD, the frequent non-closed probability of X is $\Pr(C_1 \vee ... \vee C_m)$, where C_i denotes an event that the superset of X, X+ e_i, always appear together with X at least *min_sup* times.

According to Definition 4.1, we know that computing the frequent non-closed probability is equivalent to computing the probability of the corresponding DNF formula. Hence, we can use the Inclusive-Exclusive Principle to compute the frequent non-closed probability as follows. Let $F = C_1 \vee ... \vee C_m$,

$$\Pr(F) = \sum_{1 \leq i \leq m} \Pr(C_i) - \sum_{1 \leq i < j \leq m} \Pr(C_i \wedge C_j) + ... + (-1)^{m-1} \sum_{1 \leq i_1 < ... < i_m \leq m} \Pr(C_{i_1} \wedge ... \wedge C_{i_m})$$

where $\Pr(C_i)$ means the probability that the superset of X, X+ e_i, always appear together with X at least *min_sup* times.

Based on the above formula, it is important to know how to compute Pr(C_i), which is consists of two independent parts of probabilities: 1) the first part is the probability that the superset of X, X+e_i, always appear together with X, and 2) the second part is the probability that both X and X+e_i appear at least *min_sup* times simultaneously. (Thus, this part probability is just equal to the frequent probability of X+e_i since X+e_i contains X). Thus, Pr(C_i) can be computed as follows,

$$\Pr(C_i) = \prod_{T_j \in T} (1 - \Pr(T_j)) \times \Pr_F(X + e_i)$$

where T represents a set of transactions which contain itemset X but do not contain itemset X+e_i, and the frequent probability of X+e_i can be computed by the dynamic programming approach [22].

To speed up the computation of the frequent closed probability, we introduce several pruning and bounding techniques to reduce the computation cost in next several subsections.

2. Pruning Techniques

Although we use the Inclusive-Exclusive Principle to compute the frequent closed probability of an itemset, the cost of computation is still high. Fortunately, we find two efficient pruning rules based on properties of the frequent closed probability. To explain these pruning rules clearly, we first define the count of an itemset.

Definition 4.2 (*Count of An Itemset*). Given an itemset X and an uncertain transaction database *UTD*, the count of X is the numbers of transactions which contain X.

For example, to the same uncertain transaction database UTD in Table II, given itemsets {a,b,c,d}, {a,b,c,d}.count=2 because only T1 and T4 possibly contain {a,b,c,d}.

Lemma 4.2 (*Superset Pruning*). Given an itemset X whose size is |X|, an uncertain database *UTD*, and X's superset X+e_i, whose size is |X|+1, if e_i is smaller than at least one item in X with respect to the alphabetic order (namely, X is not the prefix of X+e_i by means of the alphabetic order), and X.count= X+e_i.count, we can get the following two results:

1) X's frequent closed probability is zero and X must not be probabilistic frequent closed itemset.

2) All supersets with X as prefix based on the alphabetic order must not be probabilistic frequent closed itemsets.

Proof. According to the given conditions, there is a superset of X, X+e_i, and X.count= X+e_i.count, X and X+e_i always appear together in all possible worlds. Thus, $\Pr_{FC}(X)=0$. In addition, for each superset of X, there must be at least one superset of X+e_i, which always appears with such superset of X together in all possible worlds. Thus, frequent closed probabilities of all supersets must also be 0. Furthermore, all supersets of X are not probabilistic frequent closed itemsets. So, the lemma holds. ■

We use the following example to further explain lemma 4.2.

Example 4.1 (*Superset Pruning*). In our running example shown in Table II, we know that $\{b,c\} \subset \{a,b,c\}$. Item a is smaller than b and c according to the alphabetic order, {a,b,c} is not a superset with {b,c} as prefix and {b,c}.count= {a,b,c}.count, so {b,c} and supersets with {b,c} as prefix must not be probabilistic frequent closed itemsets.

Moreover, besides the superset pruning, we propose another pruning technique, subset pruning, to further reduce search space.

Lemma 4.3 (*Subset Pruning*).Given an uncertain transaction database *UTD*, an itemset X, and X's subset X-e_i, e_i is the last item in X according to the alphabetic order. If X.count= X-e_i.count, we can get the following two results:

1) X-e_i's frequent closed probability is zero and X-e_i must not be probabilistic frequent closed itemset.

2) Itemsets which have the same prefix X-e_i, and the same size, |X|, must not be probabilistic frequent closed itemsets, supersets of such itemsets must not be probabilistic frequent closed itemsets.

Proof. We know X.count= X-e_i.count, X-e_i always appears together with X in all possible worlds. Thus, $\Pr_{FC}$(X-e_i)=0, and X-e_i is not probabilistic frequent closed itemset. Moreover, for each itemset which has the prefix X-e_i and the same size, |X|, there must be at least one superset of X, which always appears with this itemset with the prefix X-e_i together in all possible worlds. Thus, frequent closed probabilities of itemsets which have the same prefix X-e_i, and the same size, |X| are zero. So, these itemsets and their supersets are not probabilistic frequent closed itemsets. ■

We also use an example to further explain the lemma 4. 3.

Example 4.2 (*Subset Pruning*). In our running example shown in Table II, {a,b,c} and {a,b,d} have the same prefix of {a,b} and the same size. In addition, {a,b}.count= {a,b,c}.count, so {a,b} must not be probabilistic frequent closed itemset. In addition, {a,b,d} and their all supersets are also not probabilistic frequent closed itemsets.

3. Bounding Frequent Closed Probability

To avoid meaningless computation of the frequent closed probability, we adopt the upper bound and the lower bound of frequent closed probability to filter out the unqualified itemsets.

Lemma 4.4 (*Upper Bound and Lower Bound of Frequent Closed Probability*). Given an itemset X, an uncertain transaction database *UTD*, and *min_sup*, if there are *m* other items besides items in X, $e_1, e_2, ..., e_m$, the frequent closed probability of X, $\Pr_{FC}(X)$, satisfies:

$$\begin{cases} \Pr_{FC}(X) \le \Pr_F(X) - \sum_{i=1}^{m} \dfrac{\Pr(C_i)^2}{\Pr(C_i) + \sum_{j \ne i} \Pr(C_i \cap C_j)} \\ \Pr_{FC}(X) \ge \Pr_F(X) - \min\{\sum_{i=1}^{m} \Pr(C_i) - 2\sum_{i=2}^{m}\sum_{j=1}^{i-1} \Pr(C_i \cap C_j)/m,\ 1\} \end{cases}$$

where C_i represents the event that the superset of X, X+e_i, always appear together with X at least *min_sup* times.

Proof. We know that the frequent closed probability equals the corresponding frequent probability subtracting the frequent non-closed probability. Moreover the frequent non-closed probability can be expressed as Inclusive-Exclusive Principle form, namely, the frequent non-closed probability of X is denoted as $\Pr(C_1 \vee ... \vee C_m)$. According to the de Caen probability inequality [6] and the Kwerel probability inequality [21], we can get the following formulas,

$$\begin{cases} \Pr(C_1 \vee ... \vee C_m) \geq \sum_{i=1}^{m} \frac{\Pr(C_i)^2}{\Pr(C_i) + \sum_{j \neq i} \Pr(C_i \cap C_j)} \\ \Pr(C_1 \vee ... \vee C_m) \leq \min\{\sum_{i=1}^{m} \Pr(C_i) - 2\sum_{i=2}^{m}\sum_{j=1}^{i-1} \Pr(C_i \cap C_j)/m,\ 1\} \end{cases}$$

Thus, the frequent closed probability of X, $\Pr_{FC}(X)$, satisfies the following result:

$$\begin{cases} \Pr_{FC}(X) \leq \Pr_F(X) - \sum_{i=1}^{m} \frac{\Pr(C_i)^2}{\Pr(C_i) + \sum_{j \neq i} \Pr(C_i \cap C_j)} \\ \Pr_{FC}(X) \geq \Pr_F(X) - \min\{\sum_{i=1}^{m} \Pr(C_i) - 2\sum_{i=2}^{m}\sum_{j=1}^{i-1} \Pr(C_i \cap C_j)/m,\ 1\} \end{cases}$$

So, the lemma holds. ■

According to Lemma 4.4, we can further prune an itemset without computing its frequent closed probability if the upper bound of the frequent closed probability of the itemset is smaller than the given probabilistic frequent closed threshold.

4. An Approximate Algorithm of Computing Frequent Closed Probability

For each remaining itemset after all pruning methods, we have to compute its frequent closed probability. Because the computation of the frequent closed probability of an itemset is #P-Hard, we give a Monte-Carlo approximation algorithm to compute an approximate frequent closed probability of an itemset with a bounded error. The main idea of Monte-Carlo algorithm is to approximate a real value through a number of samplings, thus, it is most important for the approximation algorithm to select a suitable sampling method.

A naïve sampling method randomly samples a possible world of a given uncertain transaction database and discovers all frequent closed itemsets in this possible world. The process repeats N times. For each itemset, we use the ratio between the number of sampled possible worlds that the itemset is a frequent closed itemset and N as the approximate frequent closed probability of this itemset. Clearly, with the increasing of the number of samplings, the approximate frequent closed probability of each itemset gets closer to its real frequent closed probability. However, the shortcoming of the naïve sampling method is that we cannot know the exact number of samplings that we need to run before all samplings end.

To address this problem, we propose a Monte-Carlo approximation algorithm which satisfies FPRAS (fully polynomial randomized approximation scheme). Namely, given a relative tolerance error, ε, and a probabilistic confidence degree, δ, the approximate frequent closed probability of an itemset X, $\hat{\Pr}_{FC}(X)$, must satisfies $\Pr(|\hat{\Pr}_{FC}(X) - \Pr_{FC}(X)| < \varepsilon) \geq 1 - \delta$. Because we have already changed the problem of computing frequent non-closed probability of an itemset to the problem of computing the probability of the corresponding DNF formula, based on the coverage algorithm for DNF Counting problem [14], we can propose the following sampling solution.

Our sampling method: Given a minimum support, *min_sup*, an uncertain database, *UTD*, which has *n* transactions and *m* distinct items, and an itemset X which contains *k* items, the DNF formula will consisted of m-k events. We firstly accumulate each $Pr(C_i)$ to a real number Z as an upper bound of the frequent non-closed probability of X. Then, we repeat N samplings and define two variables of real number, U and V, which are initialized as zero. In each sampling, we sample an event C_i at first, and then continue to sample a possible world that must satisfy the event C_i and accumulate the probability of such possible world to V. Thus, V records the total probabilities of N samplings. Meanwhile, we still check whether the sampled possible world can satisfy two following conditions: 1) the possible world only satisfies the event C_i; 2) the possible world cannot satisfy i-1 other events from C_1 to C_{i-1}. If the sampled possible world satisfies above two requirements, we also accumulate the probability of the sampled possible world to U, which measures the contribution of each event, C_i, to the probability of the DNF formula. Finally, when N samplings end, we use UZ/V as the estimate of the approximate frequent non-closed probability, $\hat{\Pr}_{FNC}(X)$. The detailed steps of our algorithm to compute the frequent closed probability is listed in Fig. 2.

Procedure ApproxFCP {
Input: an itemset, X; an uncertain transaction database, UTD; an minimum support, min_sup; a relative tolerance error, ε; a probabilistic confidence degree, δ.
Output: an approximate frequent closed probability of X.
1: Building DNF formula $F = C_1 \vee C_2 \vee ... \vee C_n$ with respect to itemset X, and uncertain transaction database UTD;
2: $N \leftarrow \frac{4n}{\varepsilon^2} \ln \frac{2}{\delta}$;
3: $Z \leftarrow 0$;
4: **for** $i \leftarrow 1$ to n
5: Computing $\Pr(C_i) \leftarrow \prod_{j \in NT} (1 - \Pr(T_j)) \times \Pr_F(X + e_i)$
6: $Z \leftarrow Z + \Pr(C_i)$
7: $U \leftarrow V \leftarrow 0$;
8: **for** $loop \leftarrow 1$ to N
9: $i \leftarrow$ random sampling an integer $i \in \{1,2,...,n\}$;
10: $tp \leftarrow$ sampling a possible world which must satisfy C_i
11: $V \leftarrow V + \Pr(tp)$
12: **if** (tp cannot satisfy any C_j and $1 \leq j \leq i-1$)
13: $U \leftarrow U + \Pr(tp)$
14: $\hat{\Pr}_{FNC}(X) \leftarrow UZ/V$
15: **return** $\Pr_F(X) - \hat{\Pr}_{FNC}(X)$ }

Fig. 2 Approximate Computing Frequent Closed Probability

Time Complexity Analysis: According to the description of Procedure ApproxFCP, given the itemset, *X*, an uncertain transaction database, *UTD*; the number of all distinct items is m, then let $k=m-|X|$, we can derive the time complexity of Procedure ApproxFCP is $O(\frac{4k^2 \ln(2/\delta)|UTD|}{\varepsilon^2})$. Please note $|UTD|$ represents the number of transactions in *UTD*. Thus, the approximate algorithm needs a smaller number of samples (than the naïve algorithm) while guaranteeing that the estimated probability is precise enough.

C. The Depth-First Mining Algorithm

According to the framework in Fig. 1, after we get a candidate set of probabilistic frequent items based on Chernoff-Hoeffding bound-based pruning, we extend each possible itemset based on the depth-first search strategy. The Procedure, ProbFC, in Fig. 3 gives the details about the depth-first search-based mining algorithm, which seamlessly integrates the above discussed solutions. In addition, if the current itemset fails to be pruned, we will compute X's frequent closed probability to decide whether X is a probabilistic frequent closed itemset.

In ProbFC algorithm, we firstly decide whether current itemset, X, needs to grow by the Superset Pruning in lines 1-2. If the pruning holds, all supersets with X as prefix based on the alphabetic order need not be generated (based on *Lemma 4.2*). If the Superset Pruning fails to hold, we continue to generate the superset of X, $X+e_j$, where e_j is larger than the last item in X with respect to the alphabetic order. (Please note all items in X are ordered by the alphabetic order from small to large) Then, in lines 5-6, we decide whether $X+e_j$ is a probabilistic frequent itemset based on the Chernoff-Hoeffding bound-based Pruning. If $X+e_j$ is not probabilistic frequent, all supersets of $X+e_j$ need not be produced as well. If $X+e_j$ fails to be filtered out in line 5, we continue to execute the Subset Pruning to $X+e_j$. If Subset Pruning holds, we can get the following two results: 1) X must not be a probabilistic frequent closed itemset, 2) other supersets of X which have the same prefix X and the same size, |X|+1, need not be enumerated in lines 7-8. Please note that e_k is larger than e_j (based on *Lemma 4.3*). If $X+e_j$ cannot be filtered by above pruning strategies, in lines 9-10, we only compute the upper bound and the lower bound of the frequent closed probability of $X+e_j$. If the upper bound is smaller than user-given threshold or the upper bound equals the lower bound, the frequent closed probability of $X+e_j$ need not be computed. Then, we do not directly compute the frequent closed probability of $X+e_j$ because $X+e_j$ still might be pruned by its superset. Hence, we continue to recursively call the function, ProbFC, to enumerate supersets of $X+e_j$ based on depth-first search strategy in line 11. Finally, if $X+e_j$ fails to be pruned, we have to execute the steps in lines 12-14. In line 12, the Monte-Carlo–based approximate algorithm, ApproxFCP, is called to compute an approximate frequent closed probability. Then, we check whether X is a probabilistic frequent closed itemset based on its approximate frequent closed probability in lines 13-14.

Procedure ProbFC {

Input: an itemset, X, an ordered candidate set of probabilistic frequent items, *Cand*, a set of probabilistic frequent closed itemsets, *PFC*, an uncertain transaction database, *UTD*; an minimum support, *min_sup*; an probabilistic frequent threshold, *pfct*; a relative tolerance error, ε; a probabilistic confidence degree, δ.

Output: an approximate result set of probabilistic frequent closed itemsets.

1: **if** Superset Pruning of X is true **then**
2: break;
3: i ← the order of last item in X in Cand by alphabetic order
4: **for** each $e_j \in$ Cand ($j \in [i+1, |Cand|]$) **do**
5: **if** Chernoff-Hoeffing bound Pruning of $X+e_j$ is true **then**
6: break;
7: **if** Subset Pruning of $X+e_j$ is true **then**
8: marking X is not a probabilistic frequent closed itemset and itemsets, $X+e_k$ ($k \in [j+1, |Cand|]$), need not grow.
9: **if** Upper Bound Checking of $X+e_j$ is true **then**
10: break;
11: **Call** ProbFC ($X+e_j$, Cand, PFC, UTD, min_sup, pfct, ε, δ)
12: $Pr_{FC}(X+e_j) \leftarrow$ **Call** ApproxFCP ($X+e_j$, Cand ,UTD, ε, δ)
13: **if** ($Pr_{FC}(X+e_j) \geq pfct$) **then**
14: PFC ← PFC $\bigcup$ $X+e_j$ }

Fig. 3 Algorithm of ProbFC

We further use an example to show how to find all probabilistic frequent closed itemsets using ProbFC.

Example 4.3 (*Depth-First Search-based Mining Algorithm*). For the uncertain database *UTD* in Table II with *min_sup*=2, probabilistic frequent closed threshold, *pfct*=0.8. The process of mining probabilistic frequent closed itemsets in *UTD* is shown in Fig. 4.

1. Find a candidate set of probabilistic frequent items. Based on Chernoff-Hoeffding bound-based pruning, we get a candidate set of probabilistic frequent items, Cand={a,b,c,d}.
2. Discover all probabilistic frequent closed itemsets
   2.1. {a} passes the test of Chernoff-Hoeffding bound-based pruning, and {a}.count={ab}.count, so we need not grow the paths of {ac} and {ad}, based on the Subset Pruning.
   2.2. Similar to step 2.1, the Subset Pruning avoids growing {abd} and their supersets. In the depth-first search of {a} as prefix, we can finally obtain two probabilistic frequent closed itemset {abc} and {abcd}. Their frequent closed probabilities can be computed by our approximate algorithm marked by grey in Fig. 4.
   2.3. After finishing all enumeration with prefix {a}, we continue to test itemsets with other prefix. According to Superset Pruning, we can stop growing any itemsets. Namely, the whole enumeration process ends.

2.4. Based on the above recursive function call, we get the result set: {abc, fcp: 0.875}, {abcd, fcp: 0.81}, where {itemset, fcp:value} denotes the itemset and the value of this frequent closed probability, respectively.

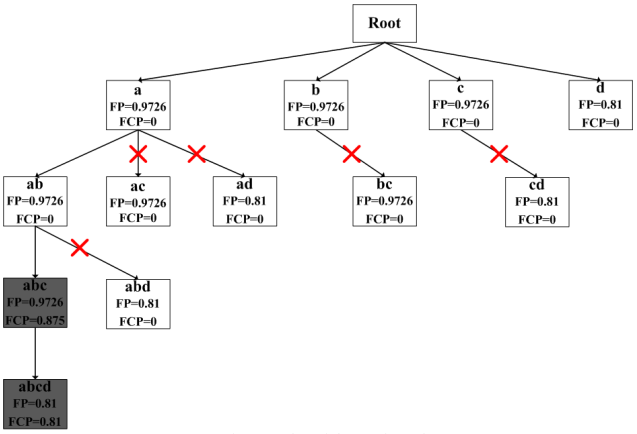


Fig. 4 Algorithm of ProbFC

TABLE VII
INDIVIDUAL FEATURES OF ALGORITHMS IN OUR EXPERIMENTS

Algorithm	CH	Super	Sub	PB	Framework
MPFCI	√	√	√	√	DFS
MPFCI-NoCH		√	√	√	DFS
MPFCI-NoBound	√	√	√		DFS
MPFCI-NoSuper	√		√	√	DFS
MPFCI-NoSub	√	√		√	DFS
MPFCI-BFS	√			√	BFS

TABLE VIII
CHARACTERISTICS OF DATASETS

Dataset	Number of Transaction	Number of Items	Average Length	Maximal Length
Mushroom	8124	120	23	23
T20I10D30KP40	30000	40	20	40

V. EXPERIMENTAL RESULTS

In this section, we report our experimental study on the performances of MPFCI and effectiveness of pruning strategies. We compare MPFCI with other four algorithms: MPFCI without Chernoff-Hoeffding bound pruning, (MPFCI-NoCH), MPFCI without superset pruning (MPFCI-NoSuper), MPFCI without subset pruning (MPFCI-NoSub) and MPFCI without probabilistic bound of the closed probability pruning (MPFCI-NoBound). Moreover, we verify the efficiencies of different search strategies, depth-first search and breadth-first search. We implemented a breadth-first search-based algorithm, MPFCI-BFS, which utilizes the Chernoff-Hoeffding bound-based pruning and the frequent closed probability bound-based pruning. However, MPFCI-BFS cannot use superset pruning and subset pruning because they won't show up in BFS's enumeration, which nullifies checking on ensuing pruning conditions. Finally, we also report the compression quality of mining probabilistic frequent closed itemsets. More details of six algorithms are listed in Table VII. Table VII has five columns where CH means the Chernoff-Hoeffding bound pruning, Super means the superset pruning, Sub means the subset pruning, PB means the frequent closed probability-based pruning, and Framework means the algorithm framework used by different algorithms. All the experiments are performed on an Intel(R) Core(TM) i5 2.40GHz Dell laptops with 4GB main memory, running on Microsoft Windows 7. All the programs are implemented in Visual C++ 2010.

Since real uncertain data sets are not available, in our experiments, we follow the experimental method adopted by the previous work [22] and generate probabilistic datasets from a real certain dataset and a synthetic certain dataset by assigning a probability generated from Gaussian distribution to each transaction. The real certain dataset is Mushroom which contains characteristics of various species of mushrooms. The synthetic dataset, T20I10D30KP40 is generated from IBM dataset generator [5], with an average transaction length 20 and an average itemset length 10 [2]. The synthetic dataset contains 30000 transactions and 40 unique items [2]. The characteristics of above two datasets are shown in Table VIII. In addition, to verity the influence of uncertainty, we test two scenarios. The first scenario is that a dataset follows Gaussian distribution with low mean and high variance, namely probabilities of transactions are similar and close to each other. We make the Mushroom follows Gaussian distribution whose mean is 0.5 and variance is 0.25. Another scenario is that a dataset follows Gaussian distribution with high mean and low variance. In our experiment, we make T20I10D30KP40 follow Gaussian distribution with mean 0.8 and variance 0.1.

A. *Efficiency and Effectiveness of Pruning Strategies*

In this subsection, we report the efficiency of algorithms and effectiveness of pruning strategies. There are 4 parameters in our mining algorithm, MPFCI, which are minimum support, min_sup, probabilistic frequent closed threshold, pfct, relative tolerance error, ε, and the probabilistic confidence degree parameter, δ. We test influences of each parameter in terms of total running time and effectiveness of pruning methods. We set default values to each parameter as follows. The *min_sup* is set to 0.4 in Mushroom dataset and 0.3 in T20I10D30KP40 dataset (which is the median of our experiment about *min_sup*). We set pfct to 0.8 since results with low probability are usually meaningless. ε and δ (the probabilistic confidence degree is 1-δ) are set to 0.1 because large error also leads to meaningless results. When a parameter changes, other three parameters are fixed to their default values. In addition, we did not report the running times over 1 hour if other algorithms can finish the same task within 1 hour.

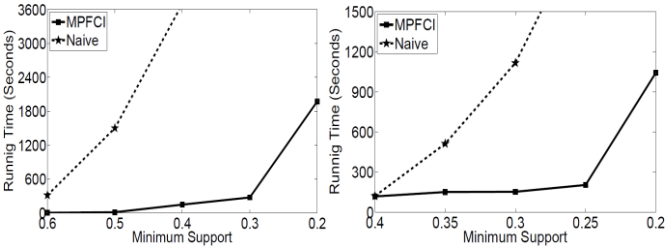
Firstly, we test the efficiency of our algorithm w.r.t. the running time compared with a naive method. Fig 5 (a) shows comparison of running time between Naive and our algorithm, MPFCI, while min_sup increase from 0.2 to 0.6 in Mushroom dataset. Here, the naive method, called Naive, directly use our approximation algorithm to compute frequent closed probability one by one after obtaining all probabilistic frequent itemsets based on TODIS algorithm [22]. We can observe that the running time of the Naive algorithm exceed 1

hour after the min_sup becomes smaller than 0.4. However, the MPFCI algorithm can find all probabilistic frequent closed itemsets efficiently. This explains that the computational cost of directly testing frequent closed probability increases sharply with the increase of the number of probabilistic frequent itemsets (The number increases w.r.t the increase of min_sup). Similar results can be found in Fig. 5(b) which also reports comparison of running time in T20I10D30KP40 dataset. Hence, it is inefficient to directly check frequent closed probabilities of all probabilistic frequent itemsets which can be obtain by existing solution, such as TODIS [22].

Next, we focus on the test of effectiveness of different pruning strategies.

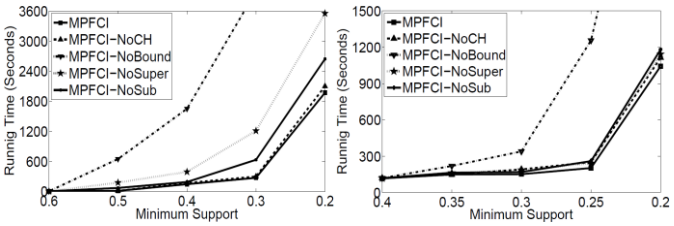
Effect of min_sup in MPFCI. We firstly investigated the running time of five compared algorithms w.r.t. the minimum support, min_sup. Fig. 6(a) shows the running time of all algorithms while min_sup changes from 0.2 to 0.6 in Mushroom dataset. When min_sup decreases, we observe that the running time of all the algorithms goes up due to the number of probabilistic frequent itemsets increases. However, we find that the growth speed of MPFCI is the lowest due to all pruning methods it employed. We also find that MPFCI-NoCH has the similar running time with that of MPFCI. This indicates that the Chernoff-Hoeffding bound pruning contributes less to reduce the total running time compared to other pruning methods. In addition, MPFCI-NoBound is the slowest algorithm. In particular the running time of MPFCI-NoBound is larger than 1 hour when min_sup is smaller 0.3. Similar results can be found in Fig. 6(b) which reports the running time of all algorithms in T20I10D30KP40 dataset.

Effect of pfct in MPFCI. We continue to test the running time of five compared algorithms with varying the probabilistic frequent closed threshold, pfct in two different datasets. In Fig. 7 (a) and (b), we can obverse that MPFCI is always the fastest algorithm. With regards to the change of pfct, the running time of all algorithms remains approximately the same. Thus, in tuple-model uncertain transaction database, we can discover that the influence of probabilistic frequent closed threshold will be smaller than that of min_sup to the total running time. Meanwhile, MPFCI is the fastest algorithm, and MPFCI-NoBound is still the slowest one, which indicates that the probabilistic bound pruning is more effective than other pruning methods.



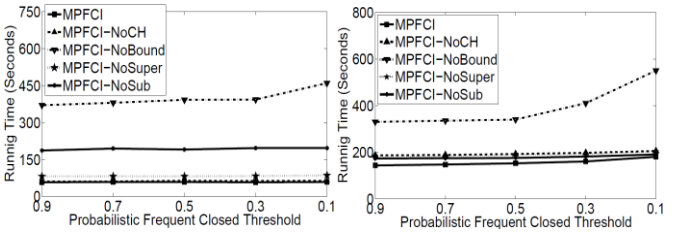
(a) *Mushroom* (b) *T20I10D30KP40*

Fig. 5 Efficiency Comparison between MPFCI and Naïve



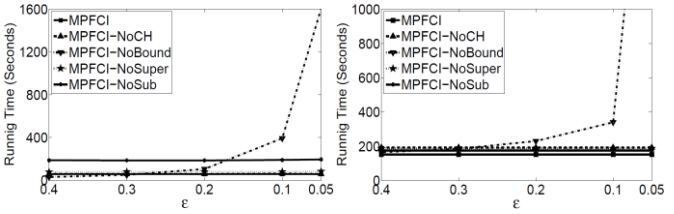
(a) *Mushroom* (b) *T20I10D30KP40*

Fig. 6 Running time w.r.t min_sup



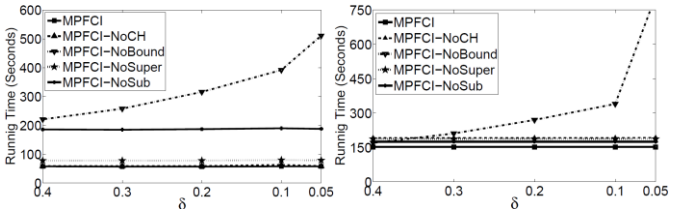
(a) *Mushroom* (b) *T20I10D30KP40*

Fig. 7 Running time w.r.t pfct



(a) *Mushroom* (b) *T20I10D30KP40*

Fig. 8 Running time w.r.t ε



(a) *Mushroom* (b) *T20I10D30KP40*

Fig. 9 Running time w.r.t δ

Effect of ε in MPFCI. In this part, we mainly analyse the influence of the relative tolerance error, ε, to the running time in Fig. 8 (a) and (b). With the change of ε, we can observe: 1) four algorithms, MPFCI, MPFCI-NoCH, MPFCI-NoSuper and MPFCI-NoSub, keep a steady running time. 2) The algorithm of MPFCI-NoBound becomes slower clearly when ε becomes lower. In addition, the above observations also verify the time complexity analysis of the approximation algorithm to compute frequent closed probability. Because the MPFCI-NoBound algorithm does not use any probabilistic bound pruning to reduce redundant computation for frequent closed probability, we have to compute closed probabilities of all itemsets which cannot be removed by other pruning methods. Please note that the time complexity of our approximation algorithm to compute frequent closed probability is $O(4k\ln(2/\delta)|UTD|/\varepsilon^2)$. Moreover, the running time of approximation algorithm will increase $O(1/\varepsilon^2)$ with the decreasing of ε. Thus, the decreasing of ε causes more influence to the running time of the MPFCI-NoBound

algorithm. Moreover, other four algorithms utilize the probabilistic bounds pruning, so they can avoid redundant computation of frequent closed probabilities.

Effect of δ in MPFCI. We finally studied the influence of running time from the probabilistic confidence degree parameter, δ, in Fig. 9 (a) and (b). We can find the similar effect with the change of ε. Namely, the MPFCI-NoBound algorithm is also affected greatly by the change of δ. By comparing the results in Fig. 8 and Fig. 9, we can discover that the influence of δ on the running time of MPFCI-NoBound is smaller than the influence of ε. This is because with decreasing of δ, the running time of MPFCI-NoBound about computing frequent closed probability will increase $O(\ln(2/\delta))$.

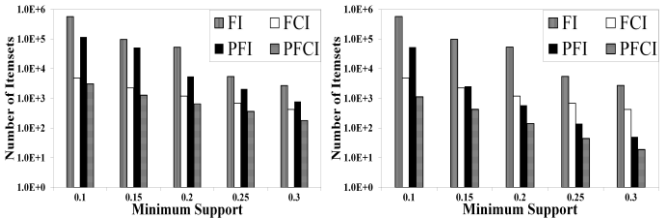
B. Compression Quality Evaluation

In this subsection, we mainly evaluate the compression quality of probabilistic frequent closed itemsets in probabilistic data against that of frequent closed itemsets in exact data. Without loss of generality, we test the compression quality under two different distributions over Mushroom dataset. The first one makes the Mushroom follows Gaussian distribution whose mean is 0.8 and variance is 0.1. The second one follows Gaussian distribution whose mean is 0.5 and variance is 0.25 (Namely, the default set of the Mushroom dataset in our experiments). In addition, we keep same default values of pfct=0.8, ε =0.1, and δ =0.1 as used in the last subsection.

We compare the number of itemsets of frequent itemsets (for short FI), frequent closed itemsets (for short FCI), probabilistic frequent itemsets (for short PFI), and probabilistic frequent closed itemsets (for short PFCI). The first two kinds of itemsets are produced by FP-growth algorithm [13] and Closet+ algorithm [24] from the exact Mushroom dataset. The PFI and PFCI are generated by TODIS algorithm [22] and our algorithm respectively.

Fig. 10 (a) shows the comparison of the number of above four kinds of itemsets under the uncertain Mushroom dataset whose mean is 0.8 and variance is 0.1. With min_sup decreasing from 0.3 to 0.1, we observe that the ratio between Number of FCI and Number of FI declines sharply. Meanwhile, the ratio between Number of PFCI and Number of PFI also declines quickly. That means the probabilistic frequent closed itemsets over uncertain data have the similar compression effect with frequent closed itemsets in exact data. Fig. 10 (b) also shows the similar compression result. However, we also observe that the compression effect of probabilistic frequent closed itemsets in Fig. 10 (b) is worse than that in Fig. 10 (a). That is because Fig. 10 (b) has the smaller mean than Fig. 10 (a) and has the higher variance than Fig. 10 (a). The smaller mean and higher variance indicate higher uncertainty, which leads to less number of probabilistic frequent itemsets and probabilistic frequent closed itemsets under same threshold.

To sum up, above results demonstrate that our definition of probabilistic frequent closed itemsets keeps the good compression property of frequent closed itemsets in exact data.



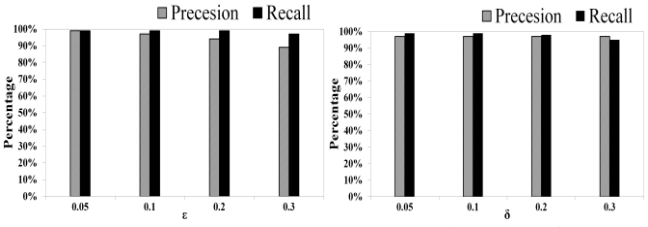
(a) Mean=0.8, Var=0.1 (b) Mean=0.5, Var=0.25

Fig. 10 Compression Quality w.r.t min_sup

C. Approximation Quality Evaluation

In this subsection, we evaluate the approximation quality of our mining algorithm since the problem of computing frequent closed probability has been proven as a #P-Hard problem. Because approximation effect of our algorithm is determined by the relative tolerance error, ε, and the probabilistic confidence degree parameter, δ, we mainly focus on testing the change of the result set w.r.t the change of ε and δ over our default uncertain Mushroom dataset. We denote the set of the final result as FR and the set of true probabilistic frequent closed itemsets as TI. We use *precision* which equals $\frac{|FR \cap TI|}{|FR|}$ and *recall* which equals $\frac{|FR \cap TI|}{|TI|}$ to measure the approximate effect. Moreover, it is hard to get the set of true probabilistic frequent closed itemsets due to #P-hard complexity. Thus, we use the result set when ε=0.01 and δ=0.01 as true result set.

Fig. 11 (a) shows that the change of precision and recall when ε increases from 0.05 to 0.3 and δ equals 0.1. We observe that the recall keeps steady, but precision decreases with ε increasing. Because the number of sampling in the ApproxFCP algorithm equals $\frac{4n}{\varepsilon^2}\ln\frac{2}{\delta}$, the number of sampling will decrease significantly with ε increasing. Moreover, δ equals 0.1, probabilistic confidence degree is also fixed. Thus, the number of true probabilistic frequent closed itemsets keeps steady and recall also remain stable. Similarly, Fig. 11 (b) shows the case where δ increases from 0.05 to 0.3 and ε equals 0.1. Because δ only influences the number of sampling on $\ln 2/\delta$ times, recall is affected slightly and precision also remains stable. To sum up, above results demonstrate that our approximation algorithm has a good approximation quality.



(a) Varying ε (b) Varying δ

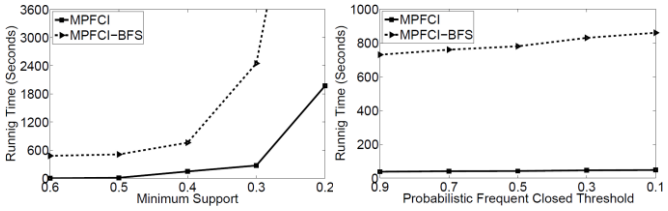
Fig. 11 Approximation Quality in Mushroom

D. Algorithm Framework Evaluation

In this subsection, we evaluate the efficiencies of the depth-first search-based and breadth-first search-based strategies.

Fig. 12(a) shows the running time of two search solutions while min_sup varies from 0.2 to 0.6 and other parameters are

default values. Clearly, MPFCI-BFS spends much more time than MPFCI does under the same parameters. In particular, when min_sup is smaller, the growth rate of running time of MPFCI-BFS is much faster than that of MPFCI. This is because MPFCI-BFS lacks of effective pruning as superset/subset pruning. Fig. 12(b) shows the running time of two algorithms while pfct varies from 0.1 to 0.9 and other parameters are default values. The running time of MPFCI-BFS has slow growth with the decrease of pfct and is more than the running time of MPFCI. This is because that MPFCI-BFS also includes Chernoff-Hoeffding bound-based pruning and such pruning mainly reduces the influence of the pfct to running time. So, the change of running time of MPFCI-BFS under pfct varying is smaller than the change under min_sup varying. We also tested other parameters; all results show that MPFCI is more efficient than MPFCI-BFS. This is because that Superset pruning and Subset pruning are effective pruning methods and MPFCI-BFS has to traverse much more search space. Due to space limit, we didn't show these results in here.



(a) min_sup varying (b)pfct varying

Fig. 12 Running time of MPFCI and MPFCI-BFS w.r.t different parameters

VI. CONCLUSIONS

In this paper, we propose a new problem of mining probabilistic threshold-based frequent closed itemsets in an uncertain transaction database. We prove that even a sub-problem of this problem, computing the frequent closed probability of an itemset, is #P-Hard. Due to the computational complexity of this problem, an effective and efficient depth-first search-based mining algorithm has been designed in order to obtain all probabilistic frequent closed itemsets. Moreover, several probabilistic pruning techniques are also designed to reduce search space and to avoid many complicated computations. Extensive experimental results show the effectiveness and efficiency of the mining algorithm.

ACKNOWLEDGMENT

The work described in this paper was partially supported by Hong Kong RGC GRF Project No. 611411; National Grand Fundamental Research 973 Program of China under Grant 2012CB316200.

REFERENCES

[1] C. Aggarwal, Y. Li, J. Wang, J. Wang, "Frequent pattern mining with uncertain data," in *KDD*, 2009.

[2] R. Agrawal, T. Imielinski, A. N. Swami, "Mining association rules between sets of items in large databases," in *SIGMOD*, 1993.

[3] R. Agrawal, R. Srikant, "Fast algorithms for mining association rules," in *VLDB*, 1994.

[4] T. Bernecker, H.-P. Kriegel, M. Renz, F. Verhein, A. Zuefle, "Probabilistic frequent itemset mining in uncertain databases," in *KDD*, 2009.

[5] C. Böhm, A. Pryakhin, M. Schubert, "The Gauss-tree: efficient object identification in databases of probabilistic feature vectors," in *ICDE,* 2006.

[6] D. de Caen, "A lower bound on the probability of a union," *Discrete Math.,* 169: 217-220, 1997.

[7] T. Calders, C. Garboni, B. Goethals, "Efficient pattern mining of uncertain data with sampling," in *PAKDD*, 2010.

[8] L. Chen, M. T. Özsu, V. Oria, "Robust and fast similarity search for moving object trajectories," in *SIGMOD*, 2005.

[9] C. Chui, B. Kao, E. Hung, "Mining frequent itemsets from uncertain data," in *PAKDD*, 2007.

[10] C. Chui, B. Kao, "A decremental approach for mining frequent itemsets from uncertain data," in *PAKDD*, 2008.

[11] B. Ding, D. Lo, J, Han, S. C. Khoo, "Efficient mining of closed repetitive gapped subsequences from a sequence database," in *ICDE*, 2009.

[12] G. Grahne and J. Zhu, "Fast algorithms for frequent itemset mining using fp-trees," *IEEE Trans. on Know. and Data Eng*., 17(10): 1347-1362, 2005.

[13] J. Han, J. Pei, Y. Yin, "Mining frequent patterns without candidate generation," in *SIGMOD*, 2000.

[14] R.M. Karp, M. Luby, "Monte-Carlo algorithms for enumeration and reliability problems," in *FOCS*, 1983.

[15] C. Leung, M. F. Mateo, D. A. Brajczuk, "A tree-based approach for frequent pattern mining from uncertain data," in *PAKDD*, 2008.

[16] M. Li, Y. Liu, "Underground coal mine monitoring with wireless sensor networks," *ACM Trans. on Sensor Networks,* 5(2): 10-29, 2009.

[17] L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X. Li, G. Dai, "Canopy closure estimates with GreenOrbs: sustainable sensing in the forest," in *SenSys*, 2009.

[18] N. Pasquier, Y. Bastide, Taouil R, Lakhal, "Discovering frequent closed itemsets for association rules," in *ICDT*, 1999.

[19] J. Pei, J. Han, R. Mao, "CLOSET: An efficient algorithm for mining frequent closed itemsets," in *DMKD*, 2000.

[20] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang, D. Yang, "H-Mine: Hyper-structure mining of frequent patterns in large databases," in *ICDM*, 2001.

[21] Y.S. Sathe, M. Pradhan, S.P. Shah, "Inequalities for the probability of the occurrence of at least m out of n events," *J. Appl. Probab.*, 17: 1127-1132, 1980.

[22] L. Sun, R. Cheng, D.W. Cheung, J. Cheng, "Mining uncertain data with probabilistic guarantees," in *KDD*, 2010.

[23] L. Wang, R. Cheng, S. Lee, D. Cheung, "Accelerating probabilistic frequent itemset mining: a model-based approach," in *CIKM*, 2010.

[24] J. Wang, J. Han, J. Pei, "CLOSET+: Searching for the best strategies for mining frequent closed itemsets," in *KDD*, 2003.

[25] J. Wang, J. Han, "BIDE: Efficient mining of frequent closed sequences," in *ICDE*, 2004.

[26] X. Yan, J. Han, "CloseGraph: mining frequent closed graph patterns," In *KDD*, 2003.

[27] X. Yan, J. Han, R. Afhar, "CloSpan: Mining closed sequential patterns in large datasets," in *SDM*, 2003.

[28] M. J. Zaki, "Scalable algorithms for association mining," *IEEE Trans. on Know. and Data Eng.*, 42(1/2): 31-60, 2001.

[29] M. J. Zaki, C-J. Hsiao, "Efficient algorithms for mining closed itemsets and their lattice structure," *IEEE Trans. on Know. and Data Eng.*, 17(4): 462-478, 2005.

[30] Q. Zhang, F. Li, K. Yi, "Finding frequent items in probabilistic data," in *SIGMOD*, 2008.

[31] Z. Zou, J. Li, H. Gao, S. Zhang, "Mining frequent subgraph patterns from uncertain graph data," *IEEE Trans. on Know. and Data Eng.*, 22(9): 1203-1218, 2010.

[32] Z. Zou, J. Li, H. Gao, "Discovering probabilistic frequent subgraphs over uncertain graph databases," in *KDD*, 2010.

[33] L. Valiant, "The complexity of enumeration and reliability problems," *SIAM Journal on Computing*, 8(3): 410-421, 1979.

[34] P. Tang, E. A. Peterson, "Mining probabilistic frequent closed itemsets in uncertain databases," in *ACM Southeast Conference*, 2011.

[35] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association*, 58 (301): 13–30, 1963.