Summarizing Data with Representative Patterns

Chunyang Liu
Faculty of Engineering and Information Technology
University of Technology, Sydney

A thesis submitted for the degree of

Doctor of Philosophy

March 2016

To my parents and my wife.

Certificate of Original Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Student: Chunyang Liu

Date: 05/03/2016

Acknowledgements

I benefited and learned a lot from my advisors, my friends, my colleagues and my family during PhD study in University of Technology, Sydney. I would like to take this good opportunity to appreciate their significant helps to me.

First of all, I would like to express my appreciation and gratitude to my academic supervisors Dr. Ling Chen and Prof. Chengqi Zhang. I benefited significantly from various discussions and communications with them. They always give me advices, encouragement, and sufficient freedom to think and explore. I also want to thank them for kindly offering me invaluable suggestions and experienced instructions on my research career and life. I cannot image how this thesis can be accomplished without their high scientific criterion, endless patience, generous support, and constant guidance.

Next, I wish to thank other researchers who have gave me helpful guidance and encouragement during my PhD study and on conferences: Prof. Jian Pei, A/Prof. Ivor W. Tsang, Dr. Lu Qin, and Dr. Yi Yang. From discussions with them I learned not only much knowledge but also their attitude for doing high quality research.

I have been fortunate to work in a center gathering the most brilliant researchers and best friends in the past four years: Guodong Long, Jing Jiang, Wei Bian, Tianyi Zhou, Meng Fang, Bozhong Liu, Zhe Xu, Mingming Gong, Ruxin Wang, Zhibin Hong, Sujuan Hou, Wei Wu, Haishuai Wang, Xueping Peng, Alan Wang, Zhenxing Qin, Shirui Pan, Jia Wu, Lianhua Chi, Maoying Qiao, Tongliang Liu, Weiwei Liu, Anjin Liu, Junfu Yin, Jinjiu Li, Changxing Ding, Nannan Wang, Naiyang Guan, Li Wan, Shengzheng

Wang, Xianhua Zeng, Zhijing Xu, Bo Du, Hongshu Chen, Ting Guo, Chao Ma, Shaoli Huang, Qiang Li, Dianshuang Wu, Zhiguo Long, Weilong Hou, Yi Ji, Ming Xie and many others. I enjoyed the invaluable friendships with them, their kindly support and accompany are always my source of strength and courage in both research and daily life. I am also grateful to my dearest friends Jiawang Liu, Weipeng Zhang, Mingjian Gao, Junfeng Ye, and Jia Chen since my college, and Sen Li and Song Li since my high school. They are the ones who have given me support during both joyful and stressful times, to whom I will always be thankful.

Finally, it is my greatest honor to thank my family: my parents and my wife. They always believe in me, encourage me, give me invaluable suggestions, and fully support all my decisions. No words could possibly express my deepest gratitude for their endless love, self-sacrifice and unwavering help. To them I dedicate this dissertation.

Abstract

The advance of technology makes data acquisition and storage become unprecedentedly convenient. It contributes to the rapid growth of not only the volume but also the veracity and variety of data in recent years, which poses new challenges to the data mining area. For example, uncertain data mining emerges due to its capability to model the inherent veracity of data; spatial data mining attracts much research attention as the widespread of location-based services and wearable devices. As a fundamental topic of data mining, how to effectively and efficiently summarize data in this situation still remains to be explored.

This thesis studied the problem of summarizing data with representative patterns. The objective is to find a set of patterns, which is much more concise but still contains rich information of the original data, and may provide valuable insights for further analysis of data. In the light of this idea, we formally formulate the problem and provide effective and efficient solutions in various scenarios.

We study the problem of summarizing probabilistic frequent patterns over uncertain data. Probabilistic frequent pattern mining over uncertain data has received much research attention due to the wide applicabilities of uncertain data. It suffers from the problem of generating an exponential number of result patterns, which hinders the analysis of patterns and calls for the need to find a small number of representative patterns to approximate all other patterns. We formally formulate the problem of *probabilistic representative frequent pattern (P-RFP)* mining, which aims to find the minimal set of patterns with sufficiently high probability to represent all other patterns. The bottleneck turns out to be checking whether a pattern can probabilistically represent another, which involves the computation of a joint probability of the supports of two patterns. We propose a novel dynamic programming-based approach to address the problem and devise effective optimization strategies to improve the computation efficiency.

To enhance the practicability of P-RFP mining, we introduce a novel approximation of the joint probability with both theoretical and empirical proofs. Based on the approximation, we propose an *Approximate P-RFP Mining* (APM) algorithm, which effectively and efficiently compresses the probabilistic frequent pattern set. The error rate of APM is guaranteed to be very small when the database contains hundreds of transactions, which further affirms that APM is a practical solution for summarizing probabilistic frequent patterns.

We address the problem of directly summarizing uncertain transaction database by formulating the problem as *Minimal Probabilistic Tile Cover* Mining, which aims to find a high-quality probabilistic tile set covering an uncertain database with minimal cost. We define the concept of *Probabilistic Price* and *Probabilistic Price Order* to evaluate and compare the quality of tiles, and propose a framework to discover the minimal probabilistic tile cover. The bottleneck is to check whether a tile is better than another according to the *Probabilistic Price Order*, which involves the computation of a joint probability. We prove that it can be decomposed into independent terms and calculated efficiently. Several optimization techniques are devised to further improve the performance.

We analyze the problem of summarizing co-locations mined from spatial databases. Co-location pattern mining finds patterns of spatial features whose instances tend to locate together in geographic space. However, the traditional framework of co-location pattern mining produces an exponential number of patterns because of the downward closure property, which makes it difficult for users to understand, assess or apply the huge number of resulted patterns. To address this issue, we study the problem of mining *representative co-location patterns* (RCP). We first define a covering relationship between two co-location patterns then formally formulate the problem of *Representative Co-location Pattern mining*. To solve the problem of RCP mining, we propose the *RCPFast* algorithm adopting the post-mining framework and the *RCPMS* algorithm pushing pattern summarization into the co-location mining process.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

With the explosive growth of data in the past decades, *Data Mining* is increasingly significant and becomes a dynamic and promising research field. As defined in [1], "Data mining is the process of discovering interesting patterns and knowledge from large amounts of data." It has been widely applied in various domains, including healthcare, economics, and sociology.

The newly advances of technologies, such as social network services, location-based services, and wearable devices, make data acquisition unprecedentedly convenient, which brings both opportunities and challenges to the research community. On one hand, the rapid growth of the volume of data enables innovative applications such as recommender systems and customer behaviour analysis. On the other hand, the increasing veracity and variety of data poses new challenges to the analysis and evaluation of data.

As an important problem in data mining area, data summarization has been extensively studied in the past decades. Nevertheless, how to conduct data summarization in the new scenarios still remain to be explored. In this dissertation, we focus on summarizing data with representative patterns over uncertain databases and spatial databases. We propose novel definitions and algorithms, and investigate both the theoretical and empirical performance of the algorithms. In the remainder of this chapter, we first provide the background of uncertain database and spatial database, then give a brief overview of the proposed algorithms.

Transaction ID	Items
t_1	$a\ b$
t_2	a

(a) A deterministic database.

Transaction ID	Items
t_1	a:0.6 b:0.8
t_2	a:1.0 c:0.42

(b) An uncertain database.

Table 1.1: Example transaction databases.

1.1 Uncertain Data

Uncertain data mining emerges in the past decades because uncertainty is inherent in various applications such as sensor network monitoring, moving object tracking, and protein-protein interaction data [2]. It could be induced by different reasons including experimental error, artificial noise, and data incompleteness. Rather than cleaning the uncertain data using domain-specific rules, modeling the uncertainty of data is more rational in many applications, such as medical diagnosis and risk assessment. As a consequence, data mining over uncertain data has become an active research area recently.

Table 1.1 gives a comparison between a traditional deterministic database and an uncertain database. Table 1.1 (a) is a deterministic database, in which each transaction is a pair comprising an ID and an itemset, such as $\langle t_1, \{a, b\}\rangle$ and $\langle t_2, \{a\}\rangle$. Table 1.1 (b) is an uncertain database, in which each item is associated with a probability to indicate its existence in the transaction, e.g., the probability of item a occurring in transaction t_1 is 0.6.

Frequent pattern mining, which is one of the most fundamental data mining tasks, has been introduced into uncertain databases [3] and received a great deal of research attention [4, 5, 2, 6, 7, 8, 9]. Generally, two different definitions of frequent patterns exist in the context of uncertain data: *expected support-based frequent patterns* [3, 7], and *probabilistic frequent patterns* [4, 5]. Both definitions consider the *support* of a pattern as a discrete random variable. The former uses the expectation of the support as the measurement, while the latter considers the probability that the support of a pattern is no less than some specified minimum support threshold. Various algorithms have been designed to mine frequent patterns from uncertain data. A summarization and comparison of probabilistic frequent pattern mining algorithms has been reported [2].

Note that, the anti-monotonic property holds for both the expected support-based frequent patterns, as well as the probabilistic frequent patterns. That is, if a pattern is frequent in an uncertain database, then all of its sub-patterns are frequent as well. This property leads to the generation of an exponential number of result patterns. The large number of discovered frequent patterns makes the understanding and further analysis of generated patterns troublesome. Therefore, it is important to find a concise and informative representative pattern set to best approximate all other patterns.

Some initial research work has been undertaken to find a small set of representative patterns. For example, mining *probabilistic frequent closed patterns* over uncertain data has been studied in [10, 11, 12]. However, the number of probabilistic frequent closed patterns is still large because of the restrictive condition for a pattern being *closed*. For instance, in [12], the *closed probability* of a pattern is computed as the sum of the probabilities of the possible worlds of an uncertain database where the pattern is closed. We aim to relax the restrictive condition to further reduce the size of frequent patterns mined over uncertain data.

1.2 Spatial Data

Spatial database attracts increasingly research attention because of the popularization of location acquisition techniques, such as smart mobile phones and wearable devices. As one of the most fundamental tasks in spatial data mining, *co-location mining* aims to discover co-location patterns where each is a group of spatial features whose instances are frequently located close to each other [13]. Spatial co-location patterns yield important insights for various applications. In epidemiology, for example, different incidents of diseases may exhibit co-location patterns such that one type of disease tends to occur in spatial proximity of another [14]. In ecology, scientists are interested in finding frequent co-occurrences among spatial features, such as drought, substantial increase/drop in vegetation, and extremely high precipitation [15]. In e-commerce, companies may be interested in discovering types of services (e.g., weather, timetabling, and ticketing queries) that are requested by geographically neighboring users, so that location-sensitive recommendations can be provided [16]. Due to its importance, the problem of finding prevalent co-location patterns from spatial data has been explored extensively [13, 17, 18, 14, 19, 20, 21].

A common framework of co-location pattern mining uses the frequencies of a set of spatial features participating in a co-location to measure the prevalence (known as *participation index* [13]) and requires a user-specified minimum threshold to find interesting patterns. Typically, if the threshold is high, the framework may generate commonsense patterns. However, with a low threshold, a great number of patterns will be found. This is further exacerbated by the downward closure property that holds for the participating index measure. That is, if a set of features is prevalent with respect to a threshold of participating index, then all of its subsets will be discovered as prevalent co-location patterns. A huge pattern number will jeopardize the usability of resulted patterns, as it demands great efforts to understand or examine the discovered knowledge. Therefore, summarizing discovered patterns with informative representative patterns can effectively facilitate the understanding and analysis of data.

1.3 Main Contributions and Roadmap

In the remainder of this dissertation, we study summarizing the uncertain databases and spatial databases with representative patterns. For different data mining tasks, we develop effective and efficient algorithms and evaluate the performance theoretically and empirically.

In Chapter 2, we address the problem of summarizing probabilistic frequent patterns from uncertain databases by *Probabilistic Representative Frequent Pattern* (P-RFP) mining, which aims to find the minimal set of patterns that can cover all probabilistic frequent patterns. The cover relationship is determined by the (ε, δ)-*cover* relationship proposed in this dissertation. The approach for P-RFP mining can be divided into two steps: (1) finding the set of patterns that can be (ε, δ)-covered by others; (2) finding minimal P-RFP set by solving a set cover problem. The bottleneck of the approach is checking whether a pattern (ε, δ)-covers another in the first step, which involves the computation of a joint probability of the supports of two patterns. To address the issue, we propose a dynamic programming-based approach, which iteratively updates the (ε, δ)-*cover probability* of two patterns in the first j transactions in an uncertain database. We also develop effective optimization strategies to further improve the computation efficiency.

In Chapter 3, we aim to enhance the efficiency further by addressing the bottleneck in examining whether a pattern (ε, δ)-covers another. We analyze that the joint support probability follows a joint Poisson binomial distribution with both theoretical and empirical proofs. Based on the analysis, we propose an *Approximate P-RFP Mining* (APM) algorithm that performs outstandingly faster than the dynamic programming-based exact approach. To our knowledge, this is the first attempt to analyze the relationship between two probabilistic frequent patterns through an approximate approach.

In Chapter 4, we focus on directly summarizing uncertain transaction databases by formulating the task as *Minimal Probabilistic Tile Cover* (MPTC) Mining, which aims to find a tile set as a high-quality summarization of an uncertain database. Unlike summarizing a deterministic database, the challenge is how to determine whether a tile set covers an uncertain database or not, which is a probabilistic event. Hence, we define the concept of *cover probability*, which is, informally, the probability that a tile set covers the uncertain database. If the cover probability is greater than a threshold, the tile set is called a *probabilistic tile cover* of the database. We define the *cost* and the *density* of a tile set to measure its size and purity degree, respectively. Our goal of MPTC mining is then to find the probabilistic tile cover with the minimal cost, satisfying the condition that the density of the cover is sufficiently high. The bottleneck is to check whether a tile is a better representative than another, which involves the computation of a joint probability. We prove that the joint probability can be decomposed into independent Poisson binomial distributions and design an efficient algorithm to calculate it. Several optimization techniques are also devised to further improve the performance.

In Chapter 5, we propose to summarize co-location patterns using a set of *representative co-location patterns* (RCPs), which strikes a fine balance between improving compression rate and preserving prevalence information.

To formulate the problem of representative co-location pattern mining, we first define a new measure to appropriately quantify the distance between two co-location patterns in terms of their prevalence, based on which the ϵ-cover relationship can be stated on a pair of co-location patterns. To solve the problem of RCP mining, we first propose an algorithm, *RCPFast*, which follows existing distance-based pattern summarization techniques to adopt the *post-mining* framework that finds RCPs from the set of discovered co-location patterns. Observing a peculiar challenge in spatial data mining, we then develop another algorithm, called *RCPMS*, which employs a *mine-and-summarize* framework to discover RCPs directly from the spatial data. To our knowledge, *RCPMS* is the first work among existing distance-based pattern summarization that pushes summarization into the pattern mining process. Optimization strategies are also devised to further improve the efficiency of *RCPMS*.

In Chapter 6, we summarize the main contributions of this dissertation and conclude with remarks.

1.4 Publications

My publications during PhD candidature are listed as follows:

[1] **Chunyang Liu**, Ling Chen, and Chengqi Zhang. Mining Probabilistic Representative Frequent Patterns from Uncertain Data. In SIAM International Conferece on Data Mining (SDM), pages 73–81, Oral Presentation, 2013.

[2] **Chunyang Liu**, Ling Chen, and Chengqi Zhang. Summarizing Probabilistic Frequent Patterns: a Fast Approach. In ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), pages 527–535, Oral Presentation, 2013.

[3] Bozhong Liu, Ling Chen, **Chunyang Liu**, Chengqi Zhang, and Weidong Qiu. RCP Mining: Towards the Summarization of Spatial Co-location Patterns. In International Symposium on Spatial and Temporal Databases (SSTD), pages 451–469, Oral Presentation, 2015.

[4] **Chunyang Liu** and Ling Chen. Summarizing Uncertain Transaction Databases by Probabilistic Tiles. To appear in The annual International Joint

Conference on Neural Networks (IJCNN), 2016.

[5] **Chunyang Liu**, Ling Chen, Ivor W. Tsang, and Chengqi Zhang. Learning Classifiers from High Confidence Rules. Submitted to Pattern Recognition.

Chapter 2

Mining Probabilistic Representative Frequent Patterns From Uncertain Data

Probabilistic frequent pattern mining over uncertain data has received a great deal of attention recently due to the wide applications of uncertain data. Similar to its counterpart in deterministic databases, probabilistic frequent pattern mining suffers from the same problem of generating an exponential number of result patterns. The large number of discovered patterns hinders further evaluation and analysis, and calls for the need to find a small number of representative patterns to approximate all other patterns. This chapter formally defines the problem of *probabilistic representative frequent pattern (P-RFP) mining*, which aims to find the minimal set of patterns with sufficiently high probability to represent all other patterns. The problem's bottleneck turns out to be checking whether a pattern can probabilistically represent another, which involves the computation of a joint probability of the supports of two patterns. To address the problem, we propose a novel and efficient dynamic programming-based approach. Moreover, we have devised a set of effective optimization strategies to further improve the computation efficiency. Our experimental results demonstrate that the proposed P-RFP mining effectively reduces the size of probabilistic frequent patterns. Our proposed approach not only discovers the set of P-RFPs efficiently, but also restores the frequency probability information of patterns with an error guarantee.

2.1 Introduction

Uncertainty is inherent in data from many different domains, including sensor network monitoring, moving object tracking, and protein-protein interaction data [2]. Instead of cleaning uncertain data by considering only the most possible circumstance, it is more reasonable to model the uncertainty of data. Consequently, data mining over uncertain data has become an active area of research in recent years. A survey of state-of-the-art uncertain data mining techniques can be found in [22]. As one of the most fundamental data mining tasks, frequent pattern mining over uncertain data has also received a great deal of research attention, since it was first introduced in [3]. Currently, there exist two different definitions of frequent patterns in the context of uncertain data: *expected support-based frequent patterns* [3, 7], and *probabilistic frequent patterns* [4, 5]. Both definitions consider the *support* of a pattern as a discrete random variable. The former uses the expectation of the support as the measurement, while the latter considers the probability that the support of a pattern is no less than some specified minimum support threshold. Various algorithms have been designed to mine frequent patterns from uncertain data. A summarization and comparison of eight algorithms, proposed to mine respectively the two types of aforementioned frequent patterns from uncertain databases, have been reported in [2].

Note that, the anti-monotonic property holds for both the expected support-based frequent patterns and the probabilistic frequent patterns. That is, if a pattern is frequent in an uncertain database, then all of its sub-patterns are frequent as well. This property leads to the generation of an exponential number of result patterns. The large number of discovered frequent patterns makes the understanding and further analysis of generated patterns troublesome. Therefore, it is important to find a small number of representative patterns to best approximate all other patterns.

Some initial research work has been undertaken to find a small set of representative patterns. For example, motivated by the observation that many frequent patterns have similar items and supporting transactions, mining *probabilistic frequent closed patterns* over uncertain data has been studied in [10, 11, 12]. However, the number of probabilistic frequent closed patterns is still large because of the restrictive condition for a pattern being *closed*. For instance, in [12], the *closed probability* of a pattern is computed as the sum of the probabilities of the possible worlds of an uncertain database where the pattern is closed. In this work, we aim to relax the restrictive condition to further reduce the size of frequent patterns mined over uncertain data.

In the context of deterministic data, a pattern is closed if it is the longest pattern that appears in the same set of transactions supporting its sub-patterns. As a generalization of the concept of frequent closed patterns, Xin et al. [23] proposed the notion of an *ε-cover* relationship between patterns. A pattern X_1 is ε-covered by another pattern X_2 if X_1 is a subset of X_2 and $(\text{Supp}(X_1) - \text{Supp}(X_2))/\text{Supp}(X_1) \leq \varepsilon$. The goal is then to find a minimum set of representative patterns that can ε-cover all frequent patterns. Since the support of a pattern $\text{Supp}(X)$ becomes a discrete random variable in an uncertain database, the ε-cover relationship cannot be applied directly to probabilistic frequent patterns.

In this work, we first extend the concept of ε-cover by defining a new *(ε, δ)-cover* relationship between probabilistic frequent patterns. Informally, a pattern X_1 is (ε, δ)-covered by another pattern X_2 in an uncertain database if X_1 is a subset of X_2, and the probability that the support distance between X_1 and X_2 is no greater than ε is no less than δ. The objective of probabilistic representative frequent pattern (P-RFP) mining is then to find the minimal set of patterns that can (ε, δ)-cover all probabilistic frequent patterns.

The approach for P-RFP mining can be divided into two steps: (1) finding the set of patterns that can be (ε, δ)-covered by others; (2) finding minimal P-RFPs by solving a set cover problem. The bottleneck of approach is checking whether a pattern (ε, δ)-covers another in the first step, which involves the computation of a joint probability of the supports of two patterns. To address the problem, we propose a dynamic programming-based approach, which iteratively updates the *(ε, δ)-cover probability* of two patterns in the first j transactions in an uncertain database. We also develop a set of effective optimization strategies to further improve the computation efficiency of the proposed approach.

To our knowledge, this is the first work that summarizes frequent patterns mined over uncertain databases by probabilistic representative pattern mining. Our experimental results show that our approach summarizes frequent patterns effectively, and restores the patterns and their original frequency probability information with a guaranteed error bound.

The remainder of the chapter is structured as follows. The next section introduces related works to this chapter. We define important concepts and provide a problem statement in Section 2.3. Section 2.4 describes the proposed data mining approach. Experimental results are presented in Section 2.5. Section 2.6 closes this chapter with some conclusive remarks.

2.2 Related Work

In this section, we review related research from two sub-areas: frequent pattern mining over uncertain data and frequent pattern summarization.

2.2.1 Frequent pattern mining over uncertain data

Frequent pattern mining is first introduced in [24, 25] and then studied in [26]. Many approaches have been proposed to mine frequent patterns from uncertain databases in past years. Based on the definition of a frequent pattern, existing work on mining frequent patterns over uncertain data falls into two categories: *expected support-based frequent pattern mining* [3, 27, 6, 7] and *probabilistic frequent pattern mining* [4, 5]. The former employs the *expectation of support* as the measurement. That is, a pattern is frequent only if its expected support is no less than a specified minimum expected support. The latter considers the *frequency probability* as the measurement, which refers to the probability that a pattern appears no less than a specified minimum number of support times. A pattern is therefore frequent only if its frequency probability is no less than a specified minimum probability (i.e. $\Pr(\text{Supp}(X) \geq minsup) \geq minprob$).

For mining expected support-based frequent patterns, there are three representative algorithms: UApriori [3], UFP-growth [6], UH-Mine [7]. UApriori is the uncertain version of the well-known Apriori algorithm. Both UFP-growth and UH-Mine are based on the divide-and-conquer framework that uses the depth-first strategy to search frequent patterns. For mining probabilistic frequent patterns, two representative algorithms are DP − dynamic programming-based Apriori algorithm [4], and DC − divide-and-conquer-based Apriori algorithm [5]. Observing that the support of a pattern in an uncertain database can be represented by Poisson binomial distribution, some approximate probabilistic frequent pattern mining algorithms have also been proposed. [8, 28] respectively use the normal distribution and the Poisson method to approximate the frequency probability of patterns. Tong et al. [2] verified that the two types of frequent patterns mined from uncertain data have a tight connection and can be unified when the size of data is large enough. They also empirically compared the performance of eight existing representative algorithms with uniform measures.

2.2.2 Frequent pattern summarization

Motivated by the fact that frequent pattern mining may generate an exponential number of patterns due to the downward closure property, a lot of research work has been dedicated to summarizing the complete set of patterns with a small set of representative ones. Various concepts have been proposed, such as maximal patterns [29], frequent closed patterns [30], and non-derivable patterns [31], top-K frequent closed patterns [32, 33], and redundancy-aware top-K patterns [34]. While all frequent patterns can be recovered from maximal patterns, the support information is lost. Although the set of frequent closed patterns preserve the exact support of all frequent patterns, the number of frequent closed patterns can still be tens of thousands or even more. There are several generalizations of closed patterns, such as the pattern profiling based approaches [35, 36, 37] and the support distance based approaches [23, 38]. It was observed in [38] that the profile-based approaches [35, 36] have some drawbacks, such as no error guarantee on restored support. This work borrows the framework of the support distance based approaches to find probabilistic representative frequent patterns.

Recently, some research work has been undertaken to summarize frequent patterns mined over uncertain data. Tang and Peterson [11] proposed mining probabilistic frequent closed patterns, based on the concept called *probabilistic support*. Tong et al. [12] pointed out that frequent closed patterns defined on probabilistic support cannot guarantee the patterns are closed in possible worlds which contribute to their probabilistic supports. Instead, they defined the threshold-based frequent closed patterns over probabilistic data, which considers the probabilities of possible worlds where a pattern is closed. Our research relaxes the condition to further reduce the size of patterns by considering the probabilities of possible worlds where a pattern can ε-cover another one.

2.3 Problem Definition

This section first introduces preliminary definitions and then formulates the problem of probabilistic representative frequent pattern (P-RFP) mining.

Xin et al. [23] defined a robust distance measure between patterns in deterministic data.

Definition 2.1. (*distance measure*) *Given two patterns X_1 and X_2, the distance between them, denoted as $d(X_1, X_2)$, is defined as $1 - |T(X_1) \cap T(X_2)|/|T(X_1) \cup T(X_2)|$, where $T(X_i)$ is the set of transactions supporting pattern X_i.*

Then, an ε-cover relationship is defined on two patterns where one subsumes another.

Definition 2.2. (ε*-cover*) *Given a real number $\varepsilon \in [0, 1]$ and two patterns X_1 and X_2, we say X_1 is ε-covered by X_2 if $X_1 \subseteq X_2$ and $dist(X_1, X_2) \leq \varepsilon$.*

As commonly used in frequent pattern mining, $X_1 \subseteq X_2$ denotes X_1 is a subset of X_2 (e.g. $\{a\} \subseteq \{a, b\}$). It can be proved easily that, if X_2 ε-covers X_1, $(\text{Supp}(X_1) - \text{Supp}(X_2))/\text{Supp}(X_1) \leq \varepsilon$. The goal of representative frequent pattern mining then becomes finding the minimal set of patterns that ε-cover all frequent patterns.

In the context of uncertain data, the support of a pattern, $\text{Supp}(X_i)$, becomes a discrete random variable. Therefore, we cannot directly apply the

ID	Transactions
T_1	a:0.7 b:0.2
T_2	a:1.0 c:0.5

Table 2.1: An uncertain database with attribute uncertainty

ID	Possible World	Prob.
w_1	$\{T_1 : \phi, T_2 : \{a\}\}$	0.12
w_2	$\{T_1 : \{a\}, T_2 : \{a\}\}$	0.28
w_3	$\{T_1 : \{b\}, T_2 : \{a\}\}$	0.03
w_4	$\{T_1 : \{a, b\}, T_2 : \{a\}\}$	0.07
w_5	$\{T_1 : \phi, T_2 : \{a, c\}\}$	0.12
w_6	$\{T_1 : \{a\}, T_2 : \{a, c\}\}$	0.28
w_7	$\{T_1 : \{b\}, T_2 : \{a, c\}\}$	0.03
w_8	$\{T_1 : \{a, b\}, T_2 : \{a, c\}\}$	0.07

Table 2.2: An example of possible worlds

ε-cover relationship to probabilistic frequent patterns. Before explaining how to extend the concept of ε-covered in the context of uncertain data, we examine an uncertain database where attributes are associated with existential probabilities. Table 2.1 shows an uncertain transaction database where each transaction consists of a set of probabilistic items. For example, the probability that item a appears in the first transaction T_1 is 0.7. *Possible world semantics* are commonly used to explain the existence of data in an uncertain database. For example, the database in Table 2.1 has eight possible worlds, which are listed in Table 2.2. Each possible world is associated with an existential probability. For instance, the probability that the first possible world w_1 exists is $(1 - 0.7) \times (1 - 0.2) \times 1 \times (1 - 0.5) = 0.12$.

Considering that the occurrences of items in every possible world are deterministic, we can define the probabilistic distance between two probabilistic frequent patterns based on their distance in possible worlds.

Definition 2.3. (*probabilistic distance measure*) *Given an uncertain database D, and two patterns X_1 and X_2, let $\mathcal{PW} = \{w_1, \ldots, w_m\}$ be the set of possible worlds derived from D, the distance between X_1 and X_2 in a possible world $w_j \in$*

$\mathcal{PW}$ is

$$dist(X_1, X_2; w_j) = 1 - \frac{|T(X_1; w_j) \cap T(X_2; w_j)|}{|T(X_1; w_j) \cup T(X_2; w_j)|} \tag{2.1}$$

where $T(X_i; w_j)$ is the set of transactions containing pattern X_i in the possible world w_j. Then, the probabilistic distance between X_1 and X_2, denoted by $dist(X_1, X_2)$, is a random variable. The probability mass function of $dist(X_1, X_2)$ is:

$$\Pr(dist(X_1, X_2) = d) = \sum_{w_j \in \mathcal{PW}, dist(X_1, X_2; w_j) = d} \Pr(w_j) \tag{2.2}$$

That is, the probability that the distance between two probabilistic frequent patterns is d equals to the sum of the probabilities of possible worlds where the distance between the two patterns is d.

For example, consider the uncertain database in Table 2.1. Let $X_1 = \{a\}$ and $X_2 = \{a, b\}$. The probability that the distance between X_1 and X_2 is equal to 0.5, $\Pr(dist(X_1, X_2) = 0.5)$, can be computed by adding the probabilities of the possible worlds w_4 and w_8. This is because only in the two possible worlds, the distance between the two patterns is 0.5. Therefore, $\Pr(dist(X_1, X_2) = 0.5) = 0.14$.

Based on the probabilistic distance measure, we define the ε-cover probability as follows.

Definition 2.4. (ε*-cover probability*) *Given an uncertain database D, two patterns X_1 and X_2, and a distance threshold ε, the ε-cover probability of X_1 and X_2 is $\Pr_{cover}(X_1, X_2; \varepsilon) = \Pr(dist(X_1, X_2) \leq \varepsilon)$.*

Definition 2.5. ((ε, δ)*-cover*) *Given an uncertain database D, two patterns X_1 and X_2, a distance threshold ε and a cover probability threshold δ, we say X_2 (ε, δ)-covers X_1 if $X_1 \subseteq X_2$ and $\Pr_{cover}(X_1, X_2; \varepsilon) \geq \delta$.*

Our goal is then to obtain the minimal set of patterns that will (ε, δ)-cover all the probabilistic frequent patterns. The formal statement of the probabilistic representative frequent pattern (P-RFP) mining is as follows.

Definition 2.6. (*Problem Statement*) *Given an uncertain database D, a set of probabilistic frequent patterns $\mathcal{F}$, a probabilistic distance threshold ε and a cover* *probability threshold δ, the problem of probabilistic representative frequent pattern (P-RFP) mining is to find the minimal set of patterns $\mathcal{R}$ so that, for any frequent pattern $X \in \mathcal{F}$, there exists a representative pattern $X' \in \mathcal{R}$ where X' (ε, δ)-covers X.*

It is obvious that when $\varepsilon = 0$, the probabilistic representative pattern set is probabilistic closed patterns, and when $\varepsilon = 1$, it is probabilistic maximal patterns.

2.4 P-RFP Mining

This section first describes the framework of our proposed approach. Then, we explain the details of the main steps for P-RFP mining.

2.4.1 Framework of P-RFP Mining

Before presenting the framework of our approach for P-RFP mining, we develop some important lemmas between two patterns where one (ε, δ)-covers another.

Lemma 2.1. *Given an uncertain database D and two patterns X_1 and X_2 s.t. X_2 (ε, δ)-covers X_1, the distance between X_1 and X_2 in the possible world w_j can be represented by the support of the patterns in w_j:*

$$dist(X_1, X_2; w_j) = 1 - \frac{\text{Supp}(X_2; w_j)}{\text{Supp}(X_1; w_j)} \tag{2.3}$$

Proof. Since X_2 (ε, δ)-covers X_1, then $X_1 \subseteq X_2$,

$$\begin{aligned} dist(X_1, X_2; w_j) =& 1 - \frac{|T(X_1; w_j) \cap T(X_2; w_j)|}{|T(X_1; w_j) \cup T(X_2; w_j)|} \\ =& 1 - \frac{|T(X_2; w_j)|}{|T(X_1; w_j)|} \\ =& 1 - \frac{\text{Supp}(X_2; w_j)}{\text{Supp}(X_1; w_j)}, \end{aligned}$$

which proves the lemma. ☐

Lemma 2.2. *Given an uncertain database D and two patterns X_1 and X_2 s.t. X_2 (ε, δ)-covers X_1, the probabilistic distance $dist(X_1, X_2)$ can be represented by the support distribution of X_1 and X_2:*

$$dist(X_1, X_2) = 1 - \frac{\mathrm{Supp}(X_2)}{\mathrm{Supp}(X_1)} \tag{2.4}$$

Proof. $\mathrm{Supp}(X_i)$ is a discrete random variable with the following probability density function.

$$\Pr(\mathrm{Supp}(X_i) = k) = \sum_{w_j \in \mathcal{PW}, \mathrm{Supp}(X; w_j) = k} \Pr(w_j) \tag{2.5}$$

According to the definition of probabilistic distance and Lemma 2.1, we have

$$\begin{aligned}\Pr(dist(X_1, X_2) = d) &= \sum_{w_j \in \mathcal{PW}, dist(X_1, X_2; w_j) = d} \Pr(w_j) \\ &= \sum_{w_j \in \mathcal{PW}, 1 - \frac{\mathrm{Supp}(X_2; w_j)}{\mathrm{Supp}(X_1; w_j)} = d} \Pr(w_j)\end{aligned}$$

For brevity, let $W' = \{w_j \mid w_j \in \mathcal{PW}, \mathrm{Supp}(X_1; w_j) = k, \mathrm{Supp}(X_2; w_j) = (1-d)k\}$, where $k \in [0, |D|]$, then

$$\begin{aligned}\Pr(dist(X_1, X_2) = d) &= \sum_{k=1}^{|D|} \sum_{w_j \in W'} \Pr(w_j) \\ &= \sum_{k=1}^{|D|} \Pr(\mathrm{Supp}(X_2) = k, \mathrm{Supp}(X_1) = (1-d)k) \\ &= \Pr\left(\left(1 - \frac{\mathrm{Supp}(X_2)}{\mathrm{Supp}(X_1)}\right) = d\right),\end{aligned}$$

which proves the lemma. ☐

Lemma 2.3. *Given an uncertain database D and two patterns X_1 and X_2 s.t. X_2 (ε, δ)-covers X_1, we have*

$$\Pr\left(\text{Supp}(X_2) \geq (1-\varepsilon)\text{Supp}(X_1)\right) \geq \delta \tag{2.6}$$

Proof. Since X_2 (ε, δ)-covers X_1, according to Lemma 2.2, we have

$$\begin{aligned} & \Pr(dist(X_1, X_2) \leq \varepsilon) \\ = & \Pr\left(1 - \frac{\text{Supp}(X_2)}{\text{Supp}(X_1)} \leq \varepsilon\right) \\ = & \Pr(\text{Supp}(X_2) \geq (1-\varepsilon)\text{Supp}(X_1)) \geq \delta, \end{aligned}$$

which proves the lemma. □

Lemma 2.4. *Given an uncertain database D, two patterns X_1 and X_2, a support threshold minsup and a frequency probability threshold minprob, if X_2 (ε, δ)-covers X_1, and X_1 is a probabilistic frequent pattern w.r.t. minsup and minprob, then X_2 is a probabilistic frequent pattern w.r.t. $(1-\varepsilon)minsup$ and $(\delta \cdot minprob)$.*

Proof. Since X_1 is a probabilistic frequent pattern w.r.t. *minsup* and *minprob*, we have $\Pr\left(\text{Supp}(X_1) \geq minsup\right) \geq minprob$, which infers,

$$\Pr((1-\varepsilon)\text{Supp}(X_1) \geq (1-\varepsilon)minsup) \geq minprob$$

From Lemma 2.3, we have,

$$\Pr(\text{Supp}(X_2) \geq (1-\varepsilon)\text{Supp}(X_1)) \geq \delta$$

Hence, $\Pr\left(\text{Supp}(X_2) \geq (1-\varepsilon)minsup\right) \geq \delta \cdot minprob$. That is, X_2 is a probabilistic frequent pattern w.r.t. $(1-\varepsilon)minsup$ and $(\delta \cdot minprob)$. □

According to Lemma 2.4, to find the representative patterns to (ε, δ)-cover the complete set of probabilistic frequent patterns w.r.t. *minsup* and *minprob*, denoted as F, we need to consider the set of pseudo probabilistic frequent patterns w.r.t $(1-\varepsilon)minsup$ and $(\delta \cdot minprob)$, denoted as $\hat{F}$.

Given the two sets F and $\hat{F}$, our approach for P-RFP mining consists of the following two steps.

1. Generate the cover set for every pattern in $\hat{F}$. For each pattern X in $\hat{F}$, the cover set of X, denoted as $C(X)$, is a set of probabilistic frequent patterns in F that can be (ε, δ)-covered by X. That is, $C(X) \subseteq F$.

2. Find the minimal pattern set $R \subseteq \hat{F}$ to (ε, δ)-cover all probabilistic frequent patterns in F.

After finding the cover sets for patterns in $\hat{F}$ in the first step, the second step is equivalent to finding a minimal number of cover sets that cover all patterns in F. This is known as a set cover problem, which is NP-hard. Similar to [38], we adopt a well-known greedy set cover algorithm [39], which achieves polynomial complexity. Therefore, in the following, we focus on describing the first step, which generates the cover set for each pseudo probabilistic frequent pattern in $\hat{F}$. The framework of our approach is illustrated in Algorithm 2.1.

2.4.2 Cover Set Generation

To generate the cover set for a pattern X_2 in $\hat{F}$, for each pattern X_1 in F so that $X_1 \subseteq X_2$, we need to check if X_2 (ε, δ)-covers X_1. That is, we need to examine whether the ε-cover probability between X_1 and X_2 is no less than δ (i.e., $\Pr(dist(X_1, X_2) \leq \varepsilon) \geq \delta$). According to Lemma 2.3, the ε-cover probability $\Pr_{cover}(X_1, X_2; \varepsilon) = \Pr(dist(X_1, X_2) \leq \varepsilon)$ is equivalent to $\Pr(\mathrm{Supp}(X_2) \geq (1-\varepsilon)\mathrm{Supp}(X_1))$. Let $\mathrm{Supp}(X_1) = l$, and $\mathrm{Supp}(X_2) = k$. Then, the ε-cover probability between X_1 and X_2 is equal to,

$$\sum_{l=0}^{|D|} \sum_{k=\lceil(1-\varepsilon)l\rceil}^{l} \Pr(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k) \tag{2.7}$$

To compute the value of Equation (2.7) to find out whether it is no less than δ, we introduce the joint support probability distribution as follows.

Definition 2.7 (joint support probability)**.** *Given an uncertain database D and* *patterns X_1 and X_2, the joint support probability mass function is*

$$\begin{aligned} & \Pr(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k) \\ = & \sum_{w_i \in \mathcal{PW}, \mathrm{Supp}(X_1; w_i) = l, \mathrm{Supp}(X_2; w_i) = k} \Pr(w_i) \end{aligned} \tag{2.8}$$

To split the computation of the joint support probability of X_1 and X_2 into smaller sub-problems, we define the partial joint support probability distribution as follows.

Definition 2.8 (partial joint support probability)**.** *Given an uncertain database D and patterns X_1 and X_2, the j-partial joint support probability is the joint support probability of X_1 and X_2 in the first j transactions of D. The j-partial joint support mass function is*

$$\Pr_j(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k) = \sum_{w_i \in \mathcal{PW}, \mathrm{Supp}_j(X_1; w_i) = l, \mathrm{Supp}_j(X_2; w_i) = k} \Pr(w_i)$$

It can be proved that the partial joint support probability can be computed in a recursive strategy.

Lemma 2.5. *Given an uncertain database D and patterns X_1 and X_2, $X_1 \subseteq X_2$, $j, k, l \in \mathbb{Z}$ and $0 \leq k \leq l \leq j \leq |D|$, then:*

$$\begin{aligned} & \Pr_j(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k) \\ = & \Pr_{j-1}(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k)(1 - p_j^{X_1}) \\ + & \Pr_{j-1}(\mathrm{Supp}(X_1) = l - 1, \mathrm{Supp}(X_2) = k)(p_j^{X_1} - p_j^{X_2}) \\ + & \Pr_{j-1}(\mathrm{Supp}(X_1) = l - 1, \mathrm{Supp}(X_2) = k - 1) p_j^{X_2} \end{aligned} \tag{2.9}$$

where $p_j^{X_i}$ is the probability that X_i occurs in the jth transaction. The boundary case is: $\Pr_j(\mathrm{Supp}(X_1) = 0, \mathrm{Supp}(X_2) = 0) = \prod_{m=1}^{j}(1 - p_m^{X_1})$.

Proof. In jth-transaction $t_j \in D$, there are only three existence possibilities of X_1 and X_2, since $X_1 \subseteq X_2$. Table 2.3 lists the three situations and their respective existential probabilities. Therefore, we can split the computation of $\Pr_j(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k)$ into the three situations with corresponding probability. The equation of the boundary case is intuitive. □

Situation	Probability
$X_1 \subseteq t_j, X_2 \subseteq t_j$	$p_j^{X_2}$
$X_1 \subseteq t_j, X_2 \nsubseteq t_j$	$p_j^{X_1} - p_j^{X_2}$
$X_1 \nsubseteq t_j, X_2 \nsubseteq t_j$	$1 - p_j^{X_1}$

Table 2.3: Probability of different situations in the jth transaction

Lemma 2.5 enables us to compute the cover probability iteratively using a dynamic programming scheme. This equation is the foundation of our approach. Although it is feasible and effective, we can accelerate it through certain optimization techniques, which are stated in the next section.

2.4.3 Optimization Strategies

While Lemma 2.5 approves the cover probability between two patterns can be updated transaction by transaction, the transactions which support neither of the two patterns can actually be skipped.

Lemma 2.6. *Given an uncertain database D, two patterns X_1 and X_2 s.t. $X_1 \subseteq X_2$, and a probabilistic distance threshold ε, $\Pr_{cover}(X_1, X_2; \varepsilon)$ computed on D is equal to that computed on $D(X_1)$, where $D(X_1)$ is $\{t | P(X_1 \subseteq t) > 0, t \in D\} \subseteq D$.*

Lemma 2.6 is intuitive. According to Definition 2.3 and Definition 2.4, only the transactions supporting at least the sub-pattern X_1 will contribute to the value of probabilistic distance, which in turn affects the ε-cover probability. This lemma allows us to compute the ε-cover probability on a projected sub-database, which significantly reduces the runtime of computation.

Lemma 2.7. *Given an uncertain database D and two patterns X_1 and X_2 s.t. $X_1 \subseteq X_2$, if X_2 (ε, δ)-covers X_1, then $\forall X$ s.t. $X_1 \subseteq X \subseteq X_2$, we have X_2 (ε, δ)-covers X.*

Proof. Since X_2 (ε, δ)-covers X_1, according to Lemma 2.3 we have

$$\Pr(\mathrm{Supp}(X_2) \geq (1 - \varepsilon)\mathrm{Supp}(X_1)) \geq \delta$$

Algorithm 2.1 P-RFP Mining Framework

```
Input: $D$, $F$, $\hat{F}$, $\varepsilon$ and $\delta$
Output: Minimal P-RFP Set $R$
 1: $R \leftarrow \emptyset$
 2: $CoverSets \leftarrow \emptyset$
 3: for all $X_2 \in \hat{F}$ do
 4:   $NoCoverSet \leftarrow \emptyset$
 5:   for all $X_1 \in F$ such that $X_1 \subseteq X_2$ do
 6:     if $isCover(X_1, X_2) = True$ then
 7:       $CoverSets[X_2].add(X_1)$
 8:     else
 9:       $NoCoverSet.add(X_1)$
10: $R = setCover(CoverSets, F)$
11: return $R$
```

$\forall X$ that $X_1 \subseteq X \subseteq X_2$, we have $\text{Supp}(X_1; w_j) \geq \text{Supp}(X; w_j) \geq \text{Supp}(X_2; w_j)$ in every possible world w_j. Therefore, $\Pr(\text{Supp}(X_2) \geq (1-\varepsilon)\text{Supp}(X_1)) \geq \delta \Rightarrow$

$$\Pr(\text{Supp}(X_2) \geq (1-\varepsilon)\text{Supp}(X_1) \geq (1-\varepsilon)\text{Supp}(X)) \geq \delta$$

Hence, X_2 (ε, δ)-covers X. □

According to Lemma 2.7, we have the following corollary.

Corollary 2.1. *Given an uncertain database D and two patterns X_1 and X_2, $X_1 \subseteq X_2$, if X_2 cannot (ε, δ)-cover X_1, then $\forall X \subseteq X_1$, X_2 cannot (ε, δ)-cover X.*

Lemma 2.7 and Corollary 2.1 reduce the number of pattern pairs, for which the cover probability needs to be computed.

Lemma 2.8. *Given an uncertain database D, two patterns X_1 and X_2 s.t. $X_1 \subseteq X_2$, a distance threshold ε, and a cover probability threshold δ, if $\exists j, 1 \leq j \leq |D|$ such that $\prod_{m=1}^{j}(1 - p_m^{X_1} + p_m^{X_2}) \geq \delta$, then X_2 (ε, δ)-covers X_1, where $p_m^{X_i}$ is the probability that X_i occurs in the m-th transaction.*

Algorithm 2.2 Function *isCover*

```
Input: $X_1, X_2$
Output: If $X_2$ $(\varepsilon, \delta)$-covers $X_1$, then return True, else False
 1: for all $X \in CoverSets[X_2]$ do
 2:   if $X \subseteq X_1$ then
 3:     return True
 4: for all $X \in NoCoverSet[X_2]$ do
 5:   if $X \supseteq X_1$ then
 6:     return False
 7: $Q(|D(X_1)|) \leftarrow \prod_{m=1}^{|D(X_1)|}(1 - p_m^{X_1} + p_m^{X_2})$
 8: if $Q(|D(X_1)|) \geq \delta$ then
 9:   return True
10: for $l = 0$ to $|D(X_1)|$ do
11:   for $k = \lceil(1-\varepsilon)l\rceil$ to $l - 1$ do
12:     $P_{cover} += \Pr(\text{Supp}(X_1) = l, \text{Supp}(X_2) = k)$
13:     if $P_{cover} \geq \delta$ then
14:       return True
15: return False
```

Proof. According to the definition of cover probability, we have

$$\begin{aligned}\Pr_{cover}(X_1, X_2; \varepsilon) &= \Pr(dist(X_1, X_2) \leq \varepsilon)\\ &= \sum_{l=0}^{|D|} \sum_{k=\lceil(1-\varepsilon)l\rceil}^{l} \Pr(\text{Supp}(X_1) = l, \text{Supp}(X_2) = k)\\ &\geq \sum_{l=0}^{|D|} \Pr(\text{Supp}(X_1) = l, \text{Supp}(X_2) = l)\end{aligned}$$

Let $Q(j) = \sum_{l=0}^{j} \Pr(\text{Supp}(X_1) = l, \text{Supp}(X_2) = l)$. If $Q(|D|) \geq \delta$, then $\Pr_{cover}(X_1, X_2; \varepsilon) \geq \delta$ must be valid due to $Q(|D|) \leq \Pr(dist(X_1, X_2) \leq \varepsilon)$. Now we prove that $Q(j)$ can be computed using continued multiplications.

$$\begin{aligned}Q(j) =& \sum_{l=0}^{j} \Pr_j(\text{Supp}(X_1) = l, \text{Supp}(X_2) = l)\\ =& \Pr_j(\text{Supp}(X_1) = 0, \text{Supp}(X_2) = 0)\\ &+ \sum_{l=1}^{j} \Pr_j(\text{Supp}(X_1) = l, \text{Supp}(X_2) = l)\end{aligned}$$

Consider that there are only two situations in the jth transaction: both X_1 and X_2 occur or neither of them occur, we have

$$\begin{aligned}
Q(j) =& \Pr_j(\text{Supp}(X_1) = 0, \text{Supp}(X_2) = 0) \\
&+ \sum_{l=1}^{j} \left\{ \Pr_{j-1}(\text{Supp}(X_1) = l, \text{Supp}(X_2) = l)(1 - p_j^{X_1}) \right\} \\
&+ \sum_{l=1}^{j} \left\{ \Pr_{j-1}(\text{Supp}(X_1) = l-1, \text{Supp}(X_2) = l-1) p_j^{X_2} \right\} \\
=& \Pr_j(\text{Supp}(X_1) = 0, \text{Supp}(X_2) = 0) \\
&+ \sum_{l=0}^{j} \left\{ \Pr_{j-1}(\text{Supp}(X_1) = l, \text{Supp}(X_2) = l)(1 - p_j^{X_1}) \right\} \\
&- \Pr_{j-1}(\text{Supp}(X_1) = 0, \text{Supp}(X_2) = 0)(1 - p_j^{X_1}) \\
&+ \sum_{l=0}^{j} \left\{ \Pr_{j-1}(\text{Supp}(X_1) = l, \text{Supp}(X_2) = l) p_j^{X_2} \right\} \\
=& Q(j-1)(1 - p_j^{X_1} + p_j^{X_2})
\end{aligned}$$

Therefore,

$$\begin{aligned}
Q(j) &= Q(j-1)(1 - p_j^{X_1} + p_j^{X_2}) \\
&= Q(j-2)(1 - p_{j-1}^{X_1} + p_{j-1}^{X_2})(1 - p_j^{X_1} + p_j^{X_2}) \\
&\vdots \\
&= \prod_{m=1}^{j} (1 - p_m^{X_1} + p_m^{X_2})
\end{aligned}$$

Since $Q(j) \leq \Pr_{cover}(X_1, X_2; \varepsilon)$, if $\prod_{m=1}^{j}(1 - p_m^{X_1} + p_m^{X_2}) \geq \delta$, then X_2 (ε, δ)-cover X_1. $\square$

Lemma 2.8 defines a lower bound of ε-cover probability, which can be computed efficiently using continued multiplication. If the lower bound is less than the cover probability threshold δ, we can immediately decide X_2 cannot (ε, δ)-cover X_1 without computing their real ε-cover probability.

2.4.4 P-RFP Mining Algorithm

The overall framework of our P-RFP mining algorithm is shown in Algorithm 2.1. From lines $3 - 9$, we find the cover set for each pattern X_2 in the pseudo probabilistic frequent patterns $\hat{F}$. The most important step is to check whether X_2 covers $X_1 \in F$ (line 6). The details of the function *isCover* is illustrated in Algorithm 2.2, where lines $1 - 3$ implement the optimization stated by Lemma 2.7, and lines $4 - 6$ apply the Corollary 2.1. Lines $7 - 9$ in the function *isCover* use Lemma 2.8 to efficiently discover the lower bound of ε-cover probability. Note that, according to Lemma 2.6, the lower bound, as well as the real ε-cover probability, can be computed on a sub-database $D(X_1)$. Finally, from lines $10 - 14$, we use the dynamic programming based scheme to compute the ε-cover probability. As mentioned before, the function *setCover* in Algorithm 2.1 is solved using the greedy algorithm in [39].

2.5 Performance Study

This section evaluates the effectiveness of P-RFPs, the performance of our approach for P-RFP mining, and the optimization strategies.

2.5.1 Data sets

Two datasets are used in our experiments. The first is the Retail dataset from the Frequent Itemset Mining (FIMI) Dataset Repository.[1] This is one of the standard datasets used in frequent pattern mining in deterministic databases. In order to bring uncertainty into the dataset, following previous works [7], we synthesize an existential probability for each item based on a Gaussian distribution with the mean of 0.9 and the variance of 0.125. This dataset is an uncertain database that associates uncertainty to attributes.

The second one is the iceberg sighting record from 1993 to 1997 on the North Atlantic from the International Ice Patrol (IIP) Iceberg Sightings Database.[2] The IIP Iceberg Sighting Database collects information of iceberg activities

[1]http://fimi.cs.helsinki.fi/data/

[2]http://nsidc.org/data/g00807.html

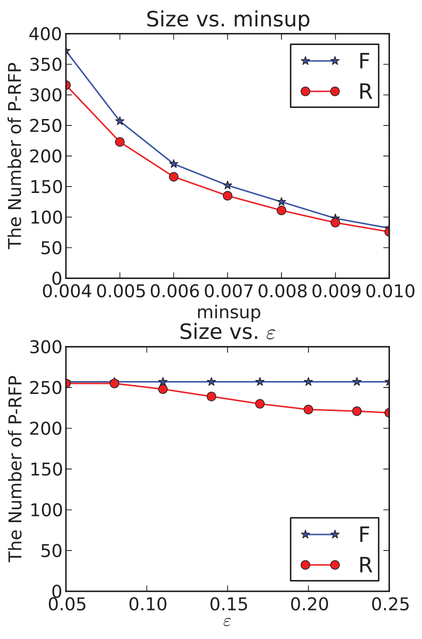


Figure 2.1: The Number of P-RFP on Retail

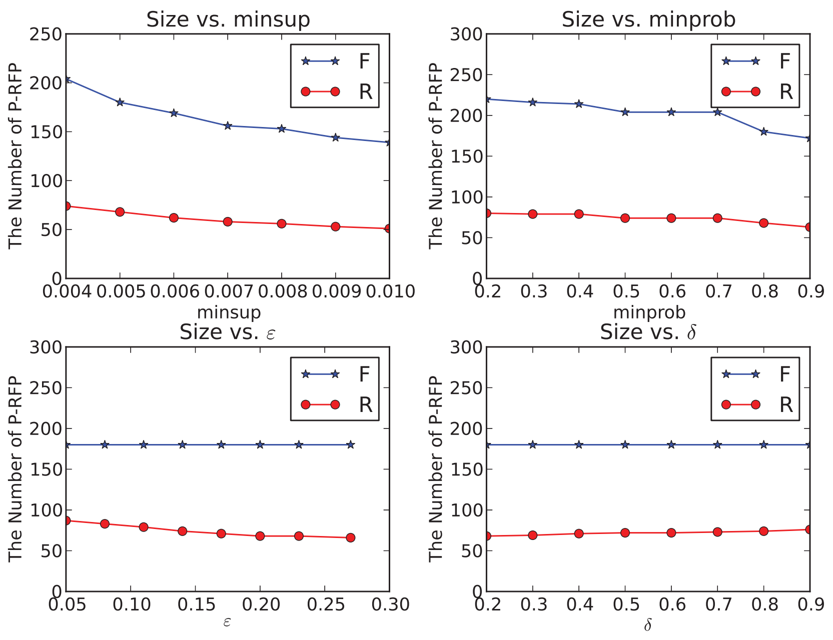


Figure 2.2: The Number of P-RFP on IIP

in the North Atlantic. Each transaction in the database contains the information of date, location, size, shape, reporting source and a confidence level.

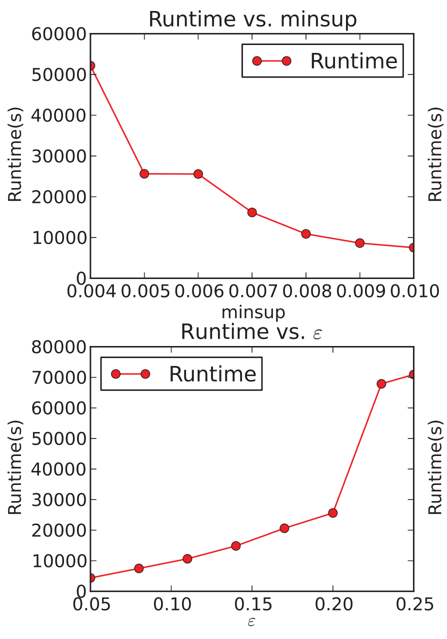


Figure 2.3: Runtime on Retail

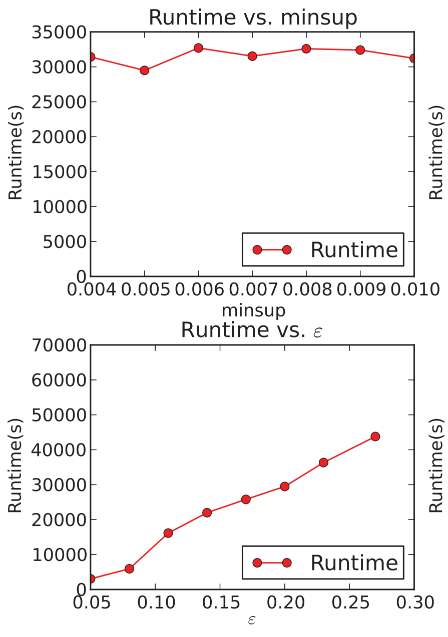


Figure 2.4: Runtime on IIP

The confidence level has six possible attributes, R/V (Radar and visual), R (Radar only), V (Visual), MEA (Measured), EST (Estimated) and GBL (Gar-

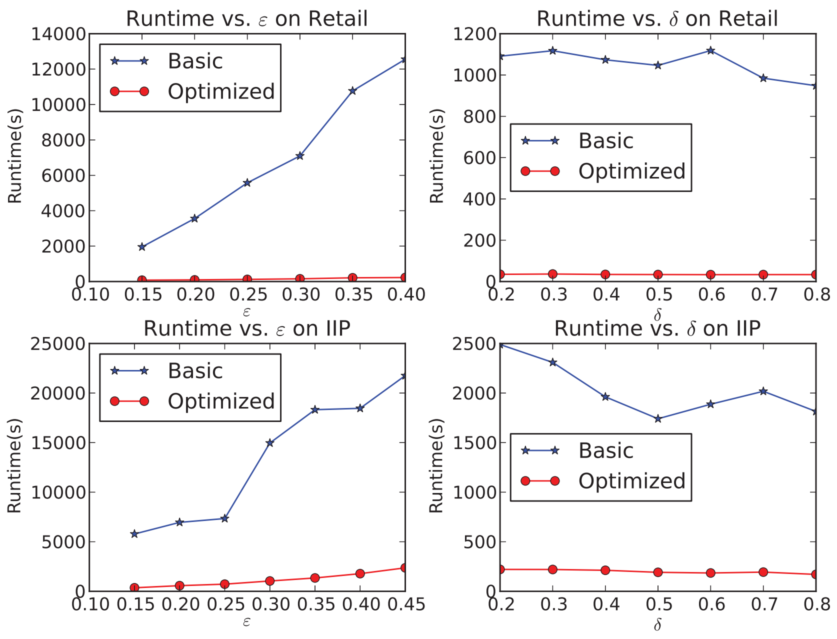


Figure 2.5: Effect of Optimization

bled), which indicate different reliabilities of that tuple. We translate confidence levels to probabilities 0.8, 0.7, 0.6, 0.5, 0.4 and 0.3, respectively. This dataset is an uncertain database that associates uncertainty to tuples.

2.5.2 Result analysis

We first evaluate the compression rate of the P-RFPs, with respect to the variation of parameters. We randomly select 1000 transactions from the two datasets respectively to conduct the experiment. The sizes of R (the set of P-RFPs), and F (the set of probabilistic frequent patterns), with respect to the variations of *minsup*, *minprob*, ε, and δ, on the two datasets are shown in Figures 2.1 and 2.2 respectively. The default values of the four parameters are set to 0.5%, 0.8, 0.2 and 0.2 respectively. It can be observed from the results on both datasets, when *minsup* and *minprob* are low, the compression rate of P-RFPs is high because there are more probabilistic frequent patterns. For the variations of ε and δ, obviously, the high compression rate can be achieved if the probabilistic distance threshold ε is high and/or the cover probability threshold δ is low.

We then examine the runtime of the proposed algorithm for P-RFP mining. Figures 2.3 and 2.4 show the runtime vs. $minsup$, $minprob$, ε, and δ curves on 1000 transactions randomly selected from the two datasets, respectively. The default values of the four parameters are same as in the first experiment. It is intuitive that, when ε is increasing or $minsup$, $minprob$ and δ are decreasing, the runtime will increase because more pattern pairs are engaged in cover probability checking. We find that the growth of both ε and δ lead to a trade-off between the number of P-RFPs and runtime.

We also evaluate the effectiveness of the optimization strategies proposed in Section 2.4.3. We randomly select 500 transactions from the two datasets, respectively, to carry out this experiment. The default values for the experiments on the Retail dataset are: $minsup = 4\%$, $minprob = 0.8$, $\varepsilon = 0.1$ and $\delta = 0.2$. On the IIP dataset, the four parameters are set to 10%, 0.8, 0.1 and 0.2 by default, respectively. Figure 2.5 shows the runtime of the basic version of our algorithm, and the runtime of the algorithm integrated with optimization strategies, with respect to the variation of ε and δ on the two datasets, respectively. The results clearly reveal the effectiveness of the optimization strategies by demonstrating that the optimized algorithm significantly reduces the runtime.

2.6 Conclusions

Due to the downward closure property, the number of probabilistic frequent patterns mined over uncertain data can be so large that they hinder further analysis and exploitation. This chapter proposes the P-RFP mining, which aims to find a small set of patterns to represent the complete set of probabilistic frequent patterns. To address the data uncertainty issue, we define the concept of probabilistic distance and an (ε, δ)-cover relationship between two patterns. The P-RFP set is the minimal set of patterns that (ε, δ)-cover the complete set of probabilistic frequent patterns. We develop a P-RFP mining algorithm that uses a dynamic programming based scheme to efficiently check whether one pattern (ε, δ)-covers another. We also exploit effective optimization strategies to further improve the computation efficiency. Our experimental results demonstrate that the devised data mining algorithm effectively and efficiently discovers the set of P-RFPs, which can substantially reduce the size of probabilistic frequent patterns.

This work extends the measure defined in deterministic databases to quantify the distance between two patterns in terms of their supporting transactions. Since the supports of patterns are random variables in the context of uncertain data, other distance measures, such as Kullback-Leibler divergence, might be applicable. We will study the effectiveness of probabilistic representative frequent patterns defined on different distance measures.

Chapter 3

Summarizing Probabilistic Frequent Patterns: A Fast Approach

Mining probabilistic frequent patterns from uncertain data has received a great deal of attention in recent years due to the wide applications. However, probabilistic frequent pattern mining suffers from the problem that an exponential number of result patterns are generated, which seriously hinders further evaluation and analysis. In this chapter, we focus on the problem of mining probabilistic representative frequent patterns (P-RFP), which is the minimal set of patterns with adequately high probability to represent all frequent patterns. Observing the bottleneck in checking whether a pattern can probabilistically represent another, which involves the computation of a joint probability of the supports of two patterns, we introduce a novel approximation of the joint probability with both theoretical and empirical proofs. Based on the approximation, we propose an Approximate P-RFP Mining (APM) algorithm, which effectively and efficiently compresses the set of probabilistic frequent patterns. To our knowledge, this is the first attempt to analyze the relationship between two probabilistic frequent patterns through an approximate approach. Our experiments on both synthetic and real-world datasets demonstrate that the APM algorithm accelerates P-RFP mining dramatically, achieving orders of magnitudes faster than an exact solution. Moreover, the error rate of APM is guaranteed to be very small when the database contains hundreds of transactions, which further affirms that APM is a practical solution for summarizing probabilistic frequent patterns.

3.1 Introduction

Data uncertainty is inherent in various applications such as sensor network monitoring, moving object tracking, and protein-protein interaction data [2]. It could be induced by different reasons including experimental error, artificial noise, and data incompleteness. Rather than cleaning the uncertain data using domain-specific rules, modeling the uncertainty of data is more rational in many applications, such as medical diagnosis and risk assessment. As a consequence, data mining over uncertain data has become an active research area recently. A survey of state-of-the-art uncertain data mining techniques may be found in [22].

As one of the most fundamental data mining tasks, frequent pattern mining has also been introduced into uncertain databases [3] and received a great deal of research attention [4, 5, 2, 6, 7, 8]. Generally, there exist two different definitions of frequent patterns in the context of uncertain data: *expected support-based frequent patterns* [3, 7] and *probabilistic frequent patterns* [4, 5]. Both definitions consider the *support* of a pattern as a discrete random variable. The former uses the expectation of the support as the measurement, while the latter considers the probability that the support of a pattern is no less than some specified minimum support threshold. Despite the different frequentness metrics employed, both the expected support-based frequent patterns and the probabilistic frequent patterns enjoy the anti-monotonic property [3, 4]. That is, if a pattern is frequent in an uncertain database, then all of its sub-patterns are frequent as well. This property leads to the generation of an exponential number of result patterns. The large number of discovered frequent patterns makes the understanding and further analysis of generated patterns troublesome. Therefore, similar to the counterpart of the problem in deterministic data, it is indeed important to find a small number of representative patterns to best approximate all other probabilistic frequent patterns.

Some initial research work has been undertaken to find a small set of representative patterns. For example, mining *probabilistic frequent closed patterns* over uncertain data has been studied in [10, 11, 12]. However, the number of probabilistic frequent closed patterns is still large because of the restrictive condition for a pattern being *closed*. For instance, in [12], the *closed probability* of a pattern is computed as the sum of the probabilities of the possible worlds of an uncertain database where the pattern is closed.

In the context of deterministic data, Xin et al. [23] has proposed the notion of an *ε-covered* relationship between patterns as a generalization of the concept of frequent closed patterns to further reduce the size of closed patterns. A pattern X_1 is ε-covered by another pattern X_2 if X_1 is a subset of X_2 and $(\text{Supp}(X_1) - \text{Supp}(X_2))/\text{Supp}(X_1) \leq \varepsilon$. The goal is then to find a minimal set of representative patterns that can ε-cover all frequent patterns.

Motivated by this idea in deterministic data, in our previous work, we have proposed to relax the restrictive condition of probabilistic frequent closed patterns to mine probabilistic representative frequent patterns (P-RFP) [40]. In particular, we extend the concept of ε-cover to define the *(ε, δ)-covered* relationship between probabilistic frequent patterns, addressing the fact that the support of a pattern becomes a discrete random variable in an uncertain database. Informally, a pattern X_1 is (ε, δ)-covered by another pattern X_2 in an uncertain database if X_1 is a subset of X_2, and the probability that the support distance between X_1 and X_2 is no greater than ε is no less than δ.

We have devised a dynamic programming-based approach to discover the minimal set of P-RFPs. Although this approach can compute exactly the probability that the support distance between two patterns is no greater than ε, it is not sufficiently efficient due to the bottleneck in examining whether a pattern (ε, δ)-covers another, which involves the computation of a joint probability of the supports of the two patterns. In this work, we analyze that the joint support probability follows a joint Poisson binomial distribution with both theoretical and empirical proofs. Based on the analysis, we propose an Approximate P-RFP Mining (APM) algorithm that performs outstandingly faster than the dynamic programming-based exact approach.

To our knowledge, this is the first attempt to analyze the relationship between two probabilistic frequent patterns through an approximate approach. Our experimental results show that our approach summarizes frequent patterns efficiently and effectively, and restores the patterns and their original frequency probability information with a guaranteed error bound. To summarize, our contributions are as follows.

- We construct a mathematical model for the joint probability of the supports of a pattern pair and study an approximation of the joint support probability.
- We develop an efficient algorithm to discover the minimal set of P-RFPs using accurate approximation techniques to estimate the probability that one pattern represents another.
- We conduct extensive experiments on both real-world and synthetic data to evaluate the performance of the proposed approach by comparing against an exact solution.

The remainder of the chapter is structured as follows. The next section reviews existing works related to this chapter. We define important concepts and provide the problem statement in Section 3.3. Section 3.4 describes the proposed data mining approach. The theoretical proof of the approximation of the joint support probability is demonstrated in Section 3.5. We evaluate the performance of the proposed approach in Section 3.6 and close this chapter with some conclusive remarks in Section 3.7.

3.2 Related Work

In this section, we review related research from two sub-areas: frequent pattern mining over uncertain data and frequent pattern summarization.

3.2.1 Frequent pattern mining over uncertain data

Mining frequent patterns from uncertain databases has been studied extensively in the past years. Existing work on frequent pattern mining from uncertain data falls into two categories: *expected support-based frequent pattern* *mining* [3, 6, 7] and *probabilistic frequent pattern mining* [4, 5]. The former utilizes the *expectation of support* as the frequentness metric. That is, a pattern is frequent only if its expected support is no less than a specified minimum expected support. The latter considers the *frequency probability* as the measurement, which refers to the probability that a pattern appears no less than a specified minimum support times. Thus, a pattern is frequent only if its frequency probability is no less than a specified minimum probability (i.e. $\Pr(\text{Supp}(X) \geq minsup) \geq minprob$).

There are three representative algorithms for mining expected support-based frequent patterns: UApriori [3], UFP-growth [6] and UH-Mine [7]. UApriori is the uncertain version of the well-known Apriori algorithm. Both UFP-growth and UH-Mine employ the divide-and-conquer framework that searches frequent patterns with depth-first strategy. For mining probabilistic frequent patterns, there are two representative algorithms: Dynamic Programming-based (DP) Apriori algorithm [4], and Divide-and-Conquer-based (DC) Apriori algorithm [5]. Tong et al. [2] verified that the two types of definitions of frequent patterns mined from uncertain data are closely related from a mathematical perspective and can be unified when the size of data is sufficiently large.

Considering that the support of a pattern in an uncertain database follows a Poisson binomial distribution, some approximate algorithms for mining probabilistic frequent patterns have been proposed as well. For example, both the Normal and Poisson distribution have been used to approximate the frequency probabilities of patterns [8, 28]. Compared with our work, existing approximate approaches focus on the approximation of the support probability of only one pattern. The approximation of joint probability of the supports of two patterns is much more challenging because the dependency of two random variables needs to be taken into account.

3.2.2 Frequent pattern summarization

Motivated by the fact that frequent pattern mining may generate an exponential number of patterns due to the anti-monotonicity, numerous research work has been dedicated to frequent pattern summarization, which aims to obtain a much smaller set of patterns to represent the complete set of frequent patterns. A variety of definitions have been proposed, such as maximal patterns [29], frequent closed patterns [30] and non-derivable patterns [31]. While all frequent patterns can be recovered from maximal patterns, the loss of support information is unacceptable in some circumstances. For frequent closed patterns, although the exact support of all frequent patterns can be preserved, the number of frequent closed patterns can still be tens of thousands, or even more. There are several generalizations of closed patterns, such as the pattern profiling-based approaches [35, 36, 37] and the support distance-based approaches [23, 38]. It was observed in [38] that the profile-based approaches [35, 36] have some drawbacks, such as no error guarantee on restored support. Hence, in our work, we borrow the framework of the support distance-based approaches to find probabilistic representative frequent patterns.

Some research work has been undertaken to summarize frequent patterns in the context of uncertain data. Tang and Peterson [11] proposed mining probabilistic frequent closed patterns, based on the concept called *probabilistic support*. Tong et al. [12] pointed out that frequent closed patterns defined on probabilistic support cannot guarantee the patterns are closed in possible worlds which contribute to their probabilistic supports. Instead, they defined the threshold-based frequent closed patterns over probabilistic data, which considers the probabilities of possible worlds where a pattern is closed. Our research relaxes the condition to further reduce the size of patterns by considering the probabilities of possible worlds where a pattern can ε-cover another one.

3.3 Background and Preliminary

In this section, we review the relevant concepts introduced in previous work and formally state the problem of probabilistic representative frequent pattern (P-RFP) mining.

Xin et al. [23] defined a robust distance measure between patterns in deterministic data.

Definition 3.1. (*distance measure*) *Given two patterns X_1 and X_2, the distance between them, denoted as* $\mathrm{dist}(X_1, X_2)$, *is defined as*

$$\mathrm{dist}(X_1, X_2) = 1 - \frac{|T(X_1) \cap T(X_2)|}{|T(X_1) \cup T(X_2)|}, \tag{3.1}$$

where $T(X_i)$ is the set of transactions supporting pattern X_i.

Then, an ε-cover relationship is defined on two patterns where one subsumes another.

Definition 3.2. (ε-*covered*) *Given a real number $\varepsilon \in [0, 1]$ and two patterns X_1 and X_2, we say X_1 is ε-covered by X_2 if $X_1 \subseteq X_2$ and* $\mathrm{dist}(X_1, X_2) \leq \varepsilon$.

It can be proved easily that, if X_2 ε-covers X_1, then

$$\frac{\mathrm{Supp}(X_1) - \mathrm{Supp}(X_2)}{\mathrm{Supp}(X_1)} \leq \varepsilon.$$

The goal of representative frequent pattern mining then becomes finding the minimal set of patterns that ε-cover all frequent patterns.

In the context of uncertain data, the support of a pattern, $\mathrm{Supp}(X_i)$, becomes a discrete random variable. Therefore, we cannot directly apply the ε-cover relationship to probabilistic frequent patterns. Before explaining how to extend the concept of ε-covered in the context of uncertain data, we examine an uncertain database where attributes are associated with existential probabilities.

Table 3.1 shows an uncertain transaction database where each transaction consists of a set of probabilistic items. For example, the probability that item a appears in the first transaction T_1 is 0.7. *Possible world semantics* are commonly used to explain the existence of data in an uncertain database. For example, the database in Table 3.1 has eight possible worlds, which are listed in Table 3.2. Each possible world is associated with an existential probability. For instance, the probability that the first possible world w_1 exists is $(1 - 0.7) \times (1 - 0.2) \times 1 \times (1 - 0.5) = 0.12$.

Considering that the occurrences of items in every possible world are deterministic, we can define the probabilistic distance between two probabilistic frequent patterns based on their distance in possible worlds.

ID	Transactions
T_1	a:0.7 b:0.2
T_2	a:1.0 c:0.5

Table 3.1: An example of attribute uncertainty.

ID	Possible World	Prob.
w_1	$\{T_1 : \phi, T_2 : \{a\}\}$	0.12
w_2	$\{T_1 : \{a\}, T_2 : \{a\}\}$	0.28
w_3	$\{T_1 : \{b\}, T_2 : \{a\}\}$	0.03
w_4	$\{T_1 : \{a,b\}, T_2 : \{a\}\}$	0.07
w_5	$\{T_1 : \phi, T_2 : \{a,c\}\}$	0.12
w_6	$\{T_1 : \{a\}, T_2 : \{a,c\}\}$	0.28
w_7	$\{T_1 : \{b\}, T_2 : \{a,c\}\}$	0.03
w_8	$\{T_1 : \{a,b\}, T_2 : \{a,c\}\}$	0.07

Table 3.2: An example of possible worlds.

Definition 3.3. (*probabilistic distance measure*) *Given an uncertain database D, and two patterns X_1 and X_2, let $\mathcal{PW} = \{w_1, \ldots, w_m\}$ be the set of possible worlds derived from D, the distance between X_1 and X_2 in a possible world $w_j \in \mathcal{PW}$ is*

$$\text{dist}(X_1, X_2; w_j) = 1 - \frac{|T(X_1; w_j) \cap T(X_2; w_j)|}{|T(X_1; w_j) \cup T(X_2; w_j)|} \tag{3.2}$$

where $T(X_i; w_j)$ is the set of transactions containing pattern X_i in the possible world w_j. Then, the probabilistic distance between X_1 and X_2, denoted by $\text{dist}(X_1, X_2)$, *is a random variable. The probability mass function of* $\text{dist}(X_1, X_2)$ *is:*

$$\Pr(\text{dist}(X_1, X_2) = d) = \sum_{\substack{w_j \in \mathcal{PW} \\ \text{dist}(X_1, X_2; w_j) = d}} \Pr(w_j) \tag{3.3}$$

That is, the probability that the distance between two probabilistic frequent patterns is d can be computed by the sum of the probabilities of corresponding possible worlds.

For example, consider the uncertain database in Table 3.1. Let $X_1 = \{a\}$ and $X_2 = \{a, b\}$. The probability that the distance between X_1 and

X_2 is equal to 0.5, $\Pr(\text{dist}(X_1, X_2) = 0.5)$, can be computed by adding the probabilities of the possible worlds w_4 and w_8. This is because only in the two possible worlds, the distance between the two patterns is 0.5. Therefore, $\Pr(\text{dist}(X_1, X_2) = 0.5) = 0.14$.

Based on the probabilistic distance measure, we define the ε-cover probability as follows.

Definition 3.4. (ε*-cover probability*) *Given an uncertain database D, two patterns X_1 and X_2, and a distance threshold ε, the ε-cover probability of X_1 and X_2 is defined as* $\Pr_{cover}(X_1, X_2; \varepsilon) = \Pr(\text{dist}(X_1, X_2) \leq \varepsilon)$.

Definition 3.5. ((ε, δ)*-covered*) *Given an uncertain database D, two patterns X_1 and X_2, a distance threshold ε and a ε-cover probability threshold δ, X_2 (ε, δ)-covers X_1 if and only if $X_1 \subseteq X_2$ and* $\Pr_{cover}(X_1, X_2; \varepsilon) \geq \delta$.

Our goal is then to obtain the minimal set of patterns that will (ε, δ)-cover all the probabilistic frequent patterns. The formal statement of the probabilistic representative frequent pattern (P-RFP) mining is as follows.

Definition 3.6. (*Problem Statement*) *Given an uncertain database D, a set of probabilistic frequent patterns $\mathcal{F}$, a probabilistic distance threshold ε and a ε-cover probability threshold δ, the problem of probabilistic representative frequent pattern (P-RFP) mining is to find the minimal set of patterns $\mathcal{R}$ so that, for any frequent pattern $X \in \mathcal{F}$, there exists a representative pattern $X' \in \mathcal{R}$ where X' (ε, δ)-covers X.*

It is obvious that when $\varepsilon = 0$, the probabilistic representative pattern set is equivalent to the set of probabilistic closed patterns, and when $\varepsilon = 1$, it is the same as probabilistic maximal pattern set.

3.4 Approximate P-RFP Mining

This section first describes the framework of our proposed approach. Then, we explain the details of the main steps of the Approximate P-RFP Mining (APM) algorithm.

3.4.1 Framework of APM

Before presenting the framework of our approximate approach for P-RFP mining, we develop some important lemmas between two patterns where one (ε, δ)-covers another.

Lemma 3.1. *Given an uncertain database D and two patterns X_1 and X_2 s.t. X_2 (ε, δ)-covers X_1, the distance between X_1 and X_2 in the possible world w_j can be represented by the support of the patterns in w_j:*

$$\text{dist}(X_1, X_2; w_j) = 1 - \frac{\text{Supp}(X_2; w_j)}{\text{Supp}(X_1; w_j)} \tag{3.4}$$

Lemma 3.2. *Given an uncertain database D and two patterns X_1 and X_2 s.t. X_2 (ε, δ)-covers X_1, the probabilistic distance* $\text{dist}(X_1, X_2)$ *can be represented by the support distribution of X_1 and X_2:*

$$\text{dist}(X_1, X_2) = 1 - \frac{\text{Supp}(X_2)}{\text{Supp}(X_1)} \tag{3.5}$$

Lemma 3.3. *Given an uncertain database D and two patterns X_1 and X_2 s.t. X_2 (ε, δ)-covers X_1, we have*

$$\Pr\left(\text{Supp}(X_2) \geq (1 - \varepsilon)\text{Supp}(X_1)\right) \geq \delta \tag{3.6}$$

These lemmas are obvious expansions of the concepts in deterministic data. The detailed proofs are stated in [40].

Lemma 3.4. *Given an uncertain database D, two patterns X_1 and X_2, a support threshold $minsup$ and a frequency probability threshold $minprob$, if X_2 (ε, δ)-covers X_1, and X_1 is a probabilistic frequent pattern w.r.t. $minsup$ and $minprob$, then X_2 is a probabilistic frequent pattern w.r.t. $(1 - \varepsilon)minsup$ and $(\delta \cdot minprob)$.*

Proof. Since X_1 is a probabilistic frequent pattern w.r.t. $minsup$ and $minprob$, we have $\Pr(\mathrm{Supp}(X_1) \geq minsup) \geq minprob$, which infers,

$$\Pr((1-\varepsilon)\mathrm{Supp}(X_1) \geq (1-\varepsilon)minsup) \geq minprob \tag{3.7}$$

From Lemma 3.3, we have,

$$\Pr(\mathrm{Supp}(X_2) \geq (1-\varepsilon)\mathrm{Supp}(X_1)) \geq \delta \tag{3.8}$$

Consider that the events in Equation 3.7 and 3.8 are independent, we have $\Pr(\mathrm{Supp}(X_2) \geq (1-\varepsilon)minsup) \geq \delta \cdot minprob$. That is, X_2 is a probabilistic frequent pattern w.r.t. $((1-\varepsilon)minsup)$ and $(\delta \cdot minprob)$. □

Denoting the set of probabilistic frequent patterns as F, Lemma 3.4 indicates that if pattern X can (ε, δ)-cover another pattern Y in F, then X must be probabilistic frequent w.r.t. $(1-\varepsilon)minsup$ and $minprob$. We call such a pattern pseudo probabilistic frequent and denote the set of pseudo probabilistic frequent patterns as $\hat{F}$. In order to achieve the minimal set of probabilistic representative frequent patterns, we have to find a subset of $\hat{F}$ that can (ε, δ)-cover all patterns of F. Given the two sets F and $\hat{F}$, our approach for P-RFP mining consists of the following two steps.

1. Generate the cover set for every pattern in $\hat{F}$. For each pattern X in $\hat{F}$, the cover set of X, denoted as $C(X)$, is a set of probabilistic frequent patterns in F that can be (ε, δ)-covered by X. That is, $C(X) \subseteq F$.

2. Find the minimal pattern set $R \subseteq \hat{F}$ to (ε, δ)-cover all probabilistic frequent patterns in F.

After finding the cover sets for patterns in $\hat{F}$ in the first step, the second step is equivalent to finding a minimal number of cover sets that cover all patterns in F. This is known as a set-covering problem, which is NP-hard. Similar to [38] and [40], we adopt a well-known greedy set-covering algorithm [39], which achieves polynomial complexity. Therefore, in the following, we focus on describing the first step, which generates the cover set for each pseudo probabilistic frequent pattern in $\hat{F}$.

3.4.2 Cover Set Generation

To generate the cover set for a pattern X_2 in $\hat{F}$, for each pattern X_1 in F such that $X_1 \subseteq X_2$, we need to check if X_2 (ε, δ)-covers X_1. That is, we need to examine whether the ε-cover probability between X_1 and X_2 is no less than δ (i.e., $\Pr(\text{dist}(X_1, X_2) \leq \varepsilon) \geq \delta$). According to Lemma 3.3, we have that the ε-cover probability $\Pr_{cover}(X_1, X_2; \varepsilon) = \Pr(\text{dist}(X_1, X_2) \leq \varepsilon)$ is equivalent to $\Pr(\text{Supp}(X_2) \geq (1-\varepsilon)\text{Supp}(X_1))$. Then, the ε-cover probability between X_1 and X_2 is equal to the following sum.

$$\sum_{l=minsup}^{|D|} \sum_{k=\lceil (1-\varepsilon)l \rceil}^{l} \Pr(\text{Supp}(X_1) = l, \text{Supp}(X_2) = k) \tag{3.9}$$

To compute the ε-cover probability to find out whether it is no less than δ, we introduce the joint support probability distribution as follows.

Definition 3.7. (*joint support probability*) *Given an uncertain database D and patterns X_1 and X_2, the joint support probability mass function is*

$$\Pr\left(\text{Supp}(X_1) = l, \text{Supp}(X_2) = k\right) = \sum_{\substack{w_i \in \mathcal{PW}, \\ \text{Supp}(X_1; w_i) = l \\ \text{Supp}(X_2; w_i) = k}} \Pr(w_i)$$

Although Definition 3.7 implies a brute-force solution, it is not feasible to implement because the number of possible worlds is exponential. Therefore, we establish the following approximation of joint support probability.

Theorem 3.1. *Given an uncertain database D and patterns X_1 and X_2, the joint support probability can be approximated by a bivariate normal distribution, which means*

$$\Pr(\text{Supp}(X_1) = l, \text{Supp}(X_2) = k) \approx \phi\left(\mathbf{\Sigma}^{-\frac{1}{2}} (\mathbf{X} - \boldsymbol{\mu})\right) \tag{3.10}$$

where $\mathbf{X} = \begin{bmatrix} l & k \end{bmatrix}^\top$, *$\boldsymbol{\mu}$ is the vector of mean values of* $\text{Supp}(X_1)$ *and* $\text{Supp}(X_2)$, *and $\mathbf{\Sigma}$ is the covariance matrix of X_1 and X_2.*

Algorithm 3.1 APM Algorithm Framework

```
Input: $D$, $F$, $\hat{F}$, $\varepsilon$ and $\delta$
Output: Minimal P-RFP Set $R$
1: $R \leftarrow \emptyset$
2: $CoverSets \leftarrow \emptyset$
3: for all $X_2 \in \hat{F}$ do
4:    $NoCoverSet \leftarrow \emptyset$
5:    for all $X_1 \in F$ such that $X_1 \subseteq X_2$ do
6:       if $isCover(X_1, X_2) = True$ then
7:          $CoverSets[X_2].add(X_1)$
8:       else
9:          $NoCoverSet.add(X_1)$
10: $R \leftarrow setCover(CoverSets, F)$
11: return $R$
```

Theorem 3.1 provides a solution to compute the joint support probability of a pair of patterns via normal distribution, rather than mining in the complete database. The detailed theoretical proof is elaborated in Section 3.5, and the empirical simulation is illustrated in Section 3.6. Similar to univariate normal distribution, we can optimize our approach with the well-known 3σ property [41].

Corollary 3.1. *Given an uncertain database D, and patterns X_1 and X_2, let the mean value and variance of* $\mathrm{Supp}(X_j)$ *be μ_j, σ_j^2, $j = 1, 2$, $l_1 = \max\{minsup, \mu_1 - 3\sigma_1\}$, $l_2 = \min\{|D|, \mu_1 + 3\sigma_1\}$, $k_1 = \max\{\lceil(1-\varepsilon)l\rceil, \mu_2 - 3\sigma_2\}$, and $k_2 = \min\{l, \mu_2 + 3\sigma_2\}$, then*

$$\sum_{l=minsup}^{|D|} \sum_{k=\lceil(1-\varepsilon)l\rceil}^{l} \Pr(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k)$$
$$\approx \sum_{l=l_1}^{l_2} \sum_{k=k_1}^{k_2} \Pr(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k) \tag{3.11}$$

Note that for better precision, we use σ_1 to calculate the support lower bound and upper bound for both X_1 and X_2 because the contour of bivariate normal distribution is an ellipse, and σ_1 is the length of semi major axis. Based on Corollary 3.1, we can reduce the computational complexity of ε-

cover probability from $O(|D|^2)$ to $O(9\sigma_1^2)$ significantly. To accelerate the progress of cover set generation further, we also take advantage of some optimization strategies in [40].

Lemma 3.5. *Given an uncertain database D, two patterns X_1 and X_2 s.t. $X_1 \subseteq X_2$, and a probabilistic distance threshold ε, $\mathrm{Pr}_{cover}(X_1, X_2; \varepsilon)$ computed on D is equal to that on $D(X_1)$, where $D(X_1)$ is $\{t|P(X_1 \subseteq t) > 0, t \in D\} \subseteq D$.*

Lemma 3.5 is intuitive because only the transactions supporting at least the sub-pattern X_1 will contribute to the value of probabilistic distance, which in turn affects the ε-cover probability. This lemma allows us to compute the ε-cover probability on a projected sub-database, which significantly reduces the runtime of computation.

Lemma 3.6. *Given an uncertain database D and two patterns X_1 and X_2 s.t. $X_1 \subseteq X_2$, if X_2 (ε, δ)-covers X_1, then $\forall X$ s.t. $X_1 \subseteq X \subseteq X_2$, we have X_2 (ε, δ)-covers X.*

According to Lemma 3.6, we have the following corollary.

Corollary 3.2. *Given an uncertain database D and two patterns X_1 and X_2, $X_1 \subseteq X_2$, if X_2 cannot (ε, δ)-cover X_1, then $\forall X \subseteq X_1$, X_2 cannot (ε, δ)-cover X.*

Lemma 3.6 and Corollary 3.2 reduce the number of pattern pairs, for which the ε-cover probability needs to be computed. The complete proofs of Lemma 3.5, Lemma 3.6 and Corollary 3.2 are stated in [40].

3.4.3 APM Algorithm

The overall framework of our APM algorithm is shown in Algorithm 3.1. From line 3 to line 9, we find the cover set for each pseudo probabilistic frequent pattern X_2 in $\hat{F}$. The most important step is to check whether X_2 covers X_1 in F (line 6). The details of the function *isCover* is illustrated in Algorithm 3.2, where lines $1-3$ implement the optimization stated by Lemma 3.6, and lines $4-6$ apply the Corollary 3.2. Finally, from line 7 to line 12, we use the approximation-based scheme to compute the ε-cover probability. As mentioned before, the function *setCover* in Algorithm 3.1 is solved using the greedy algorithm in [39].

Algorithm 3.2 Function *isCover*

```
Input: $X_1, X_2$
Output: If $X_2$ $(\varepsilon, \delta)$-covers $X_1$, then return True, else False
 1: for all $X \in CoverSets[X_2]$ do
 2:     if $X \subseteq X_1$ then
 3:         return True
 4: for all $X \in NoCoverSet[X_2]$ do
 5:     if $X \supseteq X_1$ then
 6:         return False
 7: $l_1 = \max\{minsup, \mu_1 - 3\sigma_1\}$
 8: $l_2 = \min\{|D(X_1)|, \mu_1 + 3\sigma_1\}$
 9: $k_1 = \max\{\lceil (1-\varepsilon)l \rceil, \mu_2 - 3\sigma_2\}$
10: $k_2 = \min\{l, \mu_2 + 3\sigma_2\}$
11: for $l = l_1$ to $l_2$ do
12:     for $k = k_1$ to $k_2$ do
13:         $P_{cover} += \Pr(\text{Supp}(X_1) = l, \text{Supp}(X_2) = k)$
14:         if $P_{cover} \geq \delta$ then
15:             return True
16: return False
```

3.5 Approximation of Joint Support Probability

In this section, we present the detailed proof of the bivariate normal distribution-based approximation of the joint support probability of two patterns. Given an uncertain database D, two patterns X_1 and X_2, s.t. $X_1 \subseteq X_2$, and the corresponding support random variables, denoted as $X_{n_{(1)}}$ and $X_{n_{(2)}}$ hereafter, where n is the size of D, our goal is to prove that $[X_{n_{(1)}} \ X_{n_{(2)}}]^\top$ converges to a bivariate normal distribution when $n \to \infty$.

3.5.1 Preparation

Suppose the existence probabilities of patterns X_1 and X_2 in the ith transaction t_i are $p_{ni_{(1)}}$ and $p_{ni_{(2)}}$, then

$$X_{ni_{(j)}} \sim \text{Bern}\left(p_{ni_{(j)}}\right), j = 1, 2$$

because $X_{ni_{(j)}}$ follows Bernoulli distribution.

The support of pattern X_j, $X_{n_{(j)}}$, can be computed as $X_{n_{(j)}} = \sum_{i=1}^{n} X_{ni_{(j)}}, j =$

Situation	Probability
$X_1 \nsubseteq t_i, X_2 \nsubseteq t_i$	$1 - p_{ni_{(1)}}$
$X_1 \subseteq t_i, X_2 \nsubseteq t_i$	$p_{ni_{(1)}} - p_{ni_{(2)}}$
$X_1 \subseteq t_i, X_2 \subseteq t_i$	$p_{ni_{(2)}}$

Table 3.3: All possible situations of X_1 and X_2 in t_i.

$1,2$. Since both $X_{n_{(1)}}$ and $X_{n_{(2)}}$ follow Poisson binomial distribution, the mean value and variance of $X_{n_{(j)}}$ are

$$\mu_{n_{(j)}} = \sum_{i=1}^{n} p_{ni_{(j)}}, \quad \sigma^2_{n_{(j)}} = \sum_{i=1}^{n} p_{ni_{(j)}} \left(1 - p_{ni_{(j)}}\right), \quad j = 1,2$$

The covariance of $X_{n_{(1)}}$ and $X_{n_{(2)}}$ is

$$\mathrm{Cov}\left(X_{n_{(1)}}, X_{n_{(2)}}\right) = \sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{Cov}(X_{ni_{(1)}}, X_{nj_{(2)}})$$

Table 3.3 illustrates all possible existence situations of patterns X_1 and X_2 in transaction t_i. Assuming for any i and j such that $i \neq j$, $X_{ni_{(1)}}$ and $X_{nj_{(2)}}$ are independent, we have $\mathrm{Cov}(X_{ni_{(1)}}, X_{nj_{(2)}}) = 0$. Table 3.3 indicates that $\mathrm{E}(X_{ni_{(1)}} \cdot X_{ni_{(2)}}) = p_{ni_{(2)}}$ and $\mathrm{Cov}\left(X_{n_{(1)}}, X_{n_{(2)}}\right) = \sum_{i=1}^{N}\left(\left(1 - p_{ni_{(1)}}\right) p_{ni_{(2)}}\right)$

For brevity, let $\mathbf{X}_{ni} = \begin{bmatrix} X_{ni_{(1)}} & X_{ni_{(2)}} \end{bmatrix}^\top$ and denote the sum of $\mathbf{X}_{ni}$ over database as

$$\mathbf{X}_n = \sum_{i=1}^{n} \mathbf{X}_{ni} = \begin{bmatrix} X_{n_{(1)}} & X_{n_{(2)}} \end{bmatrix}^\top \tag{3.12}$$

Then, $\{\mathbf{X}_n\}, n = 1, 2, \cdots$ is a sequence of random vectors:

$$\begin{aligned}
\mathbf{X}_1 &= \mathbf{X}_{11} \\
\mathbf{X}_2 &= \mathbf{X}_{21} + \mathbf{X}_{22} \\
&\cdots \\
\mathbf{X}_k &= \mathbf{X}_{k1} + \mathbf{X}_{k2} + \mathbf{X}_{k3} + \cdots + \mathbf{X}_{kk} \\
&\cdots
\end{aligned}$$

$\{\mathbf{X}_{ni}\}$ is called a triangular array, which is manipulated commonly in the study of sum of independent vectors.

Until now, we have laid the groundwork in preparation of the proof that $\{\mathbf{X}_n\}$ holds asymptotic normality in the next subsection.

3.5.2 Proof of Approximation

With the aforementioned concepts, we propose the following theorem, from which Theorem 3.1 can be induced directly.

Theorem 3.2. *Let $\{\mathbf{X}_{ni} \in \mathbb{R}^2\}$, $n = 1, 2, \ldots$, $i = 1, 2, \ldots, n$ be a triangular array of random vectors such that: (1) for all $n \geq 1$, $\mathbf{X}_{n1}, \ldots, \mathbf{X}_{nn}$ are independent, (2) for all $1 \leq i \leq n$, $\mathbf{X}_{ni}$ follows a bivariate Bernoulli distribution, $\mathbf{X}_n = \sum_{i=1}^n \mathbf{X}_{ni}$, then*

$$\mathbf{\Sigma}_n^{-\frac{1}{2}} (\mathbf{X}_n - \boldsymbol{\mu}_n) \xrightarrow{d} \mathbf{N}(0, I) \tag{3.13}$$

where $\boldsymbol{\mu}_n$ and $\mathbf{\Sigma}_n$ are the mean and covariance of $\mathbf{X}_n$, respectively.

Theorem 3.2 provides an important bridge between the joint support distribution of a pair of patterns and the bivariate normal distribution. Noting that suppose the cumulative density functions of X_n and X are F_n and F, $\mathbf{X}_n \xrightarrow{d} \mathbf{X}$ if and only if for any continuous point x of F, $\lim_{n\to\infty} F_n(x) = F(x)$. Before giving the detailed proof of theorem3.2, two necessary lemmas should be presented first.

Lemma 3.7. *Let $\mathbf{X}_{ni} \in \mathbb{R}^{m_i}$, $i = 1, \ldots, k_n$, be independent random vectors with $m_i \leq m$ (a fixed integer), $n = 1, 2, \ldots, k_n \to \infty$ as $n \to \infty$, and $\inf_{i,n} \lambda_{\min}[\mathrm{Cov}(\mathbf{X}_{ni})] > 0$, where $\lambda_{\min}[A]$ is the smallest eigenvalue of A. Let $c_{ni} \in \mathbb{R}^{m_i}$ be vectors such that*

$$\lim_{n\to\infty} \left(\frac{\max_{1\leq i\leq k_n} \|c_{ni}\|^2}{\sum_{i=1}^{k_n} \|c_{ni}\|^2} \right) = 0 \tag{3.14}$$

If $\sup_{i,n} \mathrm{E}\|\mathbf{X}_{ni}\|^{2+\delta} < \infty$ for some $\delta > 0$, then

$$\frac{\sum_{i=1}^{k_n} c_{ni}^T (\mathbf{X}_{ni} - \mathrm{E}\mathbf{X}_{ni})}{\left[\sum_{i=1}^{k_n} \mathrm{Cov}({c_{ni}}^T \mathbf{X}_{ni})\right]^{1/2}} \xrightarrow{d} \mathbf{N}(0, I) \tag{3.15}$$

More details of Lemma 3.7 are stated in [41]. Given a sequence of random vectors $\{\mathbf{X}_n\}$, Lemma3.7 provides a solution to prove the convergence of all possible linear combinations of $\{\mathbf{X}_n\}$. Nevertheless, it is not equivalent to the convergence of the random vector itself. Hence, we refer to the next lemma to bridge the gap.

Lemma 3.8 (**Cramér-Wold Theorem**[42]). *Suppose that $\mathbf{X}_n$ and $\mathbf{X}$ are k-dimensional random vectors. Then $\mathbf{X}_n \overset{d}{\to} \mathbf{X}$ if and only if*

$$t^\top \mathbf{X}_n \overset{d}{\to} t^\top \mathbf{X} \tag{3.16}$$

for all vectors $t \in \mathbb{R}^k$.

The Cramér-Wold theorem states that the convergence of a k-dimensional random vector is closely related to the totality of its one-dimensional projections. With Lemma 3.7 and Lemma 3.8, the complete proof of Theorem 3.2 is as follows.

Proof of Theorem 3.2. Let $k_n = n$, and $\forall i, 1 \leq i \leq n, m_i = 2$.

The determinant of covariance matrix is

$$\det[\mathrm{Cov}(\mathbf{X}_{ni})] = (1-\rho)\sigma^2_{ni_{(1)}}\sigma^2_{ni_{(2)}} \tag{3.17}$$

Considering that X_1 and X_2 are two patterns with different parameters, the correlation coefficient between their support distribution satisfies $0 < \rho < 1$. Consequently, $\mathrm{Cov}(\mathbf{X}_{ni})$ is a positive definite matrix and $\inf_{i,n} \lambda_{\min}[\mathrm{Cov}(\mathbf{X}_{ni})] > 0$.

Let $\delta = 2$, since all components of $\mathbf{X}_{ni}$ are no greater than the size of database n, we have

$$\|\mathbf{X}_{ni}\|^4 = \left[\left(\mathbf{X}_{ni_{(1)}}\right)^2 + \left(\mathbf{X}_{ni_{(2)}}\right)^2\right]^2 \leq 4n^4 \leq \infty \tag{3.18}$$

For all $i = 1, 2, \ldots, n$, assume $c_{ni} = \begin{bmatrix} c_1 & c_2 \end{bmatrix}^\top$, where $c_1, c_2 \in \mathbb{R}$. Then,

$$\lim_{n\to\infty}\left(\frac{\max_{1\leq i\leq n}\|c_{ni}\|^2}{\sum_{i=1}^n \|c_{ni}\|^2}\right) = \lim_{n\to\infty}\left(\frac{1}{n}\right) = 0$$

Therefore, Lemma 3.7 indicates that

$$\frac{\sum_{i=1}^{k_n} c_{ni}^T(\mathbf{X}_{ni} - \mathrm{E}\mathbf{X}_{ni})}{\left[\sum_{i=1}^{k_n} \mathrm{Cov}(c_{ni}^T \mathbf{X}_{ni})\right]^{\frac{1}{2}}} \xrightarrow{d} \mathbf{N}(0, I)$$

With Lemma 3.8, finally we have

$$\boldsymbol{\Sigma}_n^{-\frac{1}{2}} (\mathbf{X}_n - \boldsymbol{\mu}_n) \xrightarrow{d} \mathbf{X}$$

Hence, we prove Theorem 3.2. □

To further improve the accuracy of our approximation, we should take the continuity correction [43] into account, because we are using a continuous distribution to approximate a discrete distribution. The final equation needs to be changed slightly as follows.

$$\Pr\left(\mathrm{Supp}(X_1) = l, \mathrm{Supp}(X_2) = k\right) \approx \phi\left(\frac{\mathbf{X} + 0.5 - \boldsymbol{\mu}}{\sqrt{|\boldsymbol{\Sigma}|}}\right)$$

where $\mathbf{X} = \begin{bmatrix} l & k \end{bmatrix}^\top$, $\boldsymbol{\mu}$ is the vector of mean values of $\mathrm{Supp}(X_1)$ and $\mathrm{Supp}(X_2)$, and $\boldsymbol{\Sigma}$ is the corresponding covariance matrix. Since Theorem 3.2 is equivalent to Theorem 3.1, it is served as a solid theoretical background to support our algorithm. We will demonstrate the empirical proof and assess our approach subsequently.

3.6 Performance Study

In this section, we first empirically study the performance of the joint support probability approximation, then evaluate the effectiveness and efficiency of the APM algorithm.

3.6.1 Empirical study of approximation

We evaluate the accuracy of the approximation of joint support probability with simulation. Two probability support vectors of a pattern X_1 and its

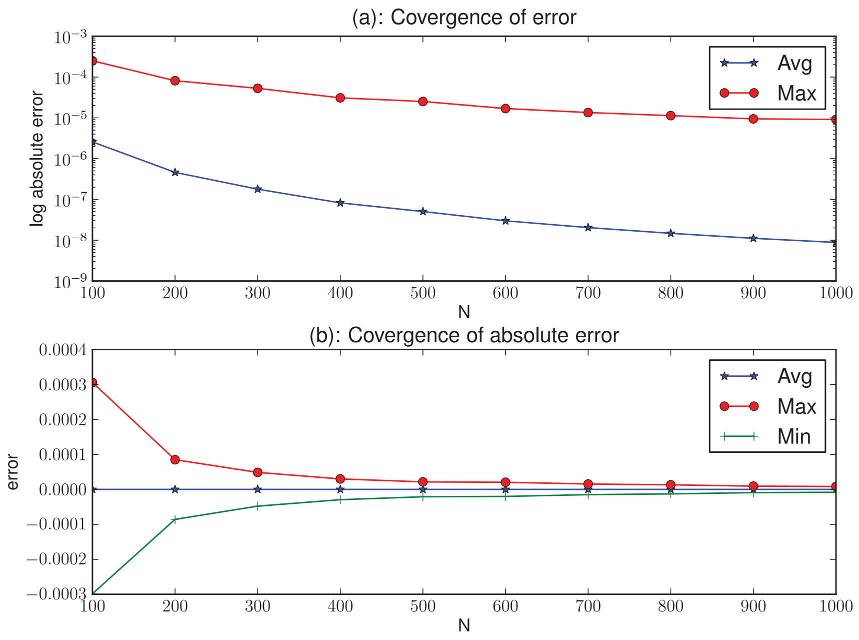


Figure 3.1: Empirical proof of approximation.

super pattern X_2 are constructed from a synthetic uncertain database with $N = 100, 200, \ldots, 1000$ transactions. The uncertainty is incorporated according to the standard normal distribution. Then, we perform both the exact and approximate algorithms to obtain the joint support probability on the sample space.

For each setting of N, we run the experiment for 500 times. Figure 3.1 (a) shows the average and maximum absolute error (e.g., $|\Pr_a(x, y) - \Pr_e(x, y)|$, where $\Pr_a$ and $\Pr_e$ are the approximate and exact probability) w.r.t. the variation of the database size. Figure 3.1 (b) demonstrates the average, minimum and maximum error (e.g., $\Pr_a(x, y) - \Pr_e(x, y)$) between the real and approximate value w.r.t. the variation of the database size. It is shown that the error decreases rapidly when N is increasing. When $N = 500$, which is much less than the size of a regular database, the average absolute error is less than 10^{-7}.

Dataset	Number of Transactions	Number of Items	Average Length
IIP	35161	467	4.0
Retail	88162	16470	10.3
Chess	3196	75	6.7

Table 3.4: Characteristics of Datasets.

3.6.2 Result analysis

3.6.2.1 Data sets

Three datasets have been used in our experiments. Two of them, the Retail dataset and the Chess dataset, are from the Frequent Itemset Mining (FIMI) Dataset Repository.[1] These are standard datasets used for frequent pattern mining in deterministic databases. In order to bring uncertainty into the datasets, we synthesize an existential probability for each item based on a Gaussian distribution with the mean of 0.9 and the variance of 0.125. The two datasets are uncertain databases with uncertainties associated with attributes.

The other one is the iceberg sighting record from 1993 to 1997 on the North Atlantic from the International Ice Patrol (IIP) Iceberg Sightings Database, which is also used in the experiments in last chapter.[2] Each transaction in the database contains the information of date, location, size, shape, reporting source and a confidence level. There are six possible attributes of the confidence level, R/V (Radar and visual), R (Radar only), V (Visual), MEA (Measured), EST (Estimated) and GBL (Garbled), which indicate different reliabilities. We convert the confidence levels to probabilities 0.8, 0.7, 0.6, 0.5, 0.4 and 0.3, respectively. This dataset forms an uncertain database that associates uncertainties to tuples. The statistics of the datasets are shown in Table 3.4.

[1]http://fimi.cs.helsinki.fi/data/

[2]http://nsidc.org/data/g00807.html

3.6.2.2 Performance of APM algorithm

To analyze the performance of the APM algorithm, we carry out two sets of experiments. In the first set, we compare the effectiveness and efficiency of the APM against the dynamic programming-based exact method [40]. Due to the low efficiency of the exact method, we randomly select 500 transactions respectively from two datasets, Retail and IIP. The sizes of FP (the set of probabilistic frequent patterns), DP (the set of P-RFPs mined by the dynamic programming-based approach), and APM (the set of P-RFPs produced by the APM algorithm) with respect to the variations of $minsup$, $minprob$, ε and δ, on the two datasets are shown respectively in Figures 3.2 and 3.3. The default values of the four parameters are set to 0.5%, 0.8, 0.2 and 0.5, respectively. It can be observed that the result of the APM algorithm is very close to that of the exact method, while both of them are able to reduce the size of the probabilistic frequent pattern set effectively. The runtime of two methods are demonstrated in Figures 3.4 and 3.5. It is impressive that the APM algorithm accelerates P-RFP mining significantly.

Then, we examine the performance of the APM algorithm on the complete database of IIP, Retail, and Chess datasets. The comparisons between the number of P-RFPs and the number of frequent patterns are illustrated in Figures 3.6, 3.7 and 3.8. These charts indicate that the APM algorithm can reduce the size of frequent pattern set effectively. Figures 3.9,3.10 and 3.11 show the runtime vs. $minsup$, $minprob$, ε, and δ curves of the APM algorithm without and with the 3σ pruning technique, which are called APM and $APM + Pruning$, on the three datasets, respectively. The default values of the four parameters for the IIP and Retail datasets are 0.5%, 0.8, 0.2, and 0.5. For the chess dataset, the default parameters are 0.6%, 0.5, 0.15, and 0.8. It is intuitive that, when ε is increasing or $minsup$, $minprob$ and δ are decreasing, the runtime will increase because more pattern pairs are engaged in the cover probability checking. We can find that the APM algorithm can mine P-RFP set quickly, and the pruning technique accelerates it even further.

3.7 Conclusions

Due to the downward closure property, the number of probabilistic frequent patterns mined over uncertain data can be so large that they hinder further analysis and exploitation. This chapter proposes the APM algorithm, which aims to efficiently and effectively find a small set of patterns to represent the complete set of probabilistic frequent patterns. To address the high computational complexity in examining the joint support probability, we introduce an approximation of the joint support probability with both theoretical and empirical proofs. Our experimental results demonstrate that the devised algorithm can substantially reduce the size of probabilistic frequent patterns efficiently.

This work adopts the measure defined in deterministic databases to quantify the distance between two patterns in terms of their supporting transactions. Since the supports of patterns are random variables in the context of uncertain data, other distance measures, such as Kullback-Leibler divergence, might be applicable. As an ongoing work, we will study the effectiveness of probabilistic representative frequent patterns defined on different distance measures.

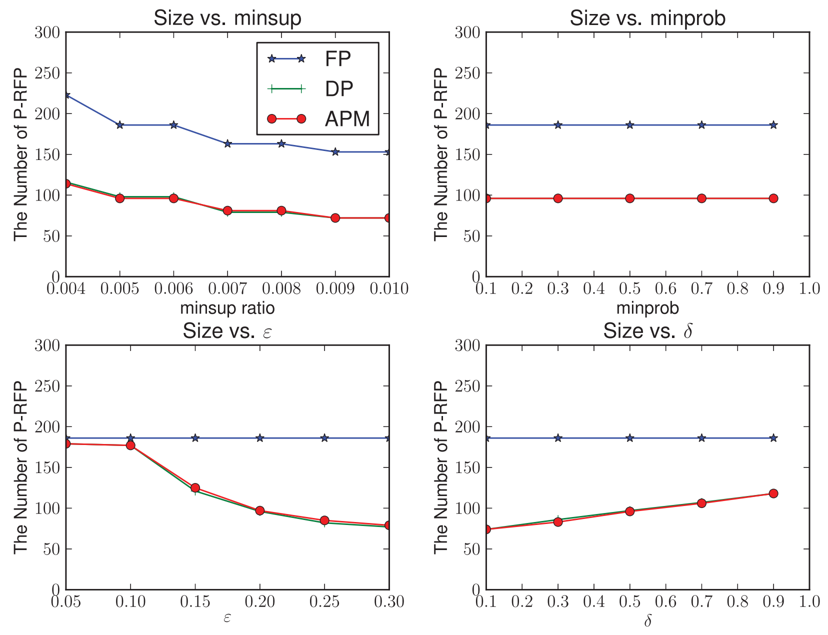


Figure 3.2: The Number of P-RFP on IIP-500.

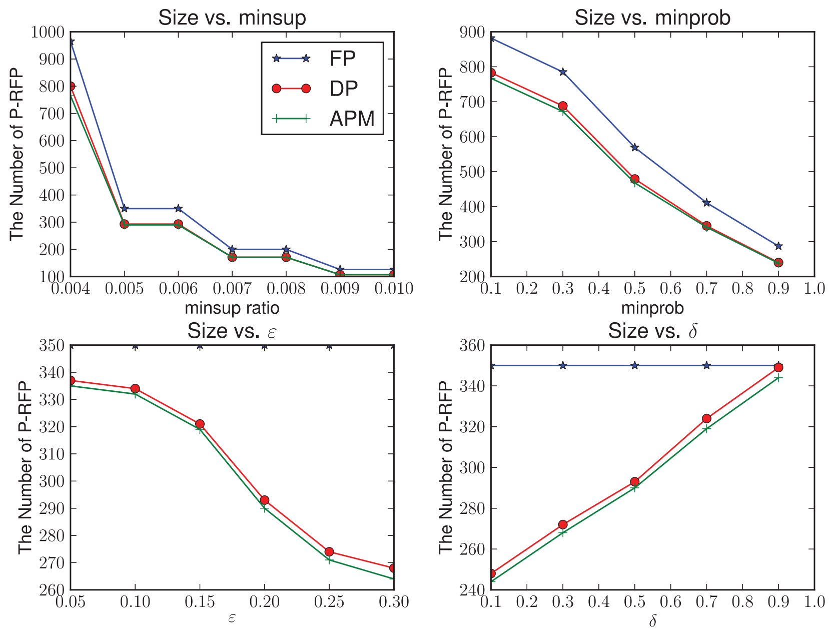


Figure 3.3: The Number of P-RFP on Retail-500.

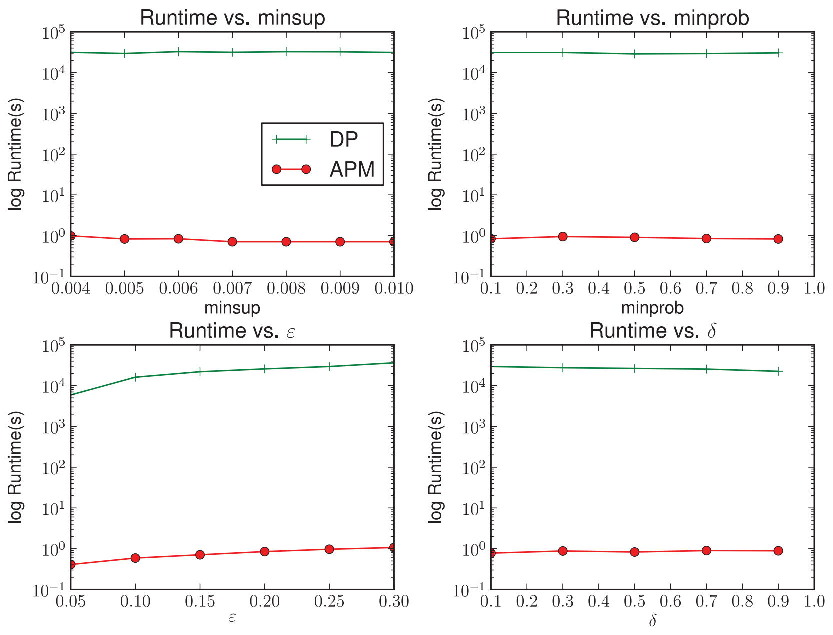


Figure 3.4: Log Runtime on IIP-500.

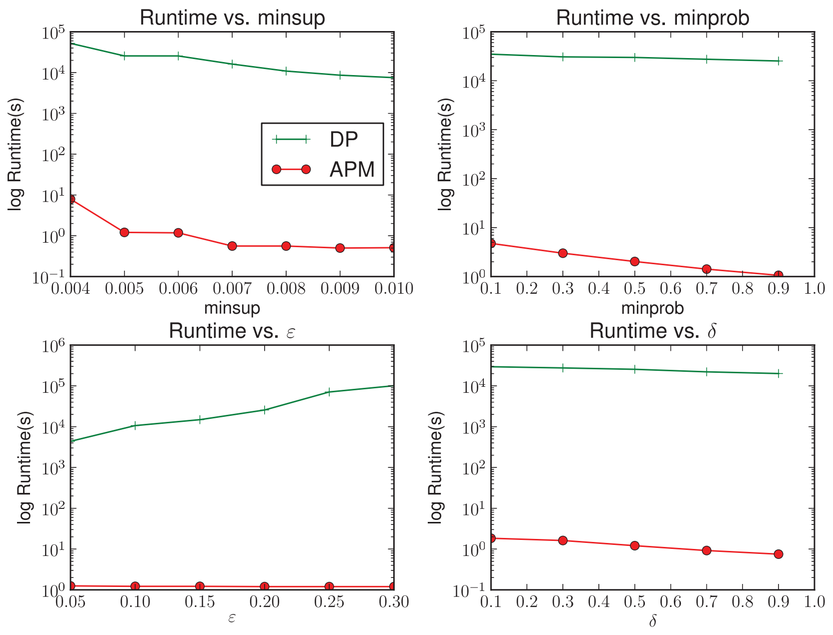


Figure 3.5: Log Runtime on Retail-500.

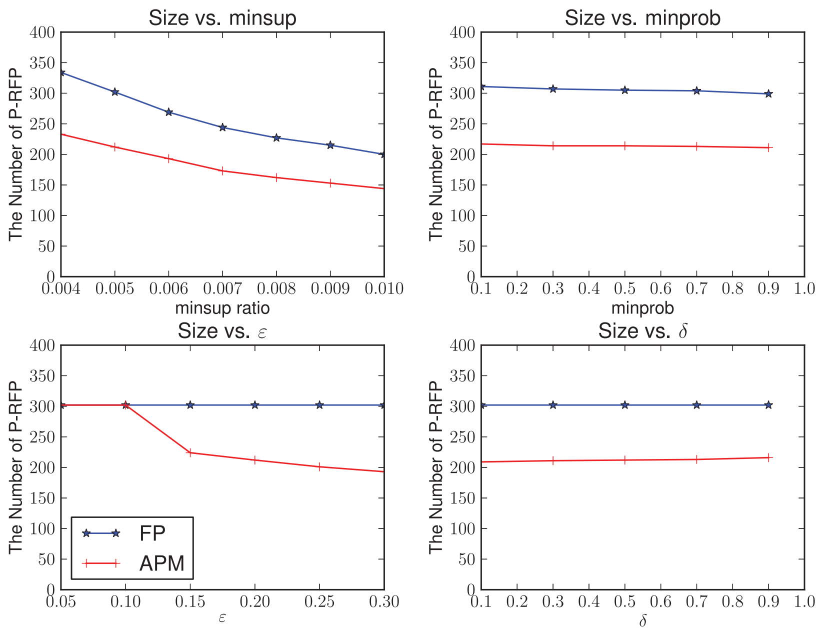


Figure 3.6: The Number of P-RFP on IIP.

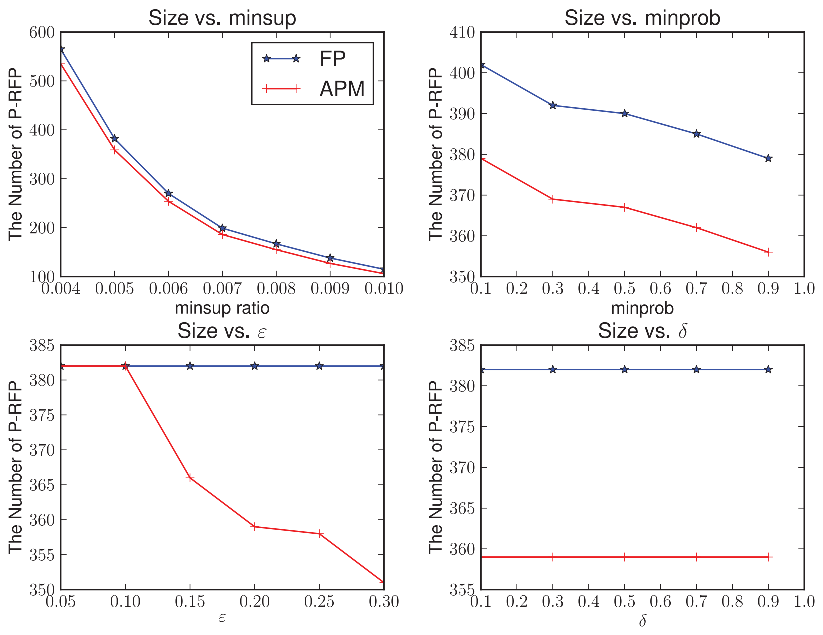


Figure 3.7: The Number of P-RFP on Retail.

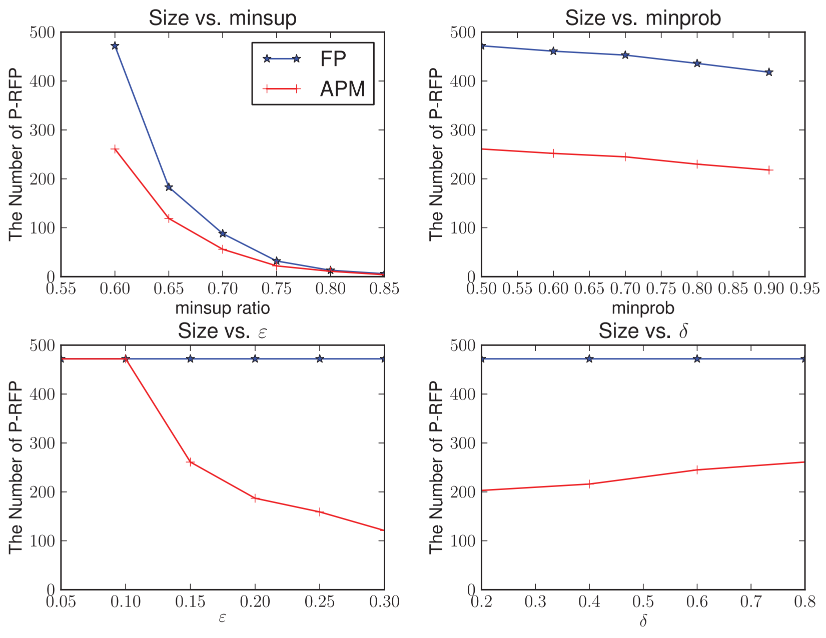


Figure 3.8: The Number of P-RFP on Chess.

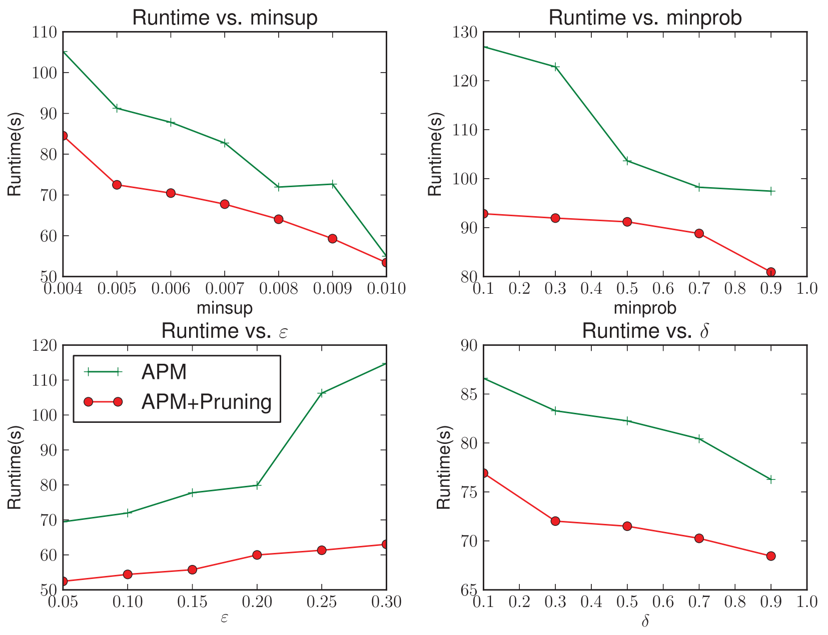


Figure 3.9: Runtime on IIP.

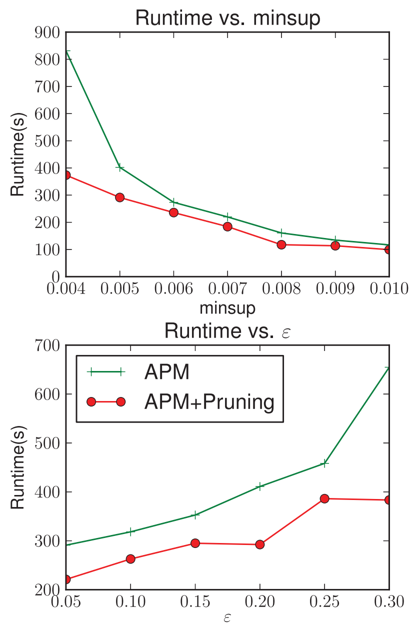


Figure 3.10: Runtime on Retail.

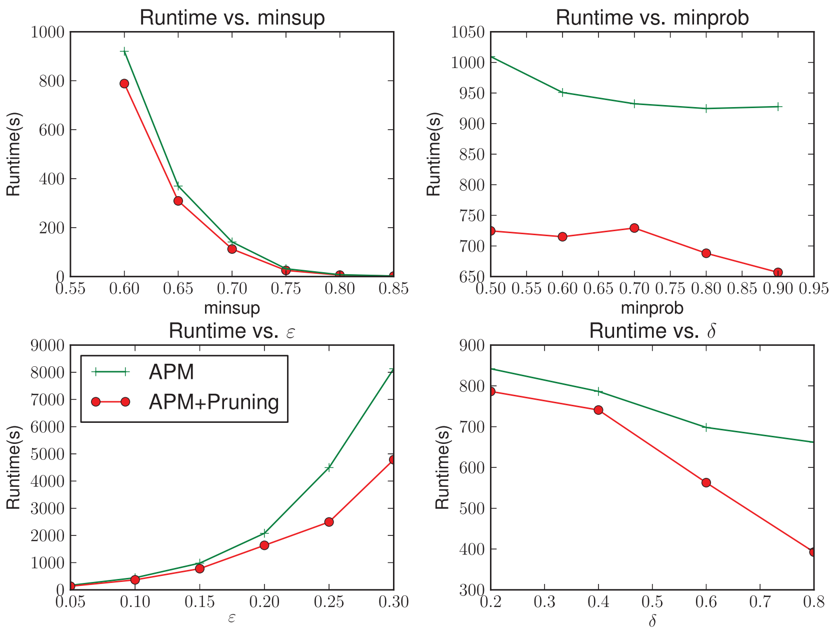


Figure 3.11: Runtime on Chess.

Chapter 4

Summarizing Uncertain Transaction Databases by Probabilistic Tiles

Transaction data mining is ubiquitous in various domains and has been researched extensively. In recent years, observing that uncertainty is inherent in many real world applications, uncertain data mining has attracted much research attention. Among the research problems, summarization is important because it produces concise and informative results, which facilitates further analysis. However, there are few works exploring how to effectively summarize uncertain transaction data. In this chapter, we formulate the problem of summarizing uncertain transaction data as *Minimal Probabilistic Tile Cover* Mining, which aims to find a high-quality probabilistic tile set covering an uncertain database with minimal cost. We define the concept of *Probabilistic Price* and *Probabilistic Price Order* to evaluate and compare the quality of tiles, and propose a framework to discover the minimal probabilistic tile cover. The bottleneck is to check whether a tile is better than another according to the *Probabilistic Price Order*, which involves the computation of a joint probability. We prove that it can be decomposed into independent terms and calculated efficiently. Several optimization techniques are devised to further improve the performance. Experimental results on both synthetic and real world datasets demonstrate the conciseness of the produced tiles and the effectiveness and efficiency of our approach.

Transaction ID	Items
t_1	a b
t_2	a

(a) A deterministic database.

Transaction ID	Items
t_1	a:0.6 b:0.8
t_2	a:1.0 c:0.42

(b) An uncertain database.

Table 4.1: Examples of transaction databases.

4.1 Introduction

Transaction data is ubiquitous throughout various domains, such as retail business, bioinformatics and text analysis [44]. It is also closely related to other data representations, such as binary matrix and bipartite graph, which further extends its potential applicability. Thus, mining transaction databases is an important and active research area. However, the analysis of transaction databases suffer from the issues of data abundance and redundancy, especially in recent years because of the emergence of big data.

Table 4.1 (a) is an example of transaction database. Each transaction is a pair comprising an ID and an itemset, such as $\langle t_1, \{a, b\}\rangle$ and $\langle t_2, \{a\}\rangle$. For brevity, we indicate a transaction by its ID hereafter.

A great deal of research effort has been dedicated to summarizing transaction databases with a small number of meaningful representatives. In order to capture the correlation between transactions and items simultaneously, many state-of-the-art algorithms address this problem by producing a set of representatives, in which a representative is a pair of a transaction set and an itemset, such as *tile* in [45], *hyperrectangle* in [46] and *approximate binary pattern* in [47]. Since these terms indicate a similar concept, we use the name *tile* in this chapter, which is proposed first. For example, given the transaction database $\mathcal{D}$ in Table 4.1 (a), both $R_1 = \{t_1, t_2\} \times \{a\}$ and $R_2 = \{t_1\} \times \{b\}$ are possible tiles. $\mathcal{D}$ can be covered by $\mathcal{R} = \{R_1, R_2\}$, because $\mathcal{D} \subseteq R_1 \cup R_2$.

Recently, uncertain data mining has attracted much research attention, because data uncertainty is inherent in many real world applications, such as sensor network monitoring, moving object tracking and protein-protein interaction data [2]. Table 4.1 (b) shows an example of an uncertain transaction database, in which each item is associated with a probability to indicate its existence in the transaction.

While many approaches and techniques have been developed for different uncertain data mining tasks, few research has been undertaken in summarization, though it is important for many real world applications.

A motivating example comes from the equipment rental industry. An equipment rental company usually has multiple branches at different locations, which can be represented by their latitudes and longitudes. For example, a branch B_1 may be located at ($33^\circ48'17''$S, $150^\circ48'14''$E). In order to support business operations (e.g., designing sales promotion strategies for branches), potential equipment demands are gathered by different means, such as mail surveys or customer interviews. A potential demand may be recorded as Crane: ($32^\circ49'12''$S, $151^\circ48'47''$E), in which the latitude and longitude represent the location of the demand — in this case a construction site. Simply assuming the equipment will be rented from the nearest branch is not reasonable, especially if the construction site is almost equally close to multiple branches. The issue can be alleviated by introducing uncertainty. For example, we may record the information as Crane: $\{(B_1{:}0.42), (B_2{:}0.58)\}$, which indicates that cranes may be rented at branches B_1 and B_2 with probabilities 0.42 and 0.58, respectively, where the probabilities can be inferred from the distances between the location of demand and the two branches. Then, the equipment demand information can be organized into an uncertain transaction database, in which each transaction corresponds to a piece of equipment and each item represents a branch. Meanwhile, each item is associated with a probability, indicating how likely the corresponding piece of equipment will be rented from the branch. We may find knowledge by mining tiles from such an uncertain transaction database, such as a set of equipments being rented from a group of branches, which can then be used to design and apply sales promotion strategies and inventory management for a branch group.

Mining tiles from uncertain transaction database can also assist data analysis and management of applications in other domains. For example, in the bioinformatics domain, the relationship between biological entities (e.g., genes, phenotypes and proteins) can be represented by a transaction database, in which each transaction consists of the related entities of an entity. Due to experimental errors and the unreliability of some data sources, some connections may not be meaningful [48]. In order to measure the *goodness*, uncertainty has been introduced to evaluate the strength of a connection from multiple aspects, including the reliability, rarity, and relevance. This associates each connection with a probability and, consequently, an uncertain transaction database can be constructed. Mining tiles from this database will produce a concise set of strongly correlated entities, which may assist further analysis, such as the relationship between genes and phenotypes.

Motivated by its importance and wide applicability, in this chapter, we focus on summarizing uncertain transaction databases and formulate the task as *Minimal Probabilistic Tile Cover* (MPTC) Mining, which aims to find a tile set as a high-quality summarization of an uncertain database. Unlike summarizing a deterministic database, the challenge is how to determine whether a tile set covers an uncertain database or not, because it becomes a probabilistic event. For example, given the uncertain database $\mathcal{D}$ in Table 4.1 (b) and tiles $R_1 = \{t_1, t_2\} \times \{a\}$ and $R_2 = \{t_1\} \times \{b\}$, we cannot tell whether $\mathcal{R} = \{R_1, R_2\}$ covers $\mathcal{D}$, because it depends on the occurrence of item c in transaction t_2, which happens with a probability 0.42. Hence, we define the concept of *cover probability*. Informally, it is the probability that a tile set $\mathcal{R}$ covers the uncertain database $\mathcal{D}$. If the cover probability is greater than a threshold η, $\mathcal{R}$ is called a *probabilistic tile cover* of $\mathcal{D}$.

However, the probabilistic tile cover of a given uncertain database is not unique. There are two special cases among all probabilistic tile covers: (1) the tile set $\mathcal{R}'$ consists of singletons constructed for individual entries of a given uncertain database; (2) the tile set $\mathcal{R}''$ contains only one tile R covering the whole database. Both $\mathcal{R}'$ and $\mathcal{R}''$ are not good summarizations in most circumstances, because the former is not concise while the latter is too noisy. Therefore, we define the *cost* and the *density* of a tile set to measure its size and purity degree respectively. Our goal of MPTC mining is then to find the probabilistic tile cover with the minimal cost, satisfying the condition that the density of the cover is sufficiently high.

The huge search space of possible tiles poses a critical challenge to MPTC mining. It is infeasible to enumerate all possible tiles, because the number of possible tiles is exponential. Observing that the problem of MPTC mining is related to the set cover problem [39], which is known to be NP-hard, we propose the concept of *Probabilistic Price Order* to compare the qualities of tiles, and develop a framework to mine the MPTC with a greedy strategy.

Our framework consists of two steps. We first generate a set of candidates. Then, we iteratively construct tiles based on the candidate set with a greedy strategy and insert the result tile into the result set. The bottleneck is to check whether a tile is a better representative than another according to the *Probabilistic Price Order*, which involves the computation of a joint probability. We prove that the joint probability can be decomposed into independent Poisson binomial distributions and design an efficient algorithm to calculate it. We also devise optimization techniques to further improve the performance.

Our main contributions are summarized as follows.

- We formalize the problem of Minimal Probabilistic Tile Cover Mining and develop a framework to discover the minimal probabilistic tile cover with a greedy strategy.
- We present the concept of Probabilistic Price Order and design an efficient algorithm to determine it by decomposing a joint probability into independent terms, then propose several optimization techniques to further improve the efficiency of the algorithm.
- We conduct extensive experiments to evaluate the performance of the proposed MPTC mining framework and optimization techniques on both synthetic and real world data.

The rest of this chapter is organized as follows. We state some definitions and formulate the problem of Minimal Probabilistic Tile Cover Mining in Section 4.2. Then, a framework to discover the minimal probabilistic tile cover is presented in Section 4.3. We define the concept of Probabilistic Price Order and devise an algorithm to calculate it in Section 4.4. Several optimization techniques are proposed to improve the efficiency of the algorithm in Section 4.5. Section 4.6 provides a detailed analysis of the proposed algorithm. We conduct extensive experiments to evaluate the performance of the proposed MPTC mining framework and optimization techniques on both synthetic and real world data in Section 4.7. Some related works are discussed in Section 4.8. Section 4.9 concludes the chapter.

4.2 Problem Definition

In this section, we first introduce relevant concepts. Then, the problem of minimal probabilistic tile cover (MPTC) mining is formally defined.

Let $\mathcal{I} = \{e_1, e_2, \ldots, e_m\}$ be a universal set of items and $\mathcal{D} = \{t_1, t_2, \ldots, t_n\}$ be an uncertain database consisting of a set of transactions, in which m and n are the number of items and transactions, respectively. Each transaction $t_j \in \mathcal{D}$ is a subset of $\mathcal{I}$ (i.e., $t_j \subseteq \mathcal{I}$). Each item e_i in transaction t_j is associated with a probability $\Pr(e_i \in t_j) \in [0, 1]$ indicating the existence likelihood of e_i in t_j. Then, the concept of *tile* is defined as follows.

Definition 4.1. (*Tile*) *Given a universal set of items $\mathcal{I}$, and an uncertain database $\mathcal{D}$, a tile R is the Cartesian product of a transaction set $T \subseteq \mathcal{D}$, and an itemset $I \subseteq \mathcal{I}$, i.e., $R = T \times I$.*

For example, given the uncertain database $\mathcal{D}$ in Table 4.1 (b), $R_1 = \{t_1, t_2\} \times \{a, b\}$ and $R_2 = \{t_1\} \times \{b, c\}$ are tiles.

Considering that a transaction database is similar to a binary matrix, we call the pair of a transaction and an item (t_j, e_i) a *cell*. Then, a tile is also a set of cells. For example, $R_1 = \{t_1, t_2\} \times \{a, b\} = \{(t_1, a), (t_2, a), (t_1, b), (t_2, b)\}$ and, similarly, $R_2 = \{(t_1, b), (t_1, c)\}$. The union and intersection operations of two tiles can thus be performed on corresponding sets of cells. For example, $R_1 \cup R_2 = \{(t_1, a), (t_2, a), (t_1, b), (t_2, b), (t_2, c)\}$, and $R_1 \cap R_2 = \{(t_1, b)\}$.

Next, we formulate the *cover relation* between an uncertain database $\mathcal{D}$ and a set of tiles $\mathcal{R}$. If $\mathcal{D}$ is a deterministic database, the relationship $\mathcal{D} \subseteq \mathcal{R}$ is certain, which means all of the cells in $\mathcal{D}$ also fall in $\mathcal{R}$. However, in the context of an uncertain database, $\mathcal{D} \subseteq \mathcal{R}$ becomes a probabilistic event, because the existence of each cell $(t, e) \in \mathcal{D}$ is uncertain. To define the cover relation between $\mathcal{D}$ and $\mathcal{R}$, we first adopt the commonly used *Possible World Semantics* to explain the existence of data in an uncertain database. A possible world is a set of deterministic transactions derived from an uncertain

ID	t_1	t_2	Probability
w_1	$\emptyset$	$\{a\}$	0.0464
w_2	$\emptyset$	$\{a, c\}$	0.0336
w_3	$\{b\}$	$\{a\}$	0.1856
w_4	$\{b\}$	$\{a, c\}$	0.1344
w_5	$\{a\}$	$\{a\}$	0.0696
w_6	$\{a\}$	$\{a, c\}$	0.0504
w_7	$\{a, b\}$	$\{a\}$	0.2784
w_8	$\{a, b\}$	$\{a, c\}$	0.2016

Table 4.2: An example of possible worlds.

database. Each possible world has an existential probability. For example, we can derive eight possible worlds from the database in Table 4.1 (b), which are listed in Table 4.2. The probability that the first possible world w_1 exists is $(1 - 0.6) \times (1 - 0.8) \times 1 \times (1 - 0.42) = 0.0464$.

According to the possible world semantics, we define the concept of *cover probability* to characterize the cover relation between an uncertain database and a tile set.

Definition 4.2. (*Cover Probability*) *Given an uncertain database $\mathcal{D}$ and a tile set $\mathcal{R}$, the cover probability of $\mathcal{D}$ and $\mathcal{R}$, denoted by $\mathrm{P}_{cover}(\mathcal{D}, \mathcal{R})$, is defined as follows:*

$$\mathrm{P}_{cover}(\mathcal{D}, \mathcal{R}) = \Pr(\mathcal{D} \subseteq \mathcal{R}) = \sum_{w \in \mathcal{PW}, \mathcal{D}^{(w)} \subseteq \mathcal{R}} \Pr(w)$$

where $\mathcal{PW}$ is the set of possible worlds derived from $\mathcal{D}$ and $\mathcal{D}^{(w)}$ is the database instance in possible world w.

For example, given the uncertain database $\mathcal{D}$ in Table 4.1 (b), and the tile set $\mathcal{R} = \{R_1, R_2\}$, where $R_1 = \{t_1, t_2\} \times \{a\}$ and $R_2 = \{t_1\} \times \{b\}$, the cover probability $\mathrm{P}_{cover}(\mathcal{D}, \mathcal{R}) = 0.58$, because $\mathcal{R}$ covers $\mathcal{D}$ in possible worlds w_1, w_3, w_5 and w_7.

Note that we can also compute the cover probability by:

$$\mathrm{P}_{cover}(\mathcal{D}, \mathcal{R}) = \prod_{(t,e) \in \mathcal{D} \setminus \mathcal{R}} (1 - \Pr(e \in t)), \tag{4.1}$$

based on the fact that the event $\mathcal{D} \subseteq \mathcal{R}$ happens only if all cells in $\mathcal{D} \setminus \mathcal{R}$ do not exist.

Cover probability provides a measure to evaluate how likely a tile set $\mathcal{R}$ may cover an uncertain database $\mathcal{D}$. We then define a tile set as a *probabilistic tile cover* based on cover probability as follows.

Definition 4.3. (*Probabilistic Tile Cover*) *Given an uncertain database $\mathcal{D}$, a tile set $\mathcal{R}$, and a cover probability threshold η, if $\mathrm{P}_{cover}(\mathcal{D}, \mathcal{R}) > \eta$, then $\mathcal{R}$ is a probabilistic tile cover of $\mathcal{D}$.*

However, as we have discussed in the first section, multiple probabilistic tile covers exist for an uncertain database. For the purpose of summarizing an uncertain database with succinct and high-quality representatives, we define the *cost* and *density* of a tile set to measure the efficacy and quality of tiles respectively.

Definition 4.4. (*Cost*) *Given a tile $R = T \times I$, the cost of R, denoted as $\mathrm{c}(R)$, is defined as $|T| + |I|$. The cost of a tile set $\mathcal{R}$ is $\mathrm{c}(\mathcal{R}) = \sum_{R \in \mathcal{R}} \mathrm{c}(R)$.*

For example, the cost of the tile $R = \{t_1\} \times \{a, b\}$ is $\mathrm{c}(R) = 3$.

Definition 4.5. (*Density*) *Given an uncertain database $\mathcal{D}$ and a tile $R = T \times I$, the density of R is defined as $\mathrm{d}(R; \mathcal{D}) = |R \cap \mathcal{D}| / (|T| \cdot |I|)$. The density of a tile set $\mathcal{R}$ is $\mathrm{d}(\mathcal{R}; \mathcal{D}) = |\mathcal{R} \cap \mathcal{D}| / |\mathcal{R}|$, where $|\mathcal{R} \cap \mathcal{D}| = |\cup_{R \in \mathcal{R}} R \cap \mathcal{D}|$ and $|\mathcal{R}| = |\cup_{R \in \mathcal{R}} R|$.*

$|R \cap \mathcal{D}|$ is the cardinality of the intersection of cells of R and database $\mathcal{D}$. Since the existence of each cell $(t, e) \in R \cap \mathcal{D}$ is a Bernoulli trial, $|R \cap \mathcal{D}|$ is a random variable. Similarly, $|\mathcal{R} \cap \mathcal{D}|$ is also a random variable. Hence, calculating the density is complex.

For example, given the uncertain database $\mathcal{D}$ in Table 4.1 (b), the density of $R = \{t_1\} \times \{a, b\}$ could be 0, 0.5 and 1 (the corresponding values of $|R \cap \mathcal{D}|$ are 0, 1 and 2), with probabilities 0.08, 0.44 and 0.48, respectively. Hereafter, we will use $\mathrm{d}(R)$ and $\mathrm{d}(\mathcal{R})$ when $\mathcal{D}$ is clear from the context.

Then, the problem of *minimal probabilistic tile cover* mining can be stated as follows.

Problem 4.1. (*Minimal Probabilistic Tile Cover Mining*) *Given an uncertain database $\mathcal{D}$, a cover probability threshold η, a density threshold θ and a density probability threshold ξ, the objective of minimal probabilistic tile cover mining is to find a probabilistic tile cover $\mathcal{R}$ of $\mathcal{D}$ with minimal cost* $\mathrm{c}(\mathcal{R})$ *and sufficient density,*

$$\begin{aligned} &\arg\min_{\mathcal{R}} \mathrm{c}(\mathcal{R}) \\ &\text{s.t. } \mathrm{P}_{cover}(\mathcal{D}, \mathcal{R}) > \eta,\ \Pr[\mathrm{d}(\mathcal{R}) > \theta] > \xi \end{aligned} \tag{4.2}$$

Before describing our algorithm for MPTC mining, we discuss some properties of the defined problem.

4.2.1 Quality of Summarization

The quality of MPTC is guaranteed by the constraints of the cover probability and the density. Consider the two types of error, false negative and false positive, which can be represented by $\mathcal{D} \setminus \mathcal{R}$ and $\mathcal{R} \setminus \mathcal{D}$, respectively. The false negative error is constrained by the cover probability condition: $\Pr(\mathcal{D} \subseteq \mathcal{R}) > \eta$. Because $\mathcal{D} \subseteq \mathcal{R} \Leftrightarrow \mathcal{D} \setminus \mathcal{R} = \emptyset$, this constrain ensures that the probability of generating no false negative is greater than η.

According to the constraint $\Pr[\mathrm{d}(\mathcal{R}) > \theta] > \xi$, we have

$$\Pr[\mathrm{d}(\mathcal{R}) > \theta] = \Pr\left[\frac{|\mathcal{R} \cap \mathcal{D}|}{|\mathcal{R}|} > \theta\right] = \Pr\left[\frac{|\mathcal{R} \setminus \mathcal{D}|}{|\mathcal{R}|} < 1 - \theta\right],$$

where $|\mathcal{R} \setminus \mathcal{D}|/|\mathcal{R}|$ is called the *false discovery rate*. Hence, the constraint of density guarantees that the probability that the false discovery rate is less than $1 - \theta$ is greater than ξ.

Hence, MPTC can summarize an uncertain database with both guaranteed false negative and false positive errors.

4.2.2 Parameter Setting

The problem of MPTC mining involves three parameters, η, θ and ξ. We discuss the semantics and connections between the parameters, which serve as a guide to choose appropriate parameter values.

The cover probability threshold η can be set first. Recall that the cover probability $\mathrm{P}_{cover}(\mathcal{D}, \mathcal{R})$ can be alternatively computed as $\prod_{(t,e)\in\mathcal{D}\setminus\mathcal{R}}(1 - \Pr(e \in t))$, which is the production of the non-occurrence probabilities of cells that are not covered by $\mathcal{R}$. Therefore, instead of directly setting η, users may set two parameters u and v, where u is the percentage of cells that are allowed to be uncovered, and v is the average existential probability of each uncovered cell. Then, $\eta = (1-v)^{u\%\times|\mathcal{D}|}$. For example, given a database of 1000 cells, if a user wants to find a MPTC such that at most 2% of cells are uncovered and the average existential probability of each cell is 0.1, the user can set $u = 2$ and $v = 0.1$, which is equivalent to setting $\eta = (1-0.1)^{2\%\times1000} \approx 0.122$.

Once the cover probability η has been set, given an uncertain database $\mathcal{D}$, the density of a tile set $\mathcal{R}$ satisfying the condition $\mathrm{P}_{cover}(\mathcal{D}, \mathcal{R}) > \eta$ is upper bounded.

Given an uncertain database $\mathcal{D}$ and a cover probability threshold η, suppose $\{X_1, \ldots, X_r\}$ are the Bernoulli random variables indicating the existence of all cells $\{c_1, \ldots, c_r\}$ in $\mathcal{D}$. Without loss of generality, we assume $\Pr(X_i = 1) \geq \Pr(X_j = 1)$ for any $1 \leq i < j \leq r$. Then, the upper bound of density d^* can be obtained as follows. We iteratively add the singleton tile constructed from the cell c in $\mathcal{D}$ with the largest existential probability to a tile set $\mathcal{R}$, until the cover probability of $\mathcal{R}$ is larger than η. The result is the tile set with maximal density given the cover probability constraint, because adding any other tiles will decrease the density, and removing any tile from it will violate the cover probability constraint. We denote the expected value of the density of $\mathcal{R}$ as the upper bound d^*. Then, users may set the density threshold θ to be $\lambda \cdot d^*$, $\lambda \in (0, 1)$, expressing to how much degree the density threshold is smaller than d^*. ξ is easier to set since it is a probability threshold in $[0, 1]$.

4.2.3 NP-Hardness

Since its counterpart in deterministic databases is NP-hard [44], compared with the problem of MPTC mining, mining tiles from deterministic database can be regarded as a special case where all existential probabilities of cells are either 0 or 1. Therefore, the problem of MPTC mining is also NP-Hard.

4.3 Algorithm

In this section, we present some preliminaries and then propose the framework for MPTC mining.

4.3.1 Preliminaries

According to the definition of Problem 4.1, a tile is more interesting if it covers more cells of the database with lower cost (i.e., smaller size). Therefore, we first define the concept of *Probabilistic Cover Quantity* to count the number of cells that can be covered by a tile. Then, we define *Probabilistic Price* to measure the covering efficiency of a tile.

Definition 4.6. (*Probabilistic Cover Quantity*) *Given an uncertain database $\mathcal{D}$, a tile R, and a cell set $Z \subset \mathcal{D}$ that has been covered by previously mined tiles, the probabilistic cover quantity of R given Z and $\mathcal{D}$, denoted by $\mathrm{q}(R; Z, \mathcal{D})$, is defined as $\mathrm{q}(R; Z, \mathcal{D}) = |(R \setminus Z) \cap \mathcal{D}|$ and the probability mass function of $\mathrm{q}(R; Z, \mathcal{D})$ is $\Pr[\mathrm{q}(R; Z, \mathcal{D}) = k] = \sum_{w \in W_k} \Pr(w)$ where $W_k = \{w \in \mathcal{PW} | |(R \setminus Z) \cap \mathcal{D}^{(w)}| = k\}$. Similarly, the probabilistic cover quantity of a tile set $\mathcal{R}$ given Z and $\mathcal{D}$ is $\mathrm{q}(\mathcal{R}; Z, \mathcal{D}) = |(\cup_{R \in \mathcal{R}} R \setminus Z) \cap \mathcal{D}|$.*

The probabilistic cover quantity $\mathrm{q}(R; Z, \mathcal{D})$ is a random variable. It describes the number of uncovered cells in $\mathcal{D}$ that are covered by R. We use $\mathrm{q}(R; Z)$ and $\mathrm{q}(\mathcal{R}; Z)$ for brevity when $\mathcal{D}$ is clear from the context.

For example, given the uncertain database $\mathcal{D}$ in Table 4.1 (b), let tile $R_1 = \{t_1, t_2\} \times \{a, b\}$, and the covered cell set $Z = \{(t_1, a), (t_2, a)\}$, the probabilistic cover quantity of R_1 given Z is $\mathrm{q}(R_1; Z) = |(R_1 \setminus Z) \cap \mathcal{D}| = |\{(t_1, b)\}| = 1$, which is a random variable such that $\Pr[\mathrm{q}(R_1; Z) = 1] = \Pr(b \in t_1)$.

Definition 4.7. (*Probabilistic Price*) *Given an uncertain database $\mathcal{D}$, a covered cell set $Z \subset \mathcal{D}$, and a tile R, the probabilistic price of R is defined as $\rho(R; Z) = \mathrm{c}(R)/\mathrm{q}(R; Z)$. Similarly, the probabilistic price of a tile set $\mathcal{R}$ is $\rho(\mathcal{R}; Z) = \mathrm{c}(\mathcal{R})/\mathrm{q}(\mathcal{R}; Z)$.*

That is, the probabilistic price of a tile is the cost of the tile divided by its probabilistic cover quantity. Hence, a tile R covers an uncertain database more efficiently when the probabilistic price tends to be lower. However, because of $\mathrm{q}(R; Z)$, the probabilistic price is also a random variable. As a result, we cannot directly compare two tiles by their probabilistic prices. One possible solution is to use their expected values. For example, given two tiles R_1 and R_2, we prefer R_1 if $\mathrm{E}[\rho(R_1; Z)] < \mathrm{E}[\rho(R_2; Z)]$. However, recent works have shown that considering only the expected value has some drawbacks such as the decision may not be confident because it neglects the distribution information [4]. Therefore, in our work, we define the *Probabilistic Price Order* to compare two tiles as follows.

Definition 4.8. (*Probabilistic Price Order*) *Given an uncertain database $\mathcal{D}$, a covered cell set Z, and two tiles R_1 and R_2, we say R_1 precedes R_2 in terms of the Probabilistic Price Order, denoted by $R_1 \prec_\rho R_2$, if and only if*

$$\Pr\left[\rho(R_1; Z) < \rho(R_2; Z)\right] > \Pr\left[\rho(R_1; Z) > \rho(R_2; Z)\right]$$

Similarly, given two tile sets $\mathcal{R}_1$ and $\mathcal{R}_2$, $\mathcal{R}_1 \prec_\rho \mathcal{R}_2$ if and only if,

$$\Pr\left[\rho(\mathcal{R}_1; Z) < \rho(\mathcal{R}_2; Z)\right] > \Pr\left[\rho(\mathcal{R}_1; Z) > \rho(\mathcal{R}_2; Z)\right]$$

For example, consider the database $\mathcal{D}$ in Table 4.1 (b) again. Suppose tiles $R_1 = \{t_1\} \times \{a, b\}$, $R_2 = \{t_2\} \times \{a, c\}$ and covered cell set $Z = \emptyset$. Then, the probability of $\rho(R_1; Z) < \rho(R_2; Z)$ is 0.256, while the probability of $\rho(R_1; Z) > \rho(R_2; Z)$ is 0.288 (the detailed calculation of the probabilities will be explained in Section 4.4). Thus, according to probabilistic price order, R_2 is preferred to R_1. However, we notice the opposite result if using the expected values, i.e., $\mathrm{E}[\rho(R_1; Z)] = 1.4 < \mathrm{E}[\rho(R_2; Z)] = 1.42$.

4.3.2 MPTC Mining Framework

We now present the framework for solving the MPTC mining problem with a greedy strategy based on the probabilistic price order.

We observe that the MPTC mining problem is related to the set cover problem [39]. Given a universal set of elements $\mathcal{I}$ and a family of sets $\mathcal{S}$,

Algorithm 4.1 MPTC Mining

```
Input: $\mathcal{D}$, $minsup$, $minprob$, $\eta$, $\theta$, $\xi$
Output: MPTC $\mathcal{R}^*$
 1: $\mathcal{R}^* \leftarrow \emptyset$
 2: Candidate set $\mathcal{C} \leftarrow generate\_candidates(\mathcal{D})$
 3: while $\mathrm{P}_{cover}(\mathcal{D}, \mathcal{R}^*) < \eta$ do
 4:   $R^* \leftarrow \emptyset$
 5:   for $I$ in $\mathcal{C}$ do
 6:     $R \leftarrow get\_tile(I, \mathcal{T}(I), \theta, \xi)$
 7:     if $\mathcal{R}^* \cup \{R\} \prec_\rho \mathcal{R}^* \cup \{R^*\}$ then
 8:       $R^* \leftarrow R$
 9:   $\mathcal{R}^* \leftarrow \mathcal{R}^* \cup \{R^*\}$
10: return $\mathcal{R}^*$
```

the Set Cover Problem is to find a minimal subset $\mathcal{S}^*$ of $\mathcal{S}$ that completely covers the elements of $\mathcal{I}$. The set cover problem is known to be NP-hard. An intuitive and effective solution is the greedy approach [39], which iteratively chooses the set covering most uncovered items, until all items are covered.

Motivated by the greedy solution of the set cover problem, we propose a two-step framework to solve the MPTC mining problem:

Step 1 generates a candidate set for producing tiles;

Step 2 constructs tiles based on the candidates and keep the best one iteratively, until reaching the threshold of cover probability.

The complete framework is shown in Algorithm 4.1. Line 1 initializes the result set $\mathcal{R}^*$. We generate the candidate set $\mathcal{C}$ in line 2. The choice of candidates will be discussed in Section 4.3.3. Lines $3-9$ construct tiles based on the candidate set $\mathcal{C}$ and select the best one iteratively. In particular, line 3 examines whether the cover probability of current $\mathcal{R}^*$ is greater than the user-specified threshold η. In lines $5-8$, we find the tile for each candidate I in $\mathcal{C}$ by invoking the function *get_tile*,and keep the smallest one in terms of the probabilistic price order. We explain the details of the function *get_tile* in Section 4.3.4, and discuss how to determine the probabilistic price order of two tiles in Section 4.4. The result tile is inserted into the result set in line 9.

4.3.3 Generating Candidates

Since enumerating an exponential number of possible tiles is infeasible, our approach first generates a set of candidates to reduce the search space. It is clear that choosing a good set of candidates is very important. Generally, a good candidate set should satisfy two criteria: (1) it should lead to a high-quality probabilistic tile cover; (2) it should be as small as possible for the sake of efficiency.

Similar to summarizing deterministic data [44], a reasonable choice of candidates for mining MPTC from uncertain databases is the probabilistic frequent pattern set $\mathcal{F}$. Given an uncertain database $\mathcal{D}$, an itemset I is a probabilistic frequent pattern if the probability that the support of I is no less than $minsup$ is greater than $minprob$ (i.e., $\Pr(\text{Supp}(I) \geq minsup) > minprob$), where the support of I is the number of occurrence of I in $\mathcal{D}$. A probabilistic frequent pattern captures the items that are likely to appear in the same set of transactions, which is suitable for the construction of tiles. If a tile $R = T \times I$ such that $I \in \mathcal{F}$, then it guarantees that $\Pr(|T| > minsup) \geq minprob$, which indicates that the tile R will not be too small.

Nevertheless, the probabilistic frequent pattern set may be large and redundant since it enjoys the anti-monotonic property [4]. To alleviate this issue, we use the summarized set − *probabilistic representative frequent patterns* (P-RFP) [40] − as candidates, which is more concise and less redundant. Interested readers may refer to [40] for details. We also insert all individual items (i.e., members of $\mathcal{I}$) into the candidate set in order to ensure that the produced tile set is possible to completely cover the uncertain database.

After the generation process, the candidate set is checked in Line 5 of Algorithm 4.1. There are several choices of orders to iterate it, such as alphabetical order, descending frequency order or descending length order. Considering that longer patterns tend to generate larger tiles, we prefer to check the longer pattern first for the conciseness of the result. Note that given a current result probabilistic tile set R^* and two probabilistic tiles R_1 and R_2, which are constructed from candidates I_1 and I_2, respectively, the order of iterating affects the result only when

$$\begin{aligned} &\Pr\left[\rho(R^* \cup \{R_1\}; Z) < \rho(R^* \cup \{R_2\}; Z)\right] \\ =&\Pr\left[\rho(R^* \cup \{R_1\}; Z) > \rho(R^* \cup \{R_2\}; Z)\right]. \end{aligned} \tag{4.3}$$

Otherwise, the one ranked higher with respect to the probabilistic price order would be preferred no matter whether it occurred earlier or later.

4.3.4 Constructing Tiles

Given a candidate I, the function *get_tile* constructs a tile $R^* = T^* \times I$, where $T^* \subseteq \mathcal{T}(I)$ and $\mathcal{T}(I)$ is the set of supporting transactions of I in the uncertain database $\mathcal{D}$.

The basic idea is to enumerate all possible tiles $R = T \times I$, such that $T \subseteq \mathcal{T}(I)$. Although a high-quality tile can be generated, the process is time-consuming because the number of possible tiles is exponential (i.e. $2^{|\mathcal{T}(I)|}$). Hence, we design a greedy strategy to iteratively add a transaction into the result set, and finally return a tile covering more cells with lower cost.

Algorithm 4.2 illustrates the function *get_tile*. Line 1 initializes the result tile R^*. Line 2 constructs a set of tiles $\mathcal{R}(I)$ from I and $\mathcal{T}(I)$, where each tile R in $\mathcal{R}(I)$ is $\{t\} \times I$, such that $t \in \mathcal{T}(I)$. Then, we sort $\mathcal{R}(I)$ in the descending probabilistic price order (line 3). Lines $4 - 9$ search for the result tile greedily. The algorithm checks each tile R constructed from sorted $\mathcal{R}(I)$ (line 4) and merge R into R^* only if the updated tile R' precedes the current tile R^* and the density of the resulted tile set $\mathcal{R} \cup \{R'\}$ satisfies the constraint $\Pr(\mathrm{d}(\mathcal{R} \cup \{R'\}) > \theta) > \xi$ (lines $6 - 7$). Otherwise, the algorithm stops and return R^* as the result (lines $8 - 10$).

According to the definition of density, we have $\mathrm{d}(\mathcal{R}) = |\mathcal{R} \cap \mathcal{D}|/|\mathcal{R}|$. The denominator can be easily computed given a tile set. Calculating the numerator is more complex because it is a random variable. Let $\mathbf{X} = |\mathcal{R} \cap \mathcal{D}| = \sum_i X_i$, where each X_i is a random variable indicating the event $e \in t$ for cell $c_i = (t, e) \in \mathcal{R} \cap \mathcal{D}$ (i.e., $\Pr(X_i = 1) = \Pr(e \in t)$). We denote $\Pr(X_i = 1)$ as p_i. Since all X_is are independent Bernoulli trials, $\mathbf{X}$ follows a Poisson binomial distribution, whose expected value and variance are $\mu = \sum_i p_i$

Algorithm 4.2 Function *get_tile*

```
Input: Itemset $I$, Supporting Transaction Set of $I$, $\mathcal{T}(I)$, density thresholds
    $\theta$ and $\xi$
Output: Probabilistic Tile $R^*$
 1: $R^* \leftarrow \emptyset$
 2: $\mathcal{R}(I) \leftarrow \{\{t\} \times I | t \in \mathcal{T}(I)\}$
 3: Sort $\mathcal{R}(I)$ in the descending probabilistic price order
 4: for $R$ in $\mathcal{R}(I)$ do
 5:     $R' \leftarrow R^* \cup R$
 6:     if $R' \prec_\rho R^*$ and $\Pr[\mathrm{d}(\mathcal{R} \cup \{R'\}) > \theta] > \xi$ then
 7:         $R^* \leftarrow R'$
 8:     else
 9:         break
10: return $R^*$
```

and $\sigma^2 = \sum_i p_i(1-p_i)$, respectively. According to [8], the density $\mathrm{d}(\mathcal{R})$ can be well approximated by a normal distribution:

$$\Pr[\mathrm{d}(\mathcal{R}) > \theta] \approx 1 - \Phi\left[(\theta \cdot |\mathcal{R}| - 0.5 - \mu)/\sigma\right], \tag{4.4}$$

where Φ is the cumulative distribution function of the standard normal distribution. According to the Berry-Esseen Theorem [49], the upper bound of the approximation error is $C \cdot \psi$, in which $\psi = \sum_{i=1}^{n} |p_i(1-p_i)(1-2p_i)| / \sigma^3$ and $C < 0.7915$.

The bottleneck of *get_tile* function is then how to determine whether $R' \prec_\rho R^*$, which involves the calculation of probability $\Pr\left[\rho(R'; Z) < \rho(R^*; Z)\right]$. We discuss the details in the next section.

4.4 Probabilistic Price Order

In this section, we discuss how to efficiently determine the probabilistic price order of two tiles. That is, to calculate the probability $\Pr\left[\rho(R_1; Z) < \rho(R_2; Z)\right]$, where R_1 and R_2 are two tiles, and Z is the set of covered cells.

Let q_1 and q_2 be the maximal values of probabilistic cover quantities of $\mathrm{q}(R_1; Z)$ and $\mathrm{q}(R_2; Z)$, respectively. According to the definition of probabilistic price, we have the following,

$$\Pr\left[\rho(R_1;Z) < \rho(R_2;Z)\right] = \sum_{k=0}^{q_2} \sum_{l=\lceil \mathrm{c}(R_1)\cdot k/\mathrm{c}(R_2)\rceil}^{q_1} \Pr\left[\mathrm{q}(R_1;Z)=l, \mathrm{q}(R_2;Z)=k\right] \tag{4.5}$$

Hence, to determine the probabilistic price order, we need to calculate the joint probability $\Pr\left[\mathrm{q}(R_1;Z)=l, \mathrm{q}(R_2;Z)=k\right]$.

We observe that the probabilistic cover quantity $\mathrm{q}(H;Z)$ of a tile R is a random variable following a Poisson binomial distribution, which is stated in Theorem 4.1.

Theorem 4.1. *Given an uncertain database $\mathcal{D}$, a tile R and a covered cell set Z, the probabilistic cover quantity of R given Z, $\mathrm{q}(R;Z)$, follows a Poisson binomial distribution, with the following expected value and variance.*

$$\begin{aligned}\mathrm{E}\left[\mathrm{q}(R;Z)\right] &= \sum_{(t,e)\in(R\setminus Z)\cap\mathcal{D}} \Pr(e\in t)\\ \mathrm{Var}\left[\mathrm{q}(R;Z)\right] &= \sum_{(t,e)\in(R\setminus Z)\cap\mathcal{D}} \Pr(e\in t)\cdot(1-\Pr(e\in t))\end{aligned}$$

To prove Theorem 4.1, we represent the existence of each cell in $(R\setminus Z)\cap\mathcal{D}$ by a binary random variable, then $\mathrm{q}(R;Z)$ becomes the sum of Bernoulli trials, which follows a Poisson binomial distribution. The expected value and variance can be derived accordingly.

Proof. Let $\mathcal{X}_{R\cap\mathcal{D}}$ and $\mathcal{X}_Z$ be sets of Bernoulli random variables indicating the existence of cells in $R\cap\mathcal{D}$ and Z, respectively. Then, we order the random variables as follows:

$$\begin{aligned}\mathcal{X}_{R\cap\mathcal{D}} &= \{X_1, X_2, \ldots, X_\alpha, X_{\alpha+1}, \ldots, X_\beta\}\\ \mathcal{X}_Z &= \{X_{\alpha+1}, X_{\alpha+2}\ldots, X_\gamma\}, 1\le\alpha\le\beta\le\gamma\end{aligned}$$

Hence, $\mathcal{X}_{(R\setminus Z)\cap\mathcal{D}} = \{X_1, X_2, \ldots, X_\alpha\}$.

Since $\mathrm{q}(R;Z) = |(R\setminus Z)\cap\mathcal{D}|$, we have $\mathrm{q}(R;Z) = \sum_{i=1}^{\alpha} X_i$. Therefore, $\mathrm{q}(R;Z)$ is the sum of independent Bernoulli trials, which follows a Poisson binomial distribution. Then, the expected value and variance of $\mathrm{q}(R;Z)$ can be deduced according to the property of Poisson binomial distribution.

$$\begin{aligned}
\mathrm{E}\left[\mathrm{q}(R;Z)\right] &= \sum_{i=1}^{\alpha} \Pr(X_i = 1) = \sum_{(t,e)\in R\setminus Z} \Pr(e \in t) \\
\mathrm{Var}\left[\mathrm{q}(R;Z)\right] &= \sum_{i=1}^{\alpha} \Pr(X_i = 1) \cdot (1 - \Pr(X_i = 1)) \\
&= \sum_{(t,e)\in R\setminus Z} \Pr(e \in t) \cdot (1 - \Pr(e \in t))
\end{aligned}$$

which proves the theorem. □

Theorem 4.1 indicates that $\Pr\left[\mathrm{q}(R_1;Z) = k, \mathrm{q}(R_2;Z) = l\right]$ is a joint probability of two Poisson binomial distributions. Both an exact method [40] and an approximate approach [50] have been proposed for computing the joint probability. However, the exact method is not adequately efficient, while the approximate solution requires the number of Bernoulli trials to be large enough. Since the joint probability is not arbitrary but produced from the cover quantities of two probabilistic tiles, we propose a technique to decompose it into three independent distributions, which enhances the efficiency without loss of accuracy. The details are stated in Theorem 4.2.

Theorem 4.2. *Given two tiles R_1 and R_2, a covered cell set Z, and two integers k and l, $0 \le k \le q_1$ and $0 \le l \le q_2$, where q_1 and q_2 are the maximum value of $\mathrm{q}(R_1;Z)$ and $\mathrm{q}(R_2;Z)$, the joint probability of $\mathrm{q}(R_1;Z)$ and $\mathrm{q}(R_2;Z)$ can be calculated as follows.*

$$\begin{aligned}
&\Pr\left[\mathrm{q}(R_1;Z) = k, \mathrm{q}(R_2;Z) = l\right] \\
=&\sum_{i=0}^{q_0} \Pr\left[\mathrm{q}(R_1 \cap R_2;Z) = i\right] \cdot \Pr\left[\mathrm{q}(R_1 \setminus R_2;Z) = k - i\right] \cdot \\
&\Pr\left[\mathrm{q}(R_2 \setminus R_1;Z) = l - i\right] \qquad (4.6)
\end{aligned}$$

where q_0 is the maximum value of $\mathrm{q}(R_1 \cap R_2;Z)$.

Theorem 4.2 can be proved by separating the cells in R_1 and R_2 into disjoint parts: $R_1 \setminus R_2$, $R_2 \setminus R_1$ and $R_1 \cap R_2$, whose probabilistic cover quantities are mutually independent.

Proof. Let $\mathcal{X}_1$ and $\mathcal{X}_2$ be sets of Bernoulli random variables indicating the existence of cells in $R_1 \setminus Z$ and $R_2 \setminus Z$, respectively. Similar to the proof of Theorem 4.1, we order the random variables as follows:

$$\mathcal{X}_1 = \{X_1, X_2, \ldots, X_\beta\}$$
$$\mathcal{X}_2 = \{X_{\alpha+1}, X_{\alpha+2} \ldots, X_q\}, 1 \leq \alpha \leq \beta \leq q$$

Suppose $\mathbf{X}_1$ and $\mathbf{X}_2$ are two random variables, such that $\mathbf{X}_1 = \sum_{i=1}^{\beta} X_i$ and $\mathbf{X}_2 = \sum_{i=\alpha+1}^{q} X_i$. Then, both $\mathbf{X}_1$ and $\mathbf{X}_2$ follow Poisson binomial distribution. The joint probability can be expressed using $\mathbf{X}_1$ and $\mathbf{X}_2$ as follows.

$$\Pr\left[\mathrm{q}(R_1; Z) = k, \mathrm{q}(R_2; Z) = l\right] = \Pr\left[\mathbf{X}_1 = k, \mathbf{X}_2 = l\right]$$

Let

$$\mathbf{Y}_1 = \sum_{i=1}^{\alpha} X_i, \mathbf{Y}_2 = \sum_{i=\alpha+1}^{\beta} X_i, \mathbf{Y}_3 = \sum_{i=\beta+1}^{q} X_i$$

That is, $\mathbf{Y}_1 = \mathrm{q}(R_1 \setminus R_2; Z)$, $\mathbf{Y}_2 = \mathrm{q}(R_1 \cap R_2; Z)$ and $\mathbf{Y}_3 = \mathrm{q}(R_2 \setminus R_1; Z)$. Since $\mathbf{X}_1 = \mathbf{Y}_1 + \mathbf{Y}_2$ and $\mathbf{X}_2 = \mathbf{Y}_2 + \mathbf{Y}_3$, the joint probability can be computed as follows.

$$\begin{aligned}
&\Pr(\mathbf{X}_1 = k, \mathbf{X}_2 = l) \\
=&\Pr(\mathbf{Y}_1 + \mathbf{Y}_2 = k, \mathbf{Y}_2 + \mathbf{Y}_3 = l) \\
=&\sum_{y_2=0}^{\beta-\alpha} \Pr(\mathbf{Y}_2 = y_2) \cdot \Pr(\mathbf{Y}_1 = k - y_2, \mathbf{Y}_3 = l - y_2 | \mathbf{Y}_2 = y_2)
\end{aligned}$$

Consider that the transactions in $R_1 \setminus R_2$, $R_1 \cap R_2$ and $R_2 \setminus R_1$ do not have intersections, $\mathbf{Y}_1$, $\mathbf{Y}_2$ and $\mathbf{Y}_3$ are mutually independent. Hence,

$$\begin{aligned}
&\Pr(\mathbf{Y}_1 + \mathbf{Y}_2 = k, \mathbf{Y}_2 + \mathbf{Y}_3 = l) \\
=&\sum_{y_2=0}^{\beta-\alpha} \Pr(\mathbf{Y}_2 = y_2) \cdot \Pr(\mathbf{Y}_1 = k - y_2) \cdot \Pr(\mathbf{Y}_3 = l - y_2) \\
=&\sum_{i=0}^{q_0} \Pr\left[\mathrm{q}(R_1 \cap R_2; Z) = i\right] \cdot \Pr\left[\mathrm{q}(R_1 \setminus R_2; Z) = k - i\right] \cdot \Pr\left[\mathrm{q}(R_2 \setminus R_1; Z) = l - i\right]
\end{aligned}$$

which proves the theorem. □

Until now, we have a general solution for determining the probabilistic price order, which can be applied in both Algorithms 4.1 (e.g., line 7) and 4.2 (e.g., line 6).

4.5 Optimization Techniques

In this section, we introduce three optimization techniques to accelerate the computation for determining the probabilistic price order.

4.5.1 Optimizing Single Transaction Difference

Note that, in line 6 of Algorithm 4.2, the two tiles, R' and R^*, differ by one transaction only. With this condition, we can determine the probabilistic price order more efficiently.

Theorem 4.3. *Given two tiles $R_1 = T_1 \times I$, $R_2 = T_2 \times I$, and $\Delta R = \{t\} \times I$, such that $R_1 = R_2 \cup \Delta R$, then*

$$\Pr\left[\rho(R_1;Z) < \rho(R_2;Z)\right] = \sum_{i=0}^{\Delta q} \sum_{j=0}^{\mathrm{c}(R_2)\cdot i} \Pr[\mathrm{q}(\Delta R;Z) = i] \cdot \Pr[\mathrm{q}(R_2;Z) = j] \tag{4.7}$$

where Δq is the maximum values of $\mathrm{q}(\Delta R;Z)$, and Z is the set of covered cells.

Theorem 4.3 can be proved based on the definition of probabilistic price and the idea that $\mathrm{c}(R_1) = |T_2 \cup \{t\}| + |I| = \mathrm{c}(R_2) + 1$ and $\mathrm{q}(R_1;Z) = \mathrm{q}(R_2;Z) + \mathrm{q}(\Delta R;Z)$.

Proof. According to the definition of probabilistic price,

$$\Pr\left[\rho(R_2;Z) < \rho(R_1;Z)\right] = \Pr\left[\frac{\mathrm{c}(R_2)}{\mathrm{q}(R_2;Z)} < \frac{\mathrm{c}(R_1)}{\mathrm{q}(R_1;Z)}\right]$$

Since $R_2 = R_1 \cup \Delta R$, we have $\mathrm{c}(R_2) = \mathrm{c}(R_1) + 1$ and $\mathrm{q}(R_2; Z) = \mathrm{q}(R_1; Z) + \mathrm{q}(\Delta R; Z)$. Hence,

$$\begin{aligned}
&\Pr\left[\frac{\mathrm{c}(R_2)}{\mathrm{q}(R_2;Z)} < \frac{\mathrm{c}(R_1)}{\mathrm{q}(R_1;Z)}\right] \\
=&\Pr\left[\frac{\mathrm{c}(R_1)+1}{\mathrm{q}(R_1;Z)+\mathrm{q}(\Delta R;Z)} < \frac{\mathrm{c}(R_1)}{\mathrm{q}(R_1;Z)}\right] \\
=&\Pr\left[\mathrm{q}(R_1;Z) < \mathrm{c}(R_1)\mathrm{q}(\Delta R;Z)\right] \\
=&\sum_{i=0}^{\Delta q}\sum_{j=0}^{\mathrm{c}(R_1)\cdot i} \Pr[\mathrm{q}(\Delta R;Z)=i, \mathrm{q}(R_1;Z)=j]
\end{aligned}$$

Consider that $\mathrm{q}(\Delta R; Z)$ and $\mathrm{q}(R_1; Z)$ are independent, we have

$$\begin{aligned}
&\Pr\left[\frac{\mathrm{c}(R_2)}{\mathrm{q}(R_2;Z)} < \frac{\mathrm{c}(R_1)}{\mathrm{q}(R_1;Z)}\right] \\
=&\sum_{i=0}^{\Delta q}\sum_{j=0}^{\mathrm{c}(R_1)\cdot i} \Pr[\mathrm{q}(\Delta R;Z)=i]\cdot\Pr[\mathrm{q}(R_1;Z)=j]
\end{aligned}$$

which proves the theorem. □

4.5.2 Adaptively Computing Cover Quantity

Computing the probability $\Pr[\mathrm{q}(R;Z) = k]$ is essential for determining the probabilistic price order. As discussed in Theorem 4.1, $\mathrm{q}(R;Z)$ follows a Poisson binomial distribution. The probability mass function of Poisson binomial distribution can be calculated by exact approaches [4, 5] or approximate methods [8, 28].

Again, both types of approach have certain limitations. Suppose $\mathrm{q}(R;Z)$ is the sum of N Bernoulli trials. If we use the exact approach, then the algorithm will be very inefficient when N is large. On the other hand, the approximate approach may produce a noticeable error when N is small. Hence, we combine both the exact and approximate approach into an adaptive method, which chooses the solution automatically depends on N.

Empirical results show that the error of approximation is less than 0.03 when the number of Bernoulli trials is more than 100. Thus, we use the approximate approach only if N is no less than 100. Otherwise, an exact algorithm is employed. The details of the exact algorithm and the approximate approach used are described in [4, 8].

4.5.3 Pruning by 3σ Property

Since $\mathrm{q}(R_1; Z)$ and $\mathrm{q}(R_2; Z)$ follow Poisson binomial distributions, they can be well-approximated by normal distributions when the maximum values of $\mathrm{q}(R_1; Z)$ and $\mathrm{q}(R_2; Z)$, denoted by q_1 and q_2, are large enough. If they can be well-approximated by normal distributions, we can employ the 3σ property of normal distribution [41] and further improve the performance of determining the probabilistic price order. Suppose the expected value and standard deviation of $\mathrm{q}(R_i; Z)$ are μ_i and σ_i, respectively. Denote $\mathrm{q}(R; Z)$ as $\mathrm{q}(R)$ and $\rho(R; Z)$ as $\rho(R)$ for brevity. The joint probability can be calculated by

$$\Pr\left[\rho(R_1) < \rho(R_2)\right] = \sum_{k=k_1}^{k_2} \sum_{l=l_1}^{l_2} \Pr\left[\mathrm{q}(R_1) = l, \mathrm{q}(R_2) = k\right]$$

where $k_1 = \max\{0, \mu_2 - 3\sigma_2\}$, $k_2 = \min\{q_2, \mu_2 + 3\sigma_2\}$, $l_1 = \max\{\left\lceil \frac{\mathrm{c}(R_1)\cdot k}{\mathrm{c}(R_2)} \right\rceil, \mu_1 - 3\sigma_1\}$, and $l_2 = \min\{q_1, \mu_1 + 3\sigma_1\}$.

Given a random variable $\mathbf{X} \sim \mathbf{N}(\mu, \sigma)$, we have $\Pr(\mu - 3\sigma \leq X \leq \mu + 3\sigma) \approx 99.7\%$. Hence, if l or k is out of the range $[\mu_1 - 3\sigma_1, \mu_1 + 3\sigma_1]$ or $[\mu_2 - 3\sigma_2, \mu_2 + 3\sigma_2]$, respectively, we can safely skip the calculation. This property can also be similarly applied to Theorem 4.3 for the situation when two tiles differ by one transaction by

$$\Pr\left[\rho(R_1) < \rho(R_2)\right] = \sum_{i=i_1}^{i_2} \sum_{j=j_1}^{j_2} \Pr[\mathrm{q}(\Delta R) = i] \cdot \Pr[\mathrm{q}(R_2) = j]$$

where $i_1 = \max\{0, \Delta\mu - 3\Delta\sigma\}$, $i_2 = \min\{\Delta q, \Delta\mu + 3\Delta\sigma\}$, $j_1 = \max\{0, \mu_2 - 3\sigma_2\}$, and $j_2 = \min\{\mathrm{c}(R_2) \cdot i, \mu_2 + 3\sigma_2\}$.

4.6 Algorithm Analysis

In this section, we analyze the proposed greedy strategy by providing the approximation ratio, then discuss the time complexity.

4.6.1 Appropriateness of the Greedy Strategy

Since our approach is related to the greedy algorithm for weighted set cover problem, which achieves an approximation ratio of $\ln(n) + 1$ over the optimal solution, we discuss the approximation ratio of MPTC in this section.

Theorem 4.4. *Given an itemset I and its supporting transactions $\mathcal{T}(I)$, the function get_tile produces the tile $R = T \times I$ such that for any $R' \subseteq \mathcal{T}(I) \times I$, $R \prec_\rho R'$.*

We prove Theorem 4.4 by the following idea. Suppose the tile R^* satisfies that $R^* \prec_\rho R'$ for all $R' \in \mathcal{T}(I)$. Given $R_i = \{t_i\} \times I \subseteq R^*$ and $R_j = \{t_j\} \times I \subseteq \mathcal{T}(I) \times I$, if $R_j \prec_\rho R_i$, then $R_j \subseteq R^*$. This claim implies that R^* can be generated by iteratively adding transactions in the ascending probabilistic price order till the probabilistic price of result set increasing, which is the same as the process of the function *get_tile*.

Proof. Suppose the tile $R^* = T^* \times I \subseteq \mathcal{T}(I) \times I$ satisfies that for all $R' \subseteq \mathcal{T}(I) \times I$, $R^* \prec_\rho R'$. We first prove that given two tiles $R_i = \{t_i\} \times I \subseteq R^*$ and $R_j = \{t_j\} \times I \subseteq \mathcal{T}(I) \times I$, if $R_j \prec_\rho R_i$, then $R_j \subseteq R^*$. For brevity, we denote $\mathrm{q}(R; Z)$ as $\mathrm{q}(R)$ and $\rho(R; Z)$ as $\rho(R)$ in this proof because the covered cell set Z does not change in our discussion.

We assume $R_j \nsubseteq R^*$ first. Let $T' = T^* \setminus \{t_i\}$ and $R' = T' \times I$.

Then, we have

$$\rho(R^*) = \frac{\mathrm{c}(R^*)}{\mathrm{q}(R^*)} = \frac{|T^*| + |I|}{\mathrm{q}(R^*)} = \frac{\mathrm{c}(R') + 1}{\mathrm{q}(R') + \mathrm{q}(R_i)}. \tag{4.8}$$

Since $R^* \prec_\rho R'$, we have

$$\Pr\left[\rho(R^*) < \rho(R')\right] > \Pr\left[\rho(R^*) > \rho(R')\right]. \tag{4.9}$$

With the result in Equation (4.8), Inequation (4.9) can be stated as follows:

$$\Pr\left[\mathrm{q}(R') < \mathrm{c}(R')\mathrm{q}(R_i)\right] > \Pr\left[\mathrm{q}(R') > \mathrm{c}(R')\mathrm{q}(R_i)\right]. \tag{4.10}$$

On the other hand, Since $R_j \prec_\rho R_i$, we have

$$\begin{aligned} &\Pr\left[\rho(R_j) < \rho(R_i)\right] > \Pr\left[\rho(R_j) > \rho(R_i)\right] \\ \Rightarrow &\Pr\left[\mathrm{q}(R_i) < \mathrm{q}(R_j)\right] > \Pr\left[\mathrm{q}(R_i) > \mathrm{q}(R_j)\right]. \end{aligned} \tag{4.11}$$

Combine Equations (4.10) and (4.11), we have

$$\begin{aligned} &\Pr\left[\mathrm{q}(R') < \mathrm{c}(R')\mathrm{q}(R_j)\right] > \Pr\left[\mathrm{q}(R') > \mathrm{c}(R')\mathrm{q}(R_j)\right] \\ \Rightarrow &\Pr\left[\mathrm{q}(R') + \mathrm{q}(R_i) < \mathrm{c}(R')\mathrm{q}(R_j) + \mathrm{q}(R_j)\right] \\ &> \Pr\left[\mathrm{q}(R') + \mathrm{q}(R_i) > \mathrm{c}(R')\mathrm{q}(R_j) + \mathrm{q}(R_j)\right] \\ \Rightarrow &\Pr\left[\mathrm{q}(R^*) < (\mathrm{c}(R')+1)\mathrm{q}(R_j)\right] > \Pr\left[\mathrm{q}(R^*) > (\mathrm{c}(R')+1)\mathrm{q}(R_j)\right] \\ \Rightarrow &\Pr\left[\mathrm{c}(R^*)\mathrm{q}(R^*) + \mathrm{q}(R^*) < \mathrm{c}(R^*)\mathrm{q}(R^*) + \mathrm{c}(R^*)\mathrm{q}(R_j)\right] \\ &> \Pr\left[\mathrm{c}(R^*)\mathrm{q}(R^*) + \mathrm{q}(R^*) > \mathrm{c}(R^*)\mathrm{q}(R^*) + \mathrm{c}(R^*)\mathrm{q}(R_j)\right] \\ \Rightarrow &\Pr\left[\frac{\mathrm{c}(R^*)+1}{\mathrm{q}(R^*)+\mathrm{q}(R_j)} < \frac{\mathrm{c}(R^*)}{\mathrm{q}(R^*)}\right] > \Pr\left[\frac{\mathrm{c}(R^*)+1}{\mathrm{q}(R^*)+\mathrm{q}(R_j)} > \frac{\mathrm{c}(R^*)}{\mathrm{q}(R^*)}\right] \end{aligned} \tag{4.12}$$

Equation (4.12) indicates that $R^* \cup R_j \prec_\rho R^*$, which means adding R_j to R^* reduces the probabilistic price of R^*. This contradicts with the assumption that $R^* \prec_\rho R'$ for all $R' = T' \times I$ such that $T' \subseteq \mathcal{T}(I)$. Hence, we have $R_j \subseteq R^*$.

This claim also implies that R^* can be generated by iteratively adding transactions in the ascending probabilistic price order till the probabilistic price of result set increasing, which is the same as the process of the function *get_tile* and completes the proof. □

Based on Theorem 4.4, we prove that the approximation ratio of our algorithm is also $\ln(n) + 1$ as follows.

Theorem 4.5. *For the problem of Minimum Probabilistic Tile Cover Mining, given a candidate set $\mathcal{C}$, the proposed approach has the same approximation ratio as the* *greedy algorithm for weighted set cover problem, which is* $\ln(n) + 1$, *where n is the number of transactions in database.*

Theorem 4.5 can be proved by showing that in each iteration, the tile produced by MPTC mining R and weighted set cover R^* satisfy that neither $R \prec_\rho R^*$ nor $R^* \prec_\rho R$. Then, the approximation factor can be proved following proof of the greedy weighted set cover algorithm [39].

Proof. We first prove that given the result tile produced by MPTC mining and weighted set cover $R = T \times I$ and $R^* = T^* \times I^*$, respectively, $R \approx_\rho R^*$, which means neither $R \prec_\rho R^*$ nor $R^* \prec_\rho R$. Then, the approximation factor can be proved following the proof of greedy weighted set cover algorithm [39].

Since R is a tile generated from the candidate set, and R^* is produced by the greedy algorithm, which always chooses the tile with lowest probabilistic price order, we have that $R \prec_\rho R^*$ is invalid.

On the other side, suppose $R^* \prec_\rho R$. Consider that there must exist a tile $R' = T' \times I'$ such that $I' = I^*$ and $T^* \subseteq T' \subset \mathcal{T}(I')$. According to Theorem 4.4, our algorithm should return a tile with the probabilistic price order not higher than R^*. Therefore, we have neither $R \prec_\rho R^*$ nor $R^* \prec_\rho R$, which completes the proof. □

Consider that the set cover problem has been proven cannot be approximated within factor $(1 - c)\ln(n)$ in polynomial time for every $c > 0$ unless $P = NP$ [51], the approximation ratio of the algorithm cannot be improved further.

4.6.2 Time Complexity

Let κ be the number of discovered probabilistic tiles. Then, there are κ iterations in Algorithm 4.1. In each iteration, the algorithm checks each candidate in $\mathcal{C}$ to find the best tile and decides if it should be added to the result set by calculating the probabilistic price order. Suppose the time complexities of the function *get_tile* and the operation of determining $\prec_\rho$ are τ_1 and τ_2, respectively. The overall complexity is $O(\kappa \cdot |\mathcal{C}| \cdot (\tau_1 + \tau_2))$. The number of tiles κ and candidates $|\mathcal{C}|$ depend on the characteristics of the database and the parameter *minsup*, which is further analyzed in Section 4.7.2.

In the function *get_tile*, we fetch and sort the supporting transactions of each candidate, then iteratively insert them to the tile until adding a new transaction violates the conditions $R' \prec_\rho R^*$ or $\Pr[\mathrm{d}(\mathcal{R} \cup \{R'\}) > \kappa] > \xi$. Since $\Pr[\mathrm{d}(\mathcal{R} \cup \{R'\}) > \kappa] > \xi$ can be approximated by normal distribution, assuming the cumulative distribution function of standard normal distribution can be calculated in constant time, the complexity is $O(1)$. Hence, the worst-case complexity of the function *get_tile* is $O(mn + n \log n + n\tau_2)$, where $O(mn)$, $O(n \log n)$ and $O(n\tau_2)$ are the complexities of fetching, sorting and inserting processes, respectively.

The complexity of deciding the probabilistic price order $\prec_\rho$, which computes the probability $\Pr[\rho(R_1; Z) < \rho(R_2; Z)]$, is $O(m^3n^3)$. We analyze it as follows. Let $c = \mathrm{c}(R_1)/\mathrm{c}(R_2)$. According to Equations (4.5) and (4.6), we have

$$\begin{aligned}
&\Pr[\rho(R_1; Z) < \rho(R_2; Z)] \\
&= \sum_{k=0}^{q_2} \sum_{l=\lceil \mathrm{c}(R_1)\cdot k/\mathrm{c}(R_2)\rceil}^{q_1} \Pr\left[\mathrm{q}(R_1; Z) = l, \mathrm{q}(R_2; Z) = k\right] \\
&= \sum_{k=0}^{q_2} \sum_{l=\lceil c\cdot k\rceil}^{q_1} \sum_{i=0}^{q_0} \Pr\left[\mathrm{q}(R_1 \cap R_2; Z) = i\right] \cdot \\
&\quad \Pr\left[\mathrm{q}(R_1 \setminus R_2; Z) = k - i\right] \cdot \Pr\left[\mathrm{q}(R_2 \setminus R_1; Z) = l - i\right] \qquad (4.13)
\end{aligned}$$

Equation (4.13) implies that $q_1 - \lceil c \cdot k \rceil \geq 0 \Rightarrow k \leq q_1/c$, otherwise the result of the second sum will be 0. Similar to calculating the density, the time complexity of computing $\Pr\left[\mathrm{q}(R; Z) = i\right]$ is $O(1)$. According to Equation (4.13), we obtain the overall complexity by summing up the number of operations as follows:

$$\begin{aligned}
O\left(\sum_{k=0}^{q_2} (q_1 - \lceil c \cdot k \rceil) q_0\right) &= O\left((q_1 + (q_1 - c) + \cdots + (q_1 - \lceil (q_1/c) \cdot c \rceil)) q_0\right) \\
&= O(q_1^2 \cdot q_0). \qquad (4.14)
\end{aligned}$$

In the worst case, q_0 and q_1 may reach $m \cdot n$, thus the complexity is $O(m^3n^3)$.

The complexity of the whole algorithm is then $O(\kappa \cdot |\mathcal{C}| \cdot (mn + n \log n + n \cdot m^3 n^3 + m^3 n^3)) = O(\kappa \cdot |\mathcal{C}| \cdot m^3 n^4)$. However, recall that q_1 is the maximum value of $\mathrm{q}(R_1; Z) = |(R \setminus Z) \cap \mathcal{D}|$. It is normally much less than $m \cdot n$ due to the sparsity of database. The value of q_0, which is the maximum value of $\mathrm{q}(R_1 \cap R_2; Z)$, is even smaller. In addition, if two probabilistic tiles differ by one transaction, the worst-case complexity reduces to $O(m^2 \cdot n^2 \cdot (m+n))$ according to Theorem 4.3. Therefore, normally the runtime is much less than the worst-case. The efficiency of MPTC mining on real world datasets and the effectiveness of optimization techniques are illustrated in Section 4.7.2.

4.7 Performance Study

In this section, we conduct experiments and show the effectiveness and efficiency of our approach. We first demonstrate the performance of the proposed algorithm by comparing MPTC with state-of-the-art algorithms. Then, we apply our method on two real world datasets and analyze the results.

4.7.1 Experiments on Synthetic Datasets

Evaluating the discovered tiles may be difficult, because the "correct" tiles in a given dataset are unknown. Thus we do not have the ground truth to compare with. Inspired by [47], we devised a process to generate synthetic dataset by embedding a set of probabilistic tiles into the dataset as the ground truth, and then introducing noise to test the robustness. For the evaluation measurements, we developed a probabilistic version of the F-measure, which is widely used in classification and clustering evaluation.

4.7.1.1 Dataset Generation

We created the synthetic datasets as follows. We first generated a set of probabilistic tiles $\mathcal{R}^*$, which were used as the ground truth and embedded into a dataset. Suppose we want to generate a synthetic dataset containing m items $\mathcal{I} = \{e_1, \ldots, e_m\}$ and n transactions $\mathcal{D} = \{t_1, \ldots, t_n\}$. Given the parameters of the minimum and maximum ratio of items and transactions, $I_{\min}$, $I_{\max}$, $T_{\min}$ and $T_{\max}$, we constructed a tile $R = T \times I$ by uniformly sampling an itemset I from $\mathcal{I}$ and a transaction set T from $\mathcal{D}$ with the constraints that $I_{\min} \cdot m < |I| < I_{\max} \cdot m$ and $T_{\min} \cdot n < |T| < T_{\max} \cdot n$. In order to guarantee that the generated tiles did not repeatedly express the same portion of data, we set a maximum overlapping ratio w and restrained that for any two generated tiles $R_1 = T_1 \times I_1$ and $R_2 = T_2 \times I_2$, $|T_1 \cap T_2| < w \cdot \min\{|T_1|, |T_2|\}$ and $|I_1 \cap I_2| < w \cdot \min\{|I_1|, |I_2|\}$. For tile R, we also generated its existential probability $\Pr(R)$ from a normal distribution $\mathcal{N}(\mu, \sigma^2)$, where μ and σ^2 were two parameters to control the mean and variance. The synthetic dataset was then produced by merging the cells of generated probabilistic tiles.

Next, we introduced two types of noise to the dataset: 1-to-0 noise and 0-to-1 noise. Given a transaction-item pair (t, e) contained in one of the generated tiles with existential probability $\Pr(t, e)$, we added 1-to-0 noise to it with the probability $\epsilon_{1\to 0}$, which is a parameter controlling how likely the 1-to-0 noise is introduced, by probabilistically flipping it to 0. The probabilistic flipping operation was implemented by subtracting a probability $p \sim \mathcal{N}(\mu_\epsilon, \sigma_\epsilon^2)$ from $\Pr(t, e)$, where μ_ϵ and σ_ϵ were the mean and variance of a normal distribution and regulated the extent of the flipping operation. Similarly, given the parameter controlling 0-to-1 noise, $\epsilon_{1\to 0}$, if transaction-item pair (t, e) was not contained by any tiles, we probabilistically flipped it to 1 with the probability $\epsilon_{1\to 0}$ by setting $\Pr(t, e) = p \sim \mathcal{N}(\mu_\epsilon, \sigma_\epsilon)$. Note that the probability after flipping was always bounded in $[0, 1]$ to ensure its validity. If $\epsilon_{0\to 1} = \epsilon_{1\to 0}$, we denote them as a uniform probability ϵ. The meaning and values of parameters used in the experiments are summarized in Table 4.3.

4.7.1.2 Evaluation Measurement

F-measures are widely used to evaluate the quality of result for various problems, such as classification and clustering [52]. Several metrics based on F-measure were proposed in [47] to assess the quality of tiles in terms of item, transaction and pattern, respectively. Since they were designed for deterministic data, we adapted them to uncertain data and defined the probabilistic versions of precision and recall.

Parameter	Description	Value
m	number of items	$10, \dots, 50$
n	number of transactions	$500, \dots, 2500$
$I_{\min}$	minimum ratio of items	0.1
$I_{\max}$	maximum ratio of items	0.3
$T_{\min}$	minimum ratio of transactions	0.1
$T_{\max}$	maximum ratio of transactions	0.3
w	overlapping ratio between tiles	0.5
μ	mean of the probability of tiles	0.6
σ^2	variance of the probability of tiles	0.1
ϵ	probability of adding noise	$0.05, \dots, 0.30$
μ_ϵ	mean of noise distribution	0.6
σ_ϵ^2	variance of noise distribution	0.1

Table 4.3: Parameters used in experiments.

Given a discovered probabilistic tile $R' = T' \times I'$ and a ground truth probabilistic tile $R^* = T^* \times I^*$, let $\Pr(R)$ be the existential probability of tile R, we defined the item-based probabilistic precision and recall as follows.

$$\mathrm{Prec}^I(R', R^*) = \frac{|I' \wedge I^*|}{|I'|} \cdot \left[1 - |\Pr(R') - \Pr(R^*)|\right] \tag{4.15}$$

$$\mathrm{Recall}^I(R', R^*) = \frac{|I' \wedge I^*|}{|I^*|} \cdot \left[1 - |\Pr(R') - \Pr(R^*)|\right] \tag{4.16}$$

In the design of the metrics, we aimed to measure the similarity of tiles from the perspectives of both the itemset and the existential probability. Hence, we employed the Manhattan similarity of the existential probabilities (i.e., $1 - |\Pr(R') - \Pr(R^*)|$) as a weight of the item-based precision and recall of the discovered tile.

As discussed, each tile in the ground truth set has a probability. Since a discovered tile implied an underlying concept in the dataset, we treated the existence of a tile as a single probabilistic event and calculated the existential probability of the tile as the mean of the existential probabilities of all the included cells.

With the definitions of probabilistic precision and recall, the probabilistic F1-score can be obtained as follows.

$$\mathrm{F1}^I(R', R^*) = \frac{2\mathrm{Prec}^I(R', R^*) \cdot \mathrm{Recall}^I(R', R^*)}{\mathrm{Prec}^I(R', R^*) + \mathrm{Recall}^I(R', R^*)} \tag{4.17}$$

Based on the F1-score of two tiles, we discuss the F1-score of two probabilistic tile sets next. Suppose the ground truth tile set is $\mathcal{R}^*$, and the discovered tile set is $\mathcal{R}'$. Similar to [47], we defined the item-based probabilistic F1-score as follows.

$$\mathcal{F}1^I(\mathcal{R}', \mathcal{R}^*) = \frac{\sum_{R^* \in \mathcal{R}^*} |I^*| \cdot \max_{R' \in \mathcal{R}'} \mathrm{F1}^I(R', R^*)}{\sum_{R^* \in \mathcal{R}'^*} |I^*|} \tag{4.18}$$

In the definition of $\mathcal{F}1^I(\mathcal{R}', \mathcal{R}^*)$, for each tile in the ground truth, we considered the discovered tile with highest F1-score as the corresponding result. Since it is more difficult to achieve higher F1-score when the size of the itemset of the ground truth tile increases, the F1-score is weighted by $|I^*|$. The denominator is to normalize the metric to $[0, 1]$.

The transaction-based probabilistic F1-score (i.e., $\mathcal{F}1^T$) and cell-based probabilistic F1-score (i.e., $\mathcal{F}1^C$) for tile set can be defined similarly. Note that the cell-based probabilistic F1-score calculates with respect to the Cartesian product of the itemset and the transaction set.

4.7.1.3 Baseline Methods

We compared our method with following baseline methods: ASSO [53], Panda+ [47] , Hyper+ [46] and STIJL [54]. These methods were developed for deterministic database and tackled the tile covering problem from different perspectives. ASSO aimed to minimize the difference between discovered tiles and the dataset; Panda+ and STIJL solved the problem by minimizing the Minimum Description Length encoding; while Hyper+ discovered a tile set to minimize the total cost. Since none of the baseline methods can directly handle the uncertainty in the synthetic datasets, we rely on the sampling technique, which is a promising tool for uncertain data mining [55]. For each experiment, we uniformly sampled the data 10 times and

n		500	1000	1500	2000	2500
ASSO	$\mathcal{F}1^I$	0.240	0.241	0.238	0.241	0.256
	$\mathcal{F}1^T$	0.333	0.345	0.335	0.328	0.332
	$\mathcal{F}1^C$	0.367	0.375	0.371	0.354	0.375
Panda+	$\mathcal{F}1^I$	0.413	0.428	0.428	0.421	0.456
	$\mathcal{F}1^T$	0.145	0.151	0.147	0.131	0.131
	$\mathcal{F}1^C$	0.246	0.262	0.253	0.244	0.284
Hyper+	$\mathcal{F}1^I$	0.304	0.331	0.298	0.286	0.330
	$\mathcal{F}1^T$	0.376	0.377	0.374	0.371	0.368
	$\mathcal{F}1^C$	0.418	0.419	0.424	0.401	0.427
STIJL	$\mathcal{F}1^I$	0.231	0.254	0.257	0.258	0.258
	$\mathcal{F}1^T$	0.210	0.221	0.217	0.207	0.207
	$\mathcal{F}1^C$	0.134	0.254	0.261	0.317	0.284
MPTC	$\mathcal{F}1^I$	0.652	0.631	0.651	0.586	0.602
	$\mathcal{F}1^T$	0.478	0.446	0.514	0.582	0.535
	$\mathcal{F}1^C$	0.545	0.522	0.544	0.583	0.549

Table 4.4: Probabilistic F1-score w.r.t. the number of transactions n.

reported the average performance. The parameters for the four methods were adjusted to achieve the best performance. In addition, the probability of result tiles were set at 1 because the sampled dataset did not contain uncertainty information.

Our method was implemented in C++. The software of the baseline methods were obtained from the authors. The experiments were conducted on a computer with 16 4-core CPUs and 32GB memory running Linux Red hat 6.5.

4.7.1.4 Comparison Study

We compared MPTC with the baseline methods by varying the following four parameters in synthetic data generation: the number of transactions n, the number of items m, the number of ground truth tiles k and the probability of noise ϵ. The default value of them were $n = 1000$, $m = 20$, $k = 10$ and $\epsilon = 0.1$. Since the process of dataset generation involved randomization, we conducted each experiment for 10 times and reported the mean of the

m		10	20	30	40	50
ASSO	$\mathcal{F}1^I$	0.390	0.241	0.158	0.139	0.104
	$\mathcal{F}1^T$	0.364	0.345	0.311	0.311	0.305
	$\mathcal{F}1^C$	0.483	0.375	0.296	0.278	0.239
Panda+	$\mathcal{F}1^I$	0.506	0.428	0.374	0.367	0.345
	$\mathcal{F}1^T$	0.113	0.151	0.164	0.169	0.180
	$\mathcal{F}1^C$	0.388	0.262	0.234	0.228	0.227
Hyper+	$\mathcal{F}1^I$	0.408	0.331	0.234	0.209	0.185
	$\mathcal{F}1^T$	0.358	0.377	0.388	0.395	0.396
	$\mathcal{F}1^C$	0.489	0.419	0.402	0.395	0.405
STIJL	$\mathcal{F}1^I$	0.294	0.254	0.252	0.235	0.236
	$\mathcal{F}1^T$	0.214	0.221	0.211	0.209	0.211
	$\mathcal{F}1^C$	0.207	0.254	0.333	0.370	0.369
MPTC	$\mathcal{F}1^I$	0.764	0.631	0.555	0.515	0.455
	$\mathcal{F}1^T$	0.520	0.446	0.461	0.507	0.502
	$\mathcal{F}1^C$	0.737	0.522	0.420	0.402	0.366

Table 4.5: Probabilistic F1-score w.r.t. the number of items m.

F1-scores.

The result $\mathcal{F}1^I$, $\mathcal{F}1^T$ and $\mathcal{F}1^C$ of all approaches with respect to the variation of n, m, k and ϵ are shown in Tables 4.4, 4.5, 4.6 and 4.7, respectively. Our approach beats all the baseline methods on all the settings.

Table 4.4 shows the F1-scores with respect to the variation of n. It is interesting that the baseline methods behave differently. For ASSO and Hyper+, the $\mathcal{F}1^I$ is normally smaller than $\mathcal{F}1^T$, while Panda+ and STIJL achieve better results of $\mathcal{F}1^I$ than $\mathcal{F}1^T$. This trend can also be found in the other experiments. In Table 4.5, we observe that the $\mathcal{F}1^I$s of all the methods decrease when m increases. This is because when the number of items is larger, more possible tiles exist in the dataset, which expands the search space and makes the dataset more challenging. In Table 4.6, the $\mathcal{F}1^C$s of all the methods except ASSO drop when k increases. This is because a larger k means more tiles are required to be found, which makes the task more difficult. For ASSO, since the number of result tiles k is an input parameter, it performs relatively stable with respect to the variation of k. In Table 4.7,

k		5	10	15	20
ASSO	$\mathcal{F}1^I$	0.234	0.241	0.237	0.244
	$\mathcal{F}1^T$	0.376	0.345	0.298	0.279
	$\mathcal{F}1^C$	0.331	0.375	0.385	0.397
Panda+	$\mathcal{F}1^I$	0.451	0.428	0.391	0.372
	$\mathcal{F}1^T$	0.115	0.151	0.165	0.178
	$\mathcal{F}1^C$	0.290	0.262	0.227	0.222
Hyper+	$\mathcal{F}1^I$	0.315	0.331	0.295	0.283
	$\mathcal{F}1^T$	0.394	0.377	0.345	0.313
	$\mathcal{F}1^C$	0.398	0.419	0.429	0.411
STIJL	$\mathcal{F}1^I$	0.277	0.254	0.247	0.235
	$\mathcal{F}1^T$	0.207	0.221	0.208	0.207
	$\mathcal{F}1^C$	0.257	0.254	0.230	0.238
MPTC	$\mathcal{F}1^I$	0.527	0.631	0.695	0.700
	$\mathcal{F}1^T$	0.642	0.446	0.365	0.300
	$\mathcal{F}1^C$	0.605	0.522	0.503	0.496

Table 4.6: Probabilistic F1-score w.r.t. the number of ground truth tiles k.

the F1-scores of all the methods decrease, because a higher value of ϵ means more noise are introduced to the dataset, which adds more difficulty to the task.

In summary, our approach is better than the baseline methods with respect to the variation of n, m, k and ϵ, which demonstrate the effectiveness and applicability of MPTC mining.

4.7.2 Experiments on Real World Datasets

We first describe the real world datasets, then analyze the results of three groups of experiments to evaluate the cost and the runtime of MPTC mining, the effectiveness of optimization techniques, and the influence of candidate selection.

ϵ		0.05	0.10	0.15	0.20	0.25	0.30
ASSO	$\mathcal{F}1^I$	0.241	0.222	0.232	0.237	0.256	0.239
	$\mathcal{F}1^T$	0.340	0.329	0.320	0.309	0.306	0.282
	$\mathcal{F}1^C$	0.362	0.347	0.346	0.336	0.355	0.320
Panda+	$\mathcal{F}1^I$	0.424	0.395	0.395	0.395	0.411	0.389
	$\mathcal{F}1^T$	0.139	0.151	0.139	0.139	0.134	0.137
	$\mathcal{F}1^C$	0.262	0.232	0.213	0.216	0.211	0.196
Hyper+	$\mathcal{F}1^I$	0.306	0.275	0.287	0.281	0.293	0.285
	$\mathcal{F}1^T$	0.382	0.372	0.351	0.341	0.333	0.304
	$\mathcal{F}1^C$	0.411	0.409	0.391	0.392	0.403	0.382
STIJL	$\mathcal{F}1^I$	0.247	0.248	0.242	0.245	0.239	0.237
	$\mathcal{F}1^T$	0.208	0.216	0.210	0.208	0.206	0.204
	$\mathcal{F}1^C$	0.254	0.254	0.222	0.211	0.175	0.179
MPTC	$\mathcal{F}1^I$	0.598	0.620	0.608	0.615	0.613	0.612
	$\mathcal{F}1^T$	0.406	0.422	0.490	0.495	0.469	0.420
	$\mathcal{F}1^C$	0.410	0.449	0.467	0.462	0.461	0.388

Table 4.7: Probabilistic F1-score w.r.t. the probability of noise ϵ.

Dataset	#Trans.	#Items	Density
Equip	4224	202	18.33%
IIP	35161	467	0.86%

Table 4.8: Characteristics of datasets.

4.7.2.1 Datasets and Experiment Setting

We ran experiments on two datasets with data characteristics listed in Table 4.8.

The Equip dataset is provided by an equipment rental company, including the equipment demand information with locations collected from January to December in 2013. Each row corresponding to an equipment type consists of a set of branches, where each branch is associated with a probability indicating how likely the given equipment type is rented from the branch. The probability was produced according to the distance between the locations of equipment demand and the branch.

The other one is the IIP dataset, which records the iceberg sightings from 1993 to 1997 on the North Atlantic from the International Ice Patrol (IIP) Iceberg Sightings Database.[1] Each transaction in the database contains the information of an iceberg sighting record, including date, location, size, shape, reporting source and a confidence level. There are six possible values of the confidence level indicating different reliabilities: R/V (Radar and visual), R (Radar only), V (Visual), MEA (Measured), EST (Estimated) and GBL (Garbled). We converted the confidence levels to probabilities 0.8, 0.7, 0.6, 0.5, 0.4 and 0.3, respectively. This dataset is an uncertain database with tuple uncertainty.

We discuss the default values of the parameters. There are three parameters of MPTC mining: η, θ and ξ. As discussed in Section 4.2, users are suggested to set the value of η by u and v, and θ by a relative value λ. We set $u = 10$ and $v = 0.1$, $\lambda = 0.5$ and $\xi = 0.5$ as the default values. For the parameters of P-RFP used for candidate generation, we set $minsup = 5\%$ and 0.5% for Equip and IIP, respectively, and $minprob = 0.5$, $\varepsilon = 0.1$, $\delta = 0.1$ for both datasets. Users may refer to [40] for the meaning and setting of the P-RFP parameters.

4.7.2.2 Performance of MPTC Mining

We evaluated the performance of MPTC Mining on the two datasets under different settings of density threshold, number of candidates and cover probability by varying the parameters λ, $minsup$ and u, respectively.

Figures 4.1 and 4.2 demonstrate the *cost* and the *runtime* with respect to the variation of u on the Equip dataset. The results on IIP dataset are shown in Figures 4.3 and 4.4. We compared the costs of summarized tiles with that of the database represented by individual cells, which is denoted by DB. We observe from Figures 4.1 (a)–4.4 (a) that MPTC mining can effectively summarize the database with much lower cost. The figures also show that both the cost and the runtime decrease when u increases. This is because a larger u indicates more cells are allowed to be uncovered and, consequently, fewer tiles are required to be constructed.

[1]http://nsidc.org/data/g00807.html

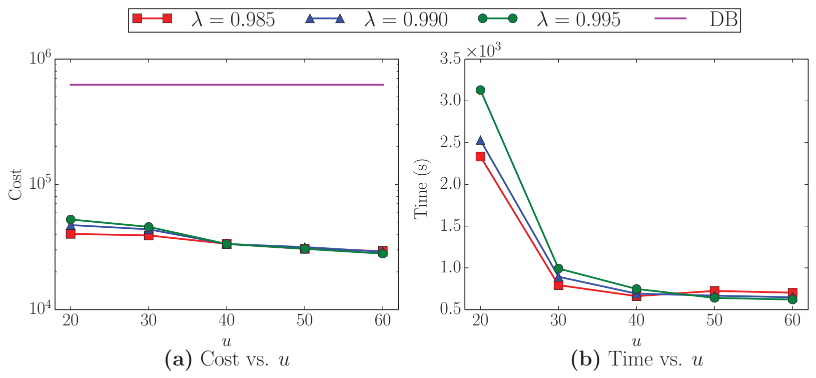


Figure 4.1: Performance w.r.t. λ on Equip.

It can be observed from Figures 4.1 and 4.3 that when λ increases, the algorithm finishes with a higher cost and longer runtime. The reason is that when λ is larger, the generated tiles tend to be denser and cover more cells at first, hence the cost is lower and the runtime is shorter. Then, more small tiles are produced because the density constraint forbids to add larger tiles with lower density into the result set, even if it covers more cells, which leads to higher cost and longer runtime.

Figures 4.2 and 4.4 show that when *minsup* increases, the cost grows while the runtime decreases. This is because the growth of *minsup* leads to the decline of the number of candidates. Thus, more cells are covered by the tiles constructed from individual items, which increases the total cost but consumes less time.

Figure 4.5 (a) shows the number of tiles with respect to u by varying the value of λ on IIP dataset. It can be observed that the number of discovered tiles decreases when u increases. Figure 4.5 (b) demonstrates the number of tiles with respect to u with the variation of *minsup* on Equip dataset. We also observe that the number of tiles reduces with the increase of *minsup*. This is because a lower value of *minsup* indicates that more candidates are generated. In addition, we notice that when u is small, the number of tiles is relatively large. This is because many small tiles are generated in order to reach the constraint of cover probability. Then the number of tiles drops significantly when the value of u grows.

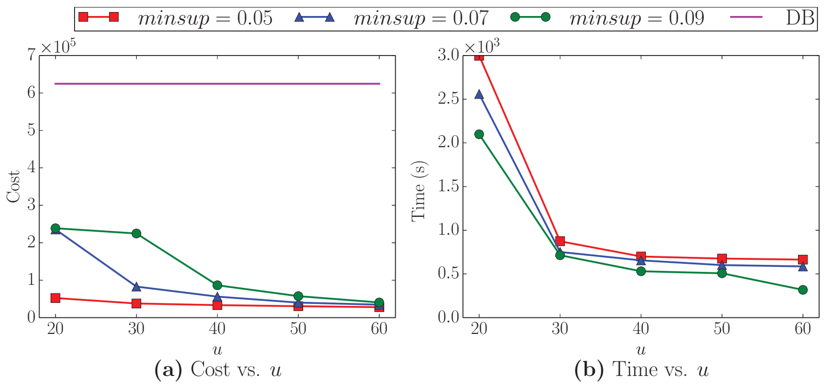


Figure 4.2: Performance w.r.t. *minsup* on Equip.

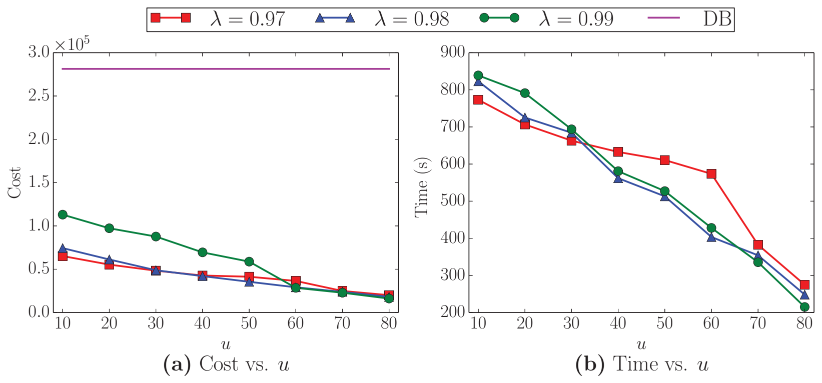


Figure 4.3: Performance w.r.t. λ on IIP.

4.7.2.3 Effectiveness of Optimization Techniques

We studied the effectiveness of the three optimization techniques introduced in Section 4.5: Optimizing single transaction difference (denoted by OPT1), Adaptively computing cover quantity (denoted by OPT2), and Pruning by 3σ property (denoted by OPT3). The methods without any and with all optimization techniques are denoted by Base and MPTC, respectively. Since the Base method is slow, we randomly picked 100, 200, . . . , 500 transactions from the Equip dataset to demonstrate the effectiveness.

We analyzed the effectiveness of the optimization techniques by evaluating the runtime and the cost. Figure 4.6 (a) illustrates the runtime with respect to the variation of the size of dataset. It shows that OPT1, OPT2 and OPT3 accelerate the algorithm for approximately 100 times, 10 times and twice. Figure 4.6 (b) shows the cost of discovered tiles. It can be observed

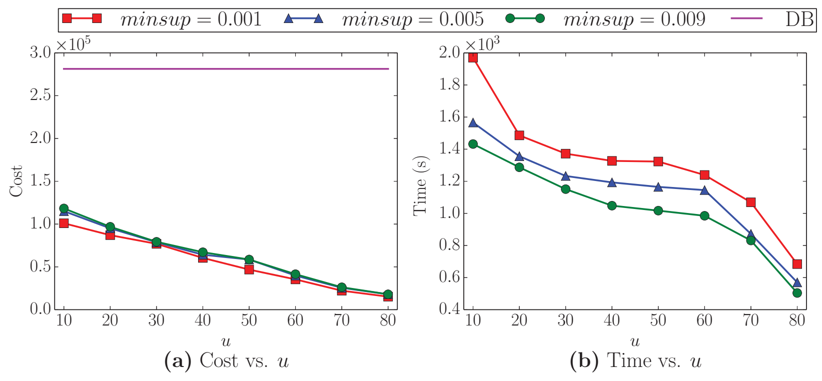


Figure 4.4: Performance w.r.t. *minsup* on IIP.

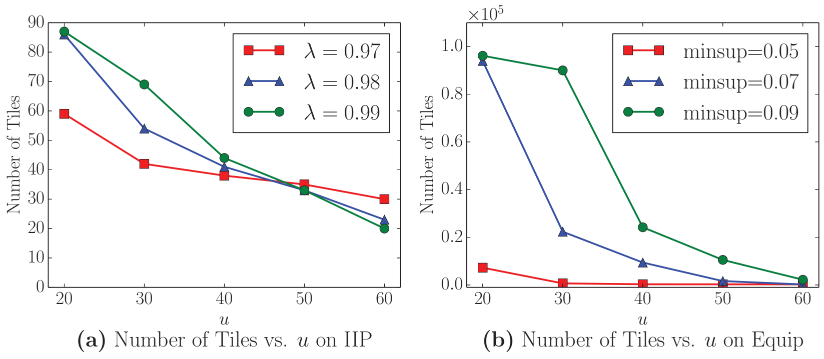


Figure 4.5: Number of Tiles on Equip and IIP.

that the costs of tiles generated with and without optimization techniques differ very slightly. Therefore, the optimization techniques effectively speed up the algorithm with slightly increase of cost.

4.7.2.4 Analysis of Candidate Selection

We further analyzed the number of candidates and the influence of parameter u in the process of candidate selection on the Equip dataset.

Figures 4.7 (a) and (b) shows the number of candidates with respect to the variation of *minsup* and *minprob*, respectively. We set $\varepsilon = 0.1$ and $\delta = 0.1$ for both figures, and $minprob = 0.5$ for Figure 4.7 (a) and $minsup = 5\%$ for Figure 4.7 (b). It clearly shows that the number of Probabilistic Representative Frequent Pattern (P-RFP) is less than the number of Probabilistic Frequent Pattern (PFP), especially when *minsup* is small, which corre-

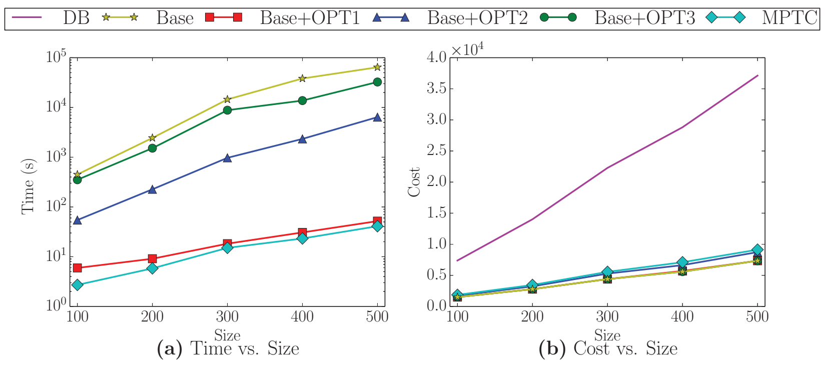


Figure 4.6: Performance of optimization techniques.

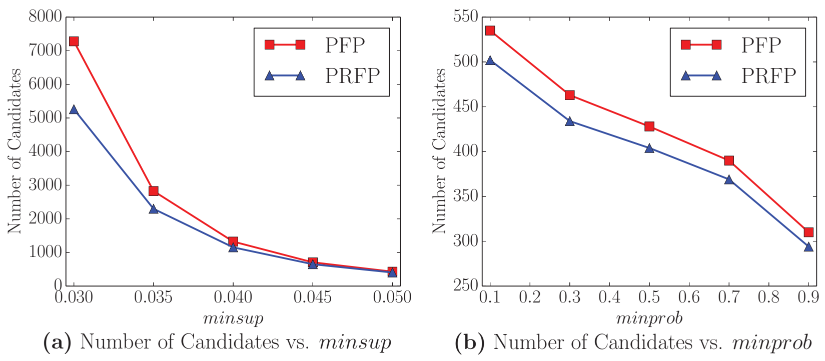


Figure 4.7: Number of candidates

sponds to our discussion in Section 4.3.3.

Figure 4.8 (a) shows that compared with Probabilistic Frequent Pattern (PFP), using Probabilistic Representative Frequent Pattern (P-RFP) reduces the runtime, especially when u is small. This is because when u is large, the resulted tiles are prone to be produced by longer patterns, which very likely exist in both PFP set and P-RFP set. Hence, using P-RFP set as candidate improves the efficiency of MPTC Mining when a higher cover probability is desired. Figure 4.8 (b) shows that the cost difference between the results of using the two types of candidates is neglectable.

4.8 Related Work

In this section, we review related works in two research areas: transaction data summarization, and frequent pattern summarization.

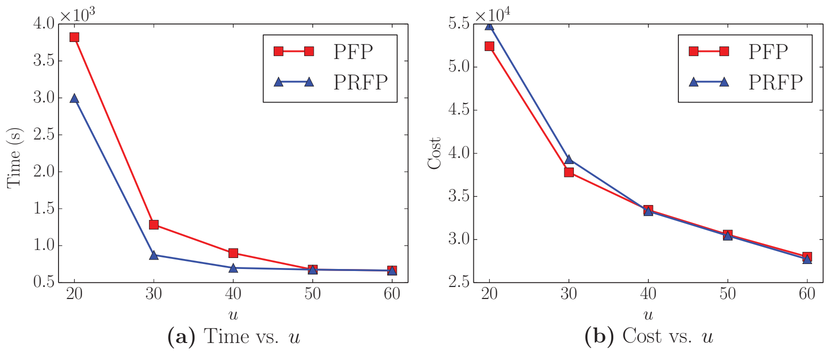


Figure 4.8: Cost and Time w.r.t. u

4.8.1 Transaction data summarization

Tiling database was previously proposed to extract knowledge from binary transaction data by covering a database with tiles [45, 56]. It finds noiseless tiles to cover the original database and does not handle the false positives presented in dataset. According to [54], tiles can also be hierarchical. Miettinen et al. [53] proposed the ASSO algorithm to solve the Binary Matrix Factorization problem, which factorizes an $m \times n$ binary matrix into an $m \times k$ matrix and a $k \times n$ matrix. The result can also be interpreted as using k tiles to cover the original database. Xiang et al. [46] defined the concept of hyperrectangle, which is similar to a tile, and formulated the problem of succinctly summarizing transaction databases. Hyperrectangle was also adopted to reduce the amount of generated frequent patterns [57]. Lucchese et al. [47] presented a unified framework for mining approximate binary patterns and propose a Minimum Description Length-based method. However, as discussed, the data uncertainty leads to challenging issues for database summarization. Therefore, none of these methods can be applied to summarize uncertain databases. Recently, Bonchi et al. [58] proposed a framework for summarizing uncertain databases with a set of patterns based on the Minimum Description Length Principle. Since this aims at generating itemsets instead of tiles, it works on a problem different from our method.

4.8.2 Frequent pattern summarization

Motivated by the fact that frequent pattern mining may generate an exponential number of patterns due to the anti-monotonicity of the support measure, much research effort has been dedicated to frequent pattern summarization, which aims to obtain a much smaller but meaningful set of patterns to represent the complete set of frequent patterns. A variety of definitions have been proposed, such as maximal patterns [29], closed patterns [30], condensed pattern bases [59, 60], pattern profiling-based approaches [35, 36, 37], support distance-based approaches [23, 38], probabilistic model-based method [61] and information theoretic-based summarizations [62, 63]. For summarizing frequent patterns in uncertain databases, We previously proposed the concept of probabilistic representative frequent patterns and developed exact and approximate approaches [40, 50].

4.9 Conclusions

Transaction data is ubiquitous in real world applications. Meanwhile, uncertainty is inherent in the data of various domains. Therefore, in this chapter, we study the problem of summarizing transaction data in uncertain databases and formulate the problem as Minimal Probabilistic Tile Cover Mining. We propose the concept of Probabilistic Price Order and a two-step framework as a solution, including candidates generation and tile construction. We discuss the selection of candidates and devise an algorithm to determine the Probabilistic Price Order of two tiles. Several optimization techniques are further designed to improve the performance. Experiments on both synthetic and real world datasets demonstrate the effectiveness of summarizing uncertain databases using minimal probabilistic tile cover and our proposed MPTC mining approach. In the future, we are interested in applying the proposed algorithm to uncertain database visualization.

Chapter 5

RCP Mining: Towards the Summarization of Spatial Co-location Patterns

Co-location pattern mining is an important task in spatial data mining. It finds patterns of spatial features whose instances tend to locate together in geographic space. However, the traditional framework of co-location pattern mining produces an exponential number of patterns because of the downward closure property, which makes it hard for users to understand, assess or apply the huge number of resulted patterns. To address this issue, in this chapter, we study the problem of mining *representative co-location patterns* (RCP). We first define a covering relationship between two co-location patterns by finding a new measure to appropriately quantify the distance between patterns in terms of their prevalence, based on which the problem of representative co-location pattern mining is formally formulated. To solve the problem of RCP mining, we first propose an algorithm called *RCPFast*, adopting the *post-mining* framework that is commonly used by existing distance-based pattern summarization techniques. To address the peculiar challenge in spatial data mining, we further propose another algorithm, *RCPMS*, which employs the *mine-and-summarize* framework that pushes pattern summarization into the co-location mining process. Optimization strategies are also designed to further improve the performance of *RCPMS*. Our experimental results on both synthetic and real-world data sets demonstrate that RCP mining effectively summarizes spatial co-location patterns, and *RCPMS* is more efficient than *RCPFast*, especially on dense data sets.

5.1 Introduction

As one of the most fundamental tasks in spatial data mining, *co-location mining* aims to discover co-location patterns where each is a group of spatial features whose instances are frequently located close to each other [13]. Spatial co-location patterns yield important insights for various applications. In epidemiology, for example, different incidents of diseases may exhibit co-location patterns such that one type of disease tends to occur in spatial proximity of another [14]. In ecology, scientists are interested in finding frequent co-occurrences among spatial features, such as drought, substantial increase/drop in vegetation, and extremely high precipitation [15]. In e-commerce, companies may be interested in discovering types of services (e.g., weather, timetabling and ticketing queries) that are requested by geographically neighboring users, so that location-sensitive recommendations can be provided [16]. Due to its importance, the problem of finding prevalent co-location patterns from spatial data has been explored extensively [13, 17, 18, 14, 19, 20, 21].

A common framework of co-location pattern mining uses the frequencies of a set of spatial features participating in a co-location to measure the prevalence (known as *participation index* [13], or PI for short) and requires a user-specified minimum threshold to find interesting patterns. Typically, if the threshold is high, the framework may generate commonsense patterns. However, with a low threshold, a great number of patterns will be found. This is further exacerbated by the downward closure property that holds for the *PI* measure. That is, if a set of features is prevalent with respect to a threshold of *PI*, then all of its subsets will be discovered as prevalent co-location patterns. A huge pattern number will jeopardize the usability of resulted patterns, as it demands great efforts to understand or examine the discovered knowledge.

The key idea of solving this problem is to find an effective way to summarize the co-location patterns, e.g., to find a high-quality representation that describes the complete set of resulted patterns precisely and concisely. Two types of compressed co-location patterns have been explored in the literature: *maximal co-location patterns* (MCP) [64] and *closed co-location patterns* (CCP) [65]. A co-location pattern is a MCP if it is prevalent itself and none of its super-patterns are prevalent. MCP mining may significantly reduce the number of co-location patterns, but it fails to preserve the prevalence information. It is therefore a lossy approximation. As for the second type, a co-location pattern is a CCP if it is prevalent itself and none of its super-patterns have the same PI as it does. CCP mining not only diminishes the number of co-location patterns but also preserves the complete PI information. However, by emphasizing too much on the PI information, the compression power of CCP mining is limited.

For example, given a spatial data set shown in Figure 5.1, where instances/events of four spatial features, A, B, C and D, are represented by different symbols and edges connecting events denote spatial neighborhood relationships, Table 5.1 lists a set of five prevalent co-location patterns and their corresponding PI in the data set (the definition of PI is provided in Section 5.2). If MCP mining is adopted, only F_3 will be output as the others are all sub-patterns of F_3. However, F_3 is significantly different from others in terms of their PIs. In contrast, if CCP mining is used, then all patterns will be returned since each of them is a closed pattern. That is, CCP mining provides no compression on this set of patterns. Therefore, to address the limitations of MCP and CCP mining, a method that not only provides optimal compression rate but also preserves reasonable prevalence information will be favored.

Similar idea has been explored in the studies of summarizing frequent itemsets [23, 38, 35]. Xin et al. [23] proposed the notion of an *ε-cover relationship* between itemsets. An itemset X_1 is ε-covered by another itemset X_2 if X_1 is a subset of X_2 and $1 - \frac{|T(X_1) \cap T(X_2)|}{|T(X_1) \cup T(X_2)|} \leq \varepsilon$, where $T(X_i)$ is the set of supporting transactions of pattern X_i. The goal is then to find a minimum set of representative itemsets that can ε-cover all frequent itemsets. In this chapter, we follow their idea and propose to summarize co-location patterns using a set of *representative co-location patterns* (RCPs), which strikes

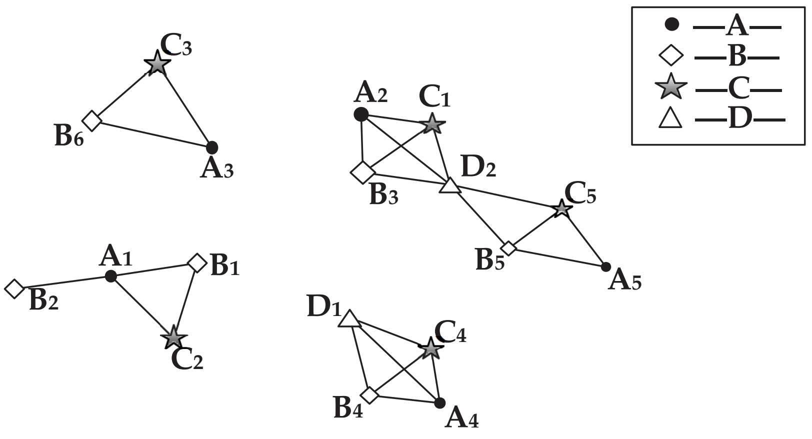


Figure 5.1: A motivating example of a spatial data set. Each symbol represents an event corresponding to a spatial feature, and each edge connecting two events represents a neighborhood relationship.

a fine balance between improving compression rate and preserving prevalence information.

However, existing methods for representative itemsets mining cannot be applied directly to representative co-location pattern mining, neither the framework of problem definition nor the mining process. This is mainly because there is no natural notion of *transactions* in co-location mining [13]. Consequently, the original definition of the ε-cover relationship cannot be adopted straightforwardly because it is defined on a supporting transaction-based distance measure. Moreover, the mining process will be more complicated as it is more expensive to examine whether a set of feature instances participate in a co-location than checking whether a set of items appear in one transaction.

To formulate the problem of representative co-location pattern mining, we first define a new measure to appropriately quantify the distance between two co-location patterns in terms of their prevalence, based on which the ε-cover relationship can be stated on a pair of co-location patterns. To solve the problem of RCP mining, we first propose an algorithm, *RCPFast*, which follows existing distance-based pattern summarization techniques to adopt the *post-mining* framework that finds RCPs from the set of discovered co-location patterns. Observing a peculiar challenge in spatial data mining,

ID	Feature Sets	Events	PI
F_1	$\{A, B\}$	A_1B_1, A_1B_2, A_2B_3 A_4B_4, A_5B_5, A_3B_6	1
F_2	$\{A, B, C\}$	$A_1B_1C_2, A_2B_3C_1$ $A_3B_6C_3, A_4B_4C_4, A_5B_5C_5$	5/6
F_3	$\{A, B, C, D\}$	$A_2B_3C_1D_2, A_4B_4C_4D_1$	1/3
F_4	$\{B, C, D\}$	$B_3C_1D_2, B_4C_4D_1, B_5C_5D_2$	1/2
F_5	$\{C, D\}$	C_1D_2, C_4D_1, C_5D_2	3/5

Table 5.1: A set of prevalent co-location patterns.

we then develop another algorithm, called *RCPMS*, which employs a *mine-and-summarize* framework to discover RCPs directly from the spatial data. To our knowledge, *RCPMS* is the first work among existing distance-based pattern summarization that pushes summarization into the pattern mining process. Optimization strategies are also devised to further improve the efficiency of *RCPMS*. The main contributions of our research are summarized as follows.

- We formally define the problem of representative co-location pattern mining based on a newly exploited measure to quantify the prevalence proximity between two co-location patterns. To our knowledge, this is the first work that summarizes spatial co-location patterns using distance-based representative patterns.
- We develop two algorithms to discover the set of RCPs, *RCPFast* and *RCPMS*, which adopt fundamentally different mining paradigms and exploit different optimization strategies to improve performance.
- We evaluate the performance of the developed algorithms on both synthetic and real-world data sets. Our experimental results demonstrate the effectiveness of RCP mining, and the efficiency of *RCPMS* compared with *RCPFast*, especially on dense data sets.

The remainder of this chapter is organized as follows. In Section 5.2, we define relevant concepts and formally formulate the problem. Section 5.3 introduces the *RCPFast* algorithm. Section 5.4 describes the *RCPMS* algorithm and optimization strategies. In Section 5.5, we evaluate the performance of the developed algorithms. Existing works related to our research are reviewed in Section 5.6. Section 5.7 closes this chapter with some conclusive remarks.

5.2 Preliminary

In this section we first review definitions related to traditional co-location patterns. Then, we introduce a distance metric to measure the prevalence difference between two patterns. Finally, we formally define the problem of representative co-location pattern mining.

5.2.1 Co-location Patterns

Given a set of spatial features $\mathcal{F} = \{f_1, f_2, \ldots, f_K\}$, a spatial data set is a collection of instances/events $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$, where each $e_i \in \mathcal{E}$ is represented by a vector $\langle$*event_id, spatial_feature_type, location*$\rangle$. We review the measures used to characterize the interestingness of a subset of features $F \subseteq \mathcal{F}$ as follows. Please refer to [13] for the details.

Definition 5.1. *Given a subset of features $F = \{f_1, \ldots, f_k\} \subseteq \mathcal{F}$, $E = \{e_1, \ldots, e_k\} \subseteq \mathcal{E}$ is a **Row Instance (RI)** of F, denoted as $RI(F)$, if $\forall i \in [1, k]$, e_i is an instance of f_i and $\forall i, j \in [1, k]$, $||e_i - e_j|| \leq \tau$, where $||e_i - e_j||$ refers to the spatial distance between two events and τ is a user-specified spatial distance threshold.*

Definition 5.2. *Given a spatial data set $\mathcal{E}$ of a set of spatial features $\mathcal{F}$, the **Table Instance (TI)** of a subset of features $F \subseteq \mathcal{F}$, denoted as $TI(F)$, is the collection of all its row instances in $\mathcal{E}$. That is, $TI(F) = \{RI_1(F), \ldots, RI_m(F)\}$.*

For example, consider the spatial data set in Figure 5.1 and $F_5 = \{C, D\}$ in Table 5.1. $\{C_1D_2\}$ is a RI of F_5. $TI(F_5) = \{C_1D_2, C_4D_1, C_5D_2\}$.

Definition 5.3. *Given a subset of features $F = \{f_1, \ldots, f_k\}$, the **Participation Ratio** of a feature $f_i \in F$, denoted as $PR(f_i, F)$, is the fraction of events of feature f_i that participate in the table instance of F. That is,*

$$PR(f_i, F) = \frac{|\{e_j | e_j \in TI(\{f_i\}), e_j \in \widehat{TI}(F)\}|}{|TI(\{f_i\})|}, \tag{5.1}$$

where $\widehat{TI}(\cdot)$ is the union of elements in TI set. Hence, the denominator refers to the total number of events of feature f_i and the numerator refers to the number of distinct events of feature f_i that appear in the table instance of F.

Definition 5.4. *The* ***Participation Index*** *of a subset of features $F = \{f_1, \ldots, f_k\}$, denoted as $PI(F)$, is defined as*

$$PI(F) = \min_{i \in [1,k]} PR(f_i, F). \tag{5.2}$$

For example, consider the spatial data set in Figure 5.1 and $F_2 = \{A, B, C\}$ in Table 5.1. Since $PR(A, F_2) = 5/5$, $PR(B, F_2) = 5/6$, $PR(C, F_2) = 5/5$, we have $PI(F_2) = \min(5/5, 5/6, 5/5) = 5/6$.

Definition 5.5. *Given a user-specified threshold minpi, a subset of features $F \subseteq \mathcal{F}$ is a* ***Prevalent Co-location Pattern (PCP)*** *if $PI(F) \geq minpi$.*

5.2.2 Co-location Distance Measure

A distance measure between traditional frequent itemsets has been proposed in [23]. It compares the supporting transactions of two itemsets and deduces a numerical value as follows,

$$D(I_1, I_2) = 1 - \frac{|T(I_1) \cap T(I_2)|}{|T(I_1) \cup T(I_2)|} \tag{5.3}$$

where $T(I_i)$ denotes the set of transactions supporting the itemset I_i. However, it is difficult to apply this measure to co-location patterns because there is no natural notion of transactions in co-location mining [13]. One possible solution is to *transactionize* the spatial data to let every maximal clique instance [66] be one transaction. For example, we can derive transactions from the data set in Figure 5.1 as: $t_1 = \{A_3B_6C_3\}$, $t_2 = \{A_1B_2\}$, $t_3 = \{A_1B_1C_2\}$, $t_4 = \{A_2B_3C_1D_2\}$, $t_5 = \{A_5B_5C_5\}$, $t_6 = \{B_5C_5D_2\}$, $t_7 = \{A_4B_4C_4D_1\}$. Then, the supporting transactions of a co-location pattern are the set of the corresponding maximal clique instances. For instance, let $F = \{ABC\}$ be a co-location pattern. $T(F) = \{t_1, t_3, t_4, t_5, t_7\}$. With this manipulation, existing supporting transaction-based distance measure can be applied to co-location patterns directly.

However, one critical problem of this solution is that it needs to find all maximal clique instances first. Maximal clique enumeration is a long-standing problem in graph theory and it is known to be NP-hard. Although many efficient algorithms have been proposed to tackle this problem, such as [67, 68], the complexity is still high when the graph is large and dense.

We thus explore a new distance measure that appropriately quantifies the prevalence difference between two co-location patterns which can be computed efficiently without manipulating the spatial data set.

For simplicity, we denote the set in the numerator of Equation (5.1) as $E_F(f_i)$ (i.e., $E_F(f_i) = \{e_j | e_j \in TI(\{f_i\}), e_j \in \widehat{TI}(F)\}$). It refers to the set of events of feature f_i that participate in the table instance of F.

Definition 5.6. *Let F_1 and F_2 be two co-location patterns and f be a feature shared by them, namely, $f \in F_1 \cap F_2$, the* ***Feature Distance*** *between F_1 and F_2 w.r.t. f is defined as*

$$FD_f(F_1, F_2) = 1 - \frac{|E_{F_1}(f) \cap E_{F_2}(f)|}{|E_{F_1}(f) \cup E_{F_2}(f)|} \tag{5.4}$$

Particularly, if $F_1 \subseteq F_2$, the formula can be rewritten as

$$FD_f(F_1, F_2) = 1 - \frac{|E_{F_2}(f)|}{|E_{F_1}(f)|} \tag{5.5}$$

Definition 5.7. *Given two co-location patterns F_1 and F_2, the* ***Co-location Distance*** *between them is defined as*

$$D(F_1, F2) = \begin{cases} \max\limits_{\forall f \in F_1 \cap F_2} FD_f(F_1, F_2), & \textit{if } F_1 \cap F_2 \neq \emptyset \\ 1, & \textit{otherwise} \end{cases} \tag{5.6}$$

Let us apply this new distance measure to co-location patterns in Table 5.1 to see if it reasonably reflects the distance/proximity between patterns in terms of their prevalence. Firstly, we consider $F_1 = \{A, B\}$ and $F_2 = \{A, B, C\}$. According to the above definitions, $E_{F_1}(A) = E_{F_2}(A) = \{A_1, A_2, \ldots, A_5\}$, $E_{F_1}(B) = \{B_1, B_2, \ldots, B_6\}$, $E_{F_2}(B) = \{B_1, B_3, B_4, B_5, B_6\}$, then $FD_A(F_1, F_2) = 1 - \frac{|E_{F_2}(A)|}{|E_{F_1}(A)|} = 1 - \frac{5}{5} = 0$, $FD_B(F_1, F_2) = 1 - \frac{5}{6} = \frac{1}{6}$. Hence, $D(F_1, F_2) = \max(0, \frac{1}{6}) = \frac{1}{6}$. This small distance value suggests that F_1 and F_2 are quite similar in terms of prevalence. Similarly, let us consider $F_2 = \{A, B, C\}$ and $F_3 = \{A, B, C, D\}$. We can have $D(F_2, F_3) = \max(1 - \frac{2}{5}, 1 - \frac{2}{5}, 1 - \frac{2}{5}) = \frac{3}{5}$, which indicates that the two patterns (F_2, F_3) are quite different. We observe that the new distance measure captures the prevalence distance between co-location patterns appropriately.

5.2.3 Problem Statement

Based on the proposed distance measure, we define the ε-cover relationship between two co-location patterns as follows.

Definition 5.8. *Given two co-location patterns F_1 and F_2, and a real number $\varepsilon \in [0, 1]$, we say F_2 ε-covers F_1 if (1) $F_1 \subseteq F_2$ and (2) $D(F_1, F_2) \leq \varepsilon$.*

Then, given a set of prevalent co-location patterns, we can group them into *ε-clusters*, where each ε-cluster consists of a centroid pattern F_r that ε-covers all patterns in the cluster. It seems that we may return centroid patterns of ε-clusters as representative patterns. However, by doing so, we restrict the representative patterns to be prevalent themselves (i.e. $PI(F_r) \geq minpi$). The minimum number of representative co-location patterns that can be achieved using this method is the number of MCPs.

In [23], it shows that an itemset only needs to satisfy a relaxed condition (i.e., $\text{Supp}(X) \geq (1 - \varepsilon) \cdot minsup$) to ε-cover a frequent itemset. We find that this property holds as well for our newly defined distance measure and the induced ε-cover relationship.

Let us consider two co-location patterns F_1 and F_2, where $F_1 \subset F_2$ and F_1 is prevalent, $PI(F_1) \geq minpi$. According to the definition of PI, we have

$$PI(F_1) = \min_{\forall f \in F_1} \frac{|E_{F_1}(f)|}{|TI(\{f\})|} \geq minpi \tag{5.7}$$

If F_2 is able to ε-cover F_1, then $D(F_1, F_2) \leq \varepsilon$. That is,

$$\max_{\forall f \in F_1} \left(1 - \frac{|E_{F_2}(f)|}{|E_{F_1}(f)|}\right) \leq \varepsilon \tag{5.8}$$

From the above two equations, we have

$$\begin{aligned}\forall f \in F_1, |E_{F_2}(f)| &\geq (1-\varepsilon) \cdot |E_{F_1}(f)| \\ &\geq (1-\varepsilon) \cdot minpi \cdot |TI(\{f\})| \end{aligned} \tag{5.9}$$

Hence,

$$\forall f \in F_1, \frac{|E_{F_2}(f)|}{|TI(\{f\})|} \geq (1-\varepsilon) \cdot minpi \tag{5.10}$$

Recall that, the PI of F_2 can be computed as follows,

$$PI(F_2) = \min_{\forall f \in F_1, f' \in F_2 \setminus F_1} \left(\frac{|E_{F_2}(f)|}{|TI(\{f\})|}, \frac{|E_{F_2}(f')|}{|TI(\{f'\})|} \right) \tag{5.11}$$

Thus, as long as we require $PI(F_2) \geq (1-\varepsilon) \cdot minpi$, the condition in Eq. (5.10) can be satisfied, no matter $\frac{|E_{F_2}(f)|}{|TI(\{f\})|}$ is greater than $\frac{|E_{F_2}(f')|}{|TI(\{f'\})|}$ or the other way around. That is, to ε-cover a prevalent co-location pattern F_1, F_2 only needs to be prevalent with respect to a lower threshold $minpi^* = (1-\varepsilon) \cdot minpi$. Our experimental results in Section 5.5 show that this relaxation contributes to an improved compression rate.

Definition 5.9. ***(Problem Statement)*** *Given a set of spatial features $\mathcal{F}$, a spatial data set $\mathcal{E}$ on $\mathcal{F}$, a spatial distance threshold τ, a co-location distance threshold ε, and a prevalence threshold $minpi$, the problem of representative co-location pattern (RCP) mining is to discover a minimal set of co-location patterns $\mathcal{R}$ such that: (1) For all $F_r \in \mathcal{R}, PI(F_r) \geq (1-\varepsilon) \cdot minpi$; (2) For any prevalent co-location patterns F, i.e., $PI(F) \geq minpi$, there exits a $F_r \in \mathcal{R}$ s.t. F_r ε-covers F.*

5.3 The *RCPFast* Algorithm

In this section, we first introduce an algorithm, *RCPFast*, which follows existing distance-based pattern summarization approaches to mine RCPs by adopting a *post-mining* framework.

Similar to [23], the mining framework of *RCPFast* consists of three stages. Stage 1 discovers two sets of prevalent co-location patterns, *PCP* and *PCP**, with respect to *minpi* and $(1-\varepsilon) \cdot minpi$, respectively. The objective is then to select minimal number of patterns from PCP^* to cover all patterns in PCP.

Stage 2 generates the complete coverage information by finding all prevalent co-location patterns $F \in PCP$ that can be ε-covered by each pattern $F_r \in PCP^*$. All prevalent co-location patterns ε-covered by F_r is stored in $set(F_r)$.

Stage 3 finds the set of desired RCPs based on the coverage information. As discussed in [23], this is a set cover problem which is NP-hard. It can be solved by a greedy strategy that always selects the representative pattern that covers the most number of prevalent co-location patterns. According to [69], the relation between the number of RCPs selected by the greedy solution and the number of the optimal ones is bounded by Theorem 5.1.

Theorem 5.1. *Given a set of prevalent co-location patterns PCP w.r.t. minpi, a set of prevalent co-location patterns PCP^* w.r.t. $(1-\varepsilon) \cdot minpi$, let the number of RCPs generated using the greedy set cover algorithm be $\mathcal{C}_g$, and the number of optimal RCPs be $\mathcal{C}^*$, then $|\mathcal{C}_g| \leq |\mathcal{C}^*| \times H(\max_{F_r \in PCP^*} |set(F_r)|)$, where $H(n) = \sum_{k=1}^{n} \frac{1}{k}$.*

Since the time complexity of the greedy algorithm is $O(\sum_{F_r \in PCP^*} |set(F_r)|)$, the computational cost of *RCPFast* mainly comes from the first two stages.

For the first stage, mining prevalent co-locations is a well-studied topic. Many efficient algorithms have been proposed, e.g., the spatial-join method [70] and the join-less method [14]. Note that it is unnecessary to run the mining process twice to discover the two sets of PCP and PCP^*. We can find prevalent patterns w.r.t. $(1-\varepsilon) \cdot minpi$ first, and then filter the results to obtain those prevalent w.r.t. $minpi$.

For the second stage, the bottleneck lies in the computations of co-location distance between two patterns to verify the ε-cover relationship. The complexity of generating the complete coverage information is $O(|PCP| \cdot |PCP^*|)$, which will become a performance issue when there are many prevalent patterns. Therefore, we aim to exploit strategies to skip verifying the ε-cover relationship for as many pairs of patterns as possible.

Theorem 5.2. *Given three co-location patterns F_1, F_2, and F_3 s.t. $F_1 \subseteq F_2 \subseteq F_3$, if F_3 ε-covers F_1, then F_2 ε-covers F_1.*

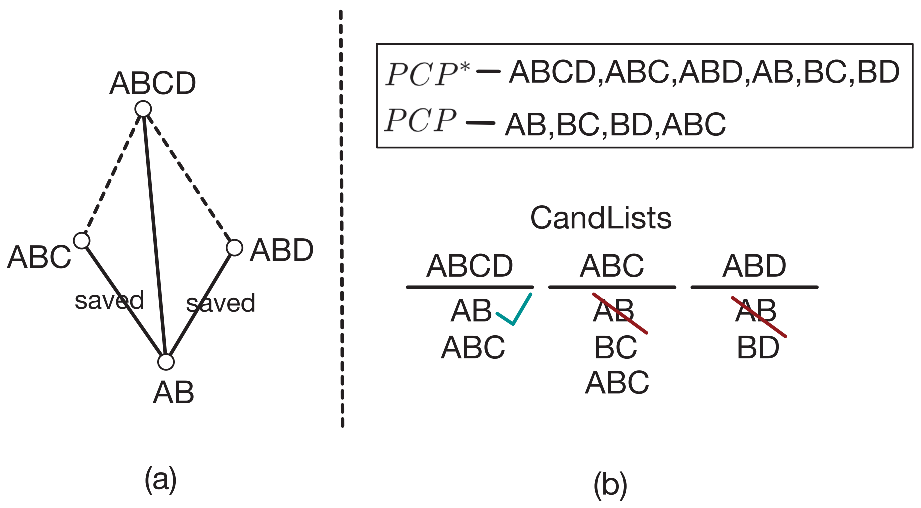


Figure 5.2: An example illustrating *RCPFast* algorithm.

Proof. From $D(F_1, F_3) \leq \varepsilon$, we have $\forall f \in F_1, 1 - \frac{|E_{F_3}(f)|}{|E_{F_1}(f)|} \leq \varepsilon$. Because $\forall f \in F_1$, $|E_{F_2}(f)| \geq |E_{F_3}(f)|$. Thus, we have $1 - \frac{|E_{F_2}(f)|}{|E_{F_1}(f)|} \leq 1 - \frac{|E_{F_3}(f)|}{|E_{F_1}(f)|} \leq \varepsilon$, or $D(F_1, F_2) \leq \varepsilon$, which proves the result. □

According to Theorem 5.2, we are allowed to skip computing co-location distance for certain pairs of co-location patterns. For example, as shown in Figure 5.2 (a), if we have found that $\{A, B, C, D\}$ ε-covers $\{A, B\}$, then we conclude immediately that $\{A, B, C\}$ ε-covers $\{A, B\}$ and $\{A, B, D\}$ ε-covers $\{A, B\}$ without computing their corresponding co-location distances. To maximize the benefit introduced by Theorem 5.2, we order the co-location patterns according to pattern lengths. Then, the procedure of *RCPFast* is illustrated in Algorithm 5.1.

Algorithm 5.1 follows the three-stage framework. The first stage (lines 1–2) mines two sets of prevalent co-location patterns and the third stage (lines 15–19) discovers the RCPs using a greedy strategy. The second stage starts with sorting the patterns in PCP^* in decreasing order of pattern length, and sorting patterns in PCP in the reverse order (line 3). Then, a candidate list (*CandList*) is constructed for each representative pattern F_r in PCP^*, which stores all prevalent patterns that may be ε-covered by F_r (lines 4–7). Lines 8–14 find the complete coverage information for each pattern F_r in

Algorithm 5.1 *RCPFast*

Input: (1) A set of spatial events $\mathcal{E}$, (2) a spatial distance threshold τ, (3) a prevalence threshold $minpi$, (4) a co-location distance threshold ε.
Output: The set of RCPs $\mathcal{R}$

```
1:  $PCP = \text{MinePCP}(\mathcal{E}, \tau, minpi)$
2:  $PCP^* = \text{MinePCP}(\mathcal{E}, \tau, (1-\varepsilon) \cdot minpi)$
3:  Sort $PCP^*$ in decreasing order of pattern length, and $PCP$ in increasing
    order of pattern length.
4:  for all $F_r \in PCP^*$ do
5:    for all $F \in PCP$ do
6:      if $F \subseteq F_r$ then
7:        Insert $F$ into $CandList(F_r)$
8:  for all $F_r \in PCP^*$ do
9:    for all $F \in CandList(F_r)$ do
10:     if $F_r$ $\varepsilon$-covers $F$ then
11:       Insert $F$ into $set(F_r)$
12:       Find a set of patterns $\mathcal{Q} \subseteq PCP^*$ s.t. $\forall Q \in \mathcal{Q}$, $F \subseteq Q \subseteq F_r$
13:       for all $Q \in \mathcal{Q}$ do
14:         Remove $F$ from $CandList(Q)$ to $set(Q)$
15: while $PCP \neq \emptyset$ do
16:   Find a $F_r$ that maximizes $|set(\mathcal{R})|$
17:   for all $F \in set(F_r)$ do
18:     Delete $F$ from $PCP$
19:   $\mathcal{R} = \mathcal{R} \cup \{F_r\}$
20: return $\mathcal{R}$
```

PCP^* by implementing the optimization enabled by Theorem 5.2. In particular, once it is confirmed that F_r ε-covers a prevalent pattern F (line 10), we find a set of patterns $\mathcal{Q} \subseteq PCP^*$ where each $Q \in \mathcal{Q}$ is a sub-pattern of the current F_r and a super-pattern of F (line 12). According to Theorem 5.2, F can be immediately added to $set(Q)$ (line 14).

Note that, the purpose of sorting PCP and PCP^* in the specified orders is to allow early discovery of ε-cover relationship between a representative pattern and its short sub-patterns so that more pairs of patterns can be skipped for co-location distance computation. For example, Figure 5.2 (b) shows the candidate lists of three representative patterns, $ABCD$, ABC and ABD. Due to the ordering of patterns, we examine first whether $ABCD$ ε-covers AB. If it happens, we can delete AB from $CandList(ABC)$ and

$CandList(ABD)$ because AB should be covered by these two patterns according to Theorem 5.2. Therefore, the computations of $D(ABC, AB)$ and $D(ABD, AB)$ are omitted.

5.4 The *RCPMS* Algorithm

Recall that, to verify the ε-cover relationship in the second stage of *RCP-Fast*, we need to compute the co-location distance between two patterns, which requires the *table instance* information of the corresponding patterns. However, the output of prevalent co-location pattern mining in the first stage contains only the prevalent patterns as well as their PI information. It may not be an issue for frequent itemset summarization as the supporting transactions of an itemset can be retrieved easily. However, for spatial data mining, it is expensive to re-scan the data to obtain the table instance of a co-location pattern whenever it is required. One possible solution is to output the information of table instances as additional results. However, if the information is stored in disk, extra I/O cost will be incurred. If the information is stored in memory, it will become problematic when the number patterns is huge. Therefore, we are motivated to push coverage validation into the co-location mining process, thereby integrating the first and the second stages in order to address the table instance acquisition problem.

Based on the idea, we devise the *RCPMS* algorithm that employs a novel *mine-and-summarize* framework, while all existing distance-based pattern summarization techniques adopt the *post-mining* paradigm. More specifically, whenever a representative pattern, prevalent w.r.t. $(1 - \varepsilon) \cdot minpi$, is discovered, all prevalent patterns, w.r.t. $minpi$, which can be ε-covered by it will be found. The feasibility of this idea is supported by the following two facts.

1. Traditional prevalent co-location pattern mining algorithms usually use an Apriori-based level-wise scheme to generate patterns [70, 14]. When a representative pattern is mined, all its prevalent sub-patterns have already been found. Hence, it is sufficient to find the coverage information for the current representative pattern.

Algorithm 5.2 *RCPMS*

```
Input: Same as RCPFast.
Output: Same as RCPFast.
1: $P_1 = \mathcal{F}, k = 2$
2: while $P_{k-1} \neq \emptyset$ do
3:     $C_k$ = gen_candidate_colo($P_{k-1}$)
4:     for all $C \in C_k$ do
5:         $pi$ = calculate_PI($C$)
6:         if $pi \geq (1-\varepsilon) \cdot minpi$ then
7:             $D\_Table \leftarrow$ cal_preval_child_dis(C)
8:             $set(C)$ = gen_cover_set(C, C, 0)
9:             if $pi \geq minpi$ then
10:                Insert C into $P_k$ and $set(C)$
11:    $k = k + 1$
12: Obtain RCPs using the greedy algorithm
```

2. When a representative pattern is output in the mining process, its information of table instance is available, which can be used to compute its co-location distances with its sub-patterns. For its sub-patterns, we store their table instance information in memory if they are child or immediate sub-patterns of the current representative pattern (e.g., $F \subset F_r$ and $|F_r| - |F| = 1$). Otherwise, we will retrieve the table instance information of a sub-pattern F (e.g. $F \subset F_r$ and $|F_r| - |F| > 1$) only if the ε-cover relationship between F_r and F cannot be inferred using our devised optimization and approximation strategies, which will be discussed in Sections 5.4.1 and 5.4.2.

The general idea of *RCPMS* is summarized in Algorithm 5.2. In the beginning, it assigns all unique spatial features to P_1 (line 1). From line 2 to line 11, an iterative process is used to generate patterns of length k from patterns of length $k-1$. In particular, line 3 calls the function *gen_candidate_colo* to generate candidate co-location patterns (e.g., using an Apriori-like strategy). For each candidate co-location pattern, we first calculate its PI (line 5). If the candidate pattern is prevalent w.r.t. $(1-\varepsilon) \cdot minpi$ (line 6), we compute its co-location distances with its child sub-patterns which are prevalent w.r.t. *minpi* and store it in a distance table (line 7). Note that, only the co-location distances between the current pattern and its prevalent child sub-patterns need to be computed at this stage. As discussed later, its co-location distances with other descendent prevalent sub-patterns will be computed only if they cannot be inferred using our proposed optimization and approximation strategies. In line 8, we call the method *gen_cover_set* to find all prevalent sub-patterns that can be covered by the current representative pattern. Finally, if the current pattern is prevalent w.r.t. *minpi*, it should be used to generate candidate patterns in the next round and should be included into its own cover set (lines 9 and 10). Line 12 is the same as the third stage of *RCPFast* which finds the minimal RCPs.

In the following, we describe the details of the function *gen_cover_set* which finds all prevalent sub-patterns that can be ε-covered by the current representative pattern. Before presenting the function, we first introduce an optimization strategy and an approximation strategy that are used by the function.

5.4.1 Optimization Strategy

Note that, the optimization strategy used by *RCPFast* (i.e., Theorem 5.2) is not applicable here. This is because when we output a representative pattern of length k in *RCPMS*, the coverage information of representative patterns of length $(k-1)$ has already been found. Therefore, we exploit a new optimization strategy based on the following theorem:

Theorem 5.3. *Given three co-location patterns F_1, F_2 and F_3 s.t. $F_1 \subseteq F_2 \subseteq F_3$, $D(F_1, F_2) + D(F_2, F_3) \geq D(F_1, F_3)$.*

We introduce an auxiliary lemma before prove this theorem.

Lemma 5.1. *Let $S_i = (n_{ia}, n_{ib}, n_{ic})$, $1 \leq i \leq m$ be tuples such that each contains three non-negative integers which satisfy $n_{ia} \geq n_{ib} \geq n_{ic}$, then the following inequality holds:*

$$\max_{1\leq i\leq m}\left(1-\frac{n_{ib}}{n_{ia}}\right)+\max_{1\leq i\leq m}\left(1-\frac{n_{ic}}{n_{ib}}\right)\geq\max_{1\leq i\leq m}\left(1-\frac{n_{ic}}{n_{ia}}\right) \tag{5.12}$$

Proof. For simplicity, let $D_1(i) = \left(1-\frac{n_{ib}}{n_{ia}}\right)$, $D_2(i) = \left(1-\frac{n_{ic}}{n_{ib}}\right)$, $D_3(i) = \left(1-\frac{n_{ic}}{n_{ia}}\right)$. The proof can be completed in two stages.

1. Prove $\forall i$, $D_1(i) + D_2(i) \geq D_3(i)$. This can be done by verifying
$$D_1(i) + D_2(i) - D_3(i) = \frac{(n_{ia} - n_{ib})(n_{ib} - n_{ic})}{n_{ia}n_{ib}} \geq 0$$
2. Suppose $\exists\ \tilde{l_1},\ \tilde{l_2},\ \tilde{l_3}$ such that $D_1(\tilde{l_1}) = \max_{1\leq i\leq m} D_1(i)$, $D_2(\tilde{l_2}) = \max_{1\leq i\leq m} D_2(i)$, $D_3(\tilde{l_3}) = \max_{1\leq i\leq m} D_3(i)$, because $D_1(\tilde{l_1}) \geq D_1(\tilde{l_3})$ and $D_2(\tilde{l_2}) \geq D_2(\tilde{l_3})$, we have
$$D_1(\tilde{l_1}) + D_2(\tilde{l_2}) - D_3(\tilde{l_3}) \geq D_1(\tilde{l_3}) + D_2(\tilde{l_3}) - D_3(\tilde{l_3}) \geq 0.$$

Therefore, Lemma 5.1 is proved. □

Next, we present the proof of Theorem 5.3.

Proof. First of all, because $F_1 \cap F_2 = F_1$, $F_1 \cap F_3 = F_1$, and $F_2 \cap F_3 = F_2$, according to the definitions of co-location distance, we have

$$D(F_1, F_2) = \max_{\forall f\in F_1}\left(1 - \frac{|E_{F_2}(f)|}{|E_{F_1}(f)|}\right) \tag{5.13}$$
$$D(F_2, F_3) = \max_{\forall f\in F_1, f'\in F_2\backslash F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_2}(f)|}, 1 - \frac{|E_{F_3}(f')|}{|E_{F_2}(f')|}\right) \tag{5.14}$$
$$D(F_1, F_3) = \max_{\forall f\in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_1}(f)|}\right) \tag{5.15}$$

Note that we divide $D(F_2, F_3)$ into two parts, such that the former only involves the features in F_1, and the latter involves the remaining. The value of $D(F_2, F_3)$ depends on the relations between the two parts. It can be divided into two situations.

1. $D(F_2, F_3)$ is only related to the former part:
$$\max_{\forall f\in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_2}(f)|}\right) > \max_{\forall f'\in F_2\backslash F_1}\left(1 - \frac{|E_{F_3}(f')|}{|E_{F_2}(f')|}\right). \tag{5.16}$$
In this case, we have
$$D(F_2, F_3) = \max_{\forall f\in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_2}(f)|}\right). \tag{5.17}$$

Because $\forall f \in F_1$, $E_{F_3}(f) \subseteq E_{F_2}(f) \subseteq E_{F_1}(f)$ holds, we have $|E_{F_1}(f)| \geq |E_{F_2}(f)| \geq |E_{F_3}(f)|$. Let each $f \in F_1$ binds to a tuple in Lemma 5.1 by building the corresponding relations: $n_{ia} = |E_{F_1}(f)|, n_{ib} = |E_{F_2}(f)|, n_{ic} = |E_{F_3}(f)|$. Therefore by Lemma 5.1, we have

$$\max_{\forall f \in F_1}\left(1 - \frac{|E_{F_2}(f)|}{|E_{F_1}(f)|}\right) + \max_{\forall f \in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_2}(f)|}\right) \geq \max_{\forall f \in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_1}(f)|}\right),$$

which means $D(F_1, F_2) + D(F_2, F_3) \geq D(F_1, F_3)$.

2. $D(F_2, F_3)$ only depends on the latter part, that is:

$$D(F_2, F_3) = \max_{\forall f' \in F_2 \backslash F_1}\left(1 - \frac{|E_{F_3}(f')|}{|E_{F_2}(f')|}\right) \tag{5.18}$$

and

$$\max_{\forall f' \in F_2 \backslash F_1}\left(1 - \frac{|E_{F_3}(f')|}{|E_{F_2}(f')|}\right) \geq \max_{\forall f \in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_2}(f)|}\right). \tag{5.19}$$

According to the first situation, we have

$$\begin{aligned}
&D(F_1, F_2) + D(F_2, F_3) - D(F_1, F_3) \\
=&\max_{\forall f \in F_1}\left(1 - \frac{|E_{F_2}(f)|}{|E_{F_1}(f)|}\right) + \max_{\forall f' \in F_2 \backslash F_1}\left(1 - \frac{|E_{F_3}(f')|}{|E_{F_2}(f')|}\right) \\
&- \max_{\forall f \in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_1}(f)|}\right) \\
\geq&\max_{\forall f \in F_1}\left(1 - \frac{|E_{F_2}(f)|}{|E_{F_1}(f)|}\right) + \max_{\forall f \in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_2}(f)|}\right) \\
&- \max_{\forall f \in F_1}\left(1 - \frac{|E_{F_3}(f)|}{|E_{F_1}(f)|}\right) \\
\geq&0,
\end{aligned}$$

which proves the result. □

Figure 5.3 illustrates how to use Theorem 5.3 to skip computing co-location distance between a representative pattern and its non-child prevalent sub-patterns. Suppose F_6 is the current representative pattern and we

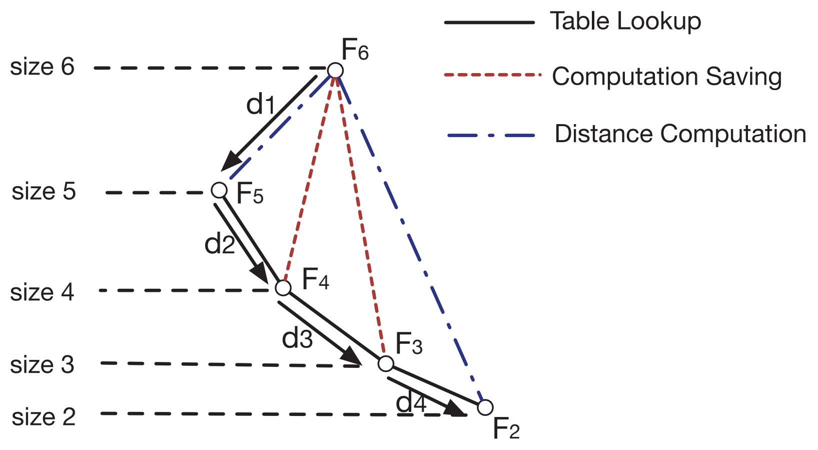


Figure 5.3: An illustration of the optimization strategy based on Theorem 5.3. Each F_i is a co-location pattern of size i. Different types of lines represent different ways of obtaining the co-location distance. Suppose $d_1 + d_2 + d_3 \leq \varepsilon$ and $d_1 + d_2 + d_3 + d_4 > \varepsilon$.

have computed its co-location distance with its child sub-pattern F_5, $D(F_6, F_5) = d_1$, stored in the *D_Table* (e.g., line 7 in Algorithm 5.2). Next, we need to examine whether F_6 ε-covers the child of F_5, e.g., F_4. Note that $D(F_5, F_4) = d_2$ should have been computed and stored in the *D_Table* when outputting F_5 in the previous round. According to Theorem 5.3, we infer that $D(F_6, F_4) < D(F_6, F_5) + D(F_5, F_4) = d_1 + d_2$. Therefore, if $d_1 + d_2 \leq \varepsilon$, we can conclude that F_6 ε-covers F_4 without computing $D(F_6, F_4)$. Similarly, when examining whether F_6 ε-covers F_3, which is a child sub-pattern of F_4, we have $D(F_6, F_3) < D(F_6, F_5) + D(F_5, F_4) + D(F_4, F_3) = d_1 + d_2 + d_3$. As indicated in the figure, $d_1 + d_2 + d_3 \leq \varepsilon$, we conclude that F_6 ε-covers F_3 and skip computing the distance $D(F_6, F_3)$. When it comes to F_2, since $d_1 + d_2 + d_3 + d_4 > \varepsilon$, we have to compute the exact value of $D(F_6, F_2)$ (we will have to re-gain the table instance of F_2 in this case). Therefore, in this particular example, Theorem 5.3 enables us to skip two of the three co-location distance computations (i.e., $D(F_6, F_4)$, $D(F_6, F_3)$ and $D(F_6, F_2)$).

5.4.2 Approximation Strategy

Although Theorem 5.3 can reduce distance computation for a certain number of pairs of patterns, the effectiveness of this single strategy may not be sufficient. Therefore, we further exploit an approximation strategy which substantially improves the computation efficiency by slightly sacrificing the compression rate.

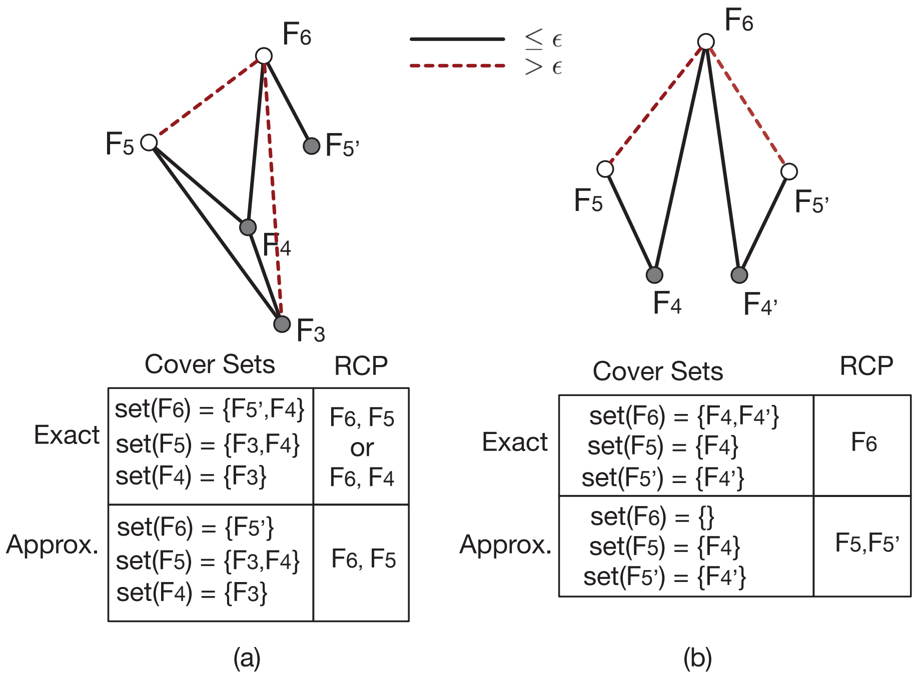
Recall that, in Figure 5.3, only if the co-location distance between the current representative pattern (e.g., F_6) and its child prevalent sub-pattern (e.g., F_5) is smaller than ε, we may use the optimization strategy to infer the distance between F_6 and F_4 (F_3). Otherwise, we have to compute the distance between F_6 and F_4 (F_3), which is expensive since we have to re-gain the table instance of F_4 (F_3). Therefore, we consider the following approximation strategy.

If a representative pattern F_r cannot ε-cover its prevalent child sub-pattern F, we skip considering whether F_r ε-covers any descendant sub-pattern of F.

For example, in Figure 5.3, if the co-location distance between F_6 and F_5 is greater than ε, all F_4, F_3 and F_2 will not be included in $set(F_6)$.

Figure 5.4 provides two examples to illustrate the influence of the approximation strategy. In Figure 5.4 (a), let us assume the set of PCP that need to be summarized are F_5', F_4 and F_3, where F_3 is a child sub-pattern of F_4, which is a child sub-pattern of F_5. F_5' is a sibling pattern of F_5 (e.g., ABC and ABD). The exact coverage information shows that F_6 ε-covers F_4. However, since F_6 does not ε-cover F_5, F_4 is removed from $set(F_6)$ according to the approximation strategy. If using the greedy algorithm to find RCPs, the final number of RCPs found from the exact cover sets will be 2, which is the same as the final number of RCPs found from the approximate cover sets. It indicates that the approximation strategy does not incur any difference to the final number of RCPs under this situation.

In contrast, Figure 5.4 (b) shows an example where this approximation strategy will result in difference in the final number of RCPs. In this example, suppose the set of PCP are F_4 and F_4'. The complete coverage information shows that F_6 ε-covers both F_4 and F_4'. Since F_6 does not ε-cover F_5 or F_5', F_4 and F_4' are removed from $set(F_6)$ in the approximate cover sets. Consequently, the final number of RCPs found from the exact cover sets is 1 while



	Cover Sets	RCP

Exact	set(F_6) = {$F_{5'}$,F_4}
	set(F_5) = {F_3,F_4}
	set(F_4) = {F_3}
		F_6, F_5
		or
		F_6, F_4

Approx.	set(F_6) = {$F_{5'}$}
	set(F_5) = {F_3,F_4}
	set(F_4) = {F_3}
		F_6, F_5

(a)

	Cover Sets	RCP

Exact	set(F_6) = {F_4,$F_{4'}$}
	set(F_5) = {F_4}
	set($F_{5'}$) = {$F_{4'}$}
		F_6

Approx.	set(F_6) = {}
	set(F_5) = {F_4}
	set($F_{5'}$) = {$F_{4'}$}
		F_5,$F_{5'}$

(b)

Figure 5.4: Examples of the approximation strategy.

the final number found from the approximate cover sets is 2.

In general, we have the following lemma, implying that the final number of RCPs generated from the incomplete cover sets, produced by the approximation strategy, will be no smaller than the final number of RCPs generated from the complete cover sets.

Lemma 5.2. *Let $\mathcal{P}$ be a set of representative patterns with non-empty cover sets. That is, $\mathcal{P} \subseteq PCP^*$ and $\forall P \in \mathcal{P}$, $|set(P)| > 0$. Let $\mathcal{P}'$ be a set of representative patterns with non-empty cover sets found using the approximation strategy. Then we have (1) $\mathcal{P}' \subseteq \mathcal{P}$ and $\forall P \in \mathcal{P}'$, $|set(P)| \geq |set'(P)|$, where $set'(P)$ represents the cover set generated by the approximation strategy. (2) let $\mathcal{R}$ and $\mathcal{R}'$ be the minimum sets of RCPs generated from $\mathcal{P}$ and $\mathcal{P}'$, respectively, i.e., $\mathcal{R} \subseteq \mathcal{P}$ and $\mathcal{R}' \subseteq \mathcal{P}'$, we have $|\mathcal{R}| \leq |\mathcal{R}'|$.*

Proof. The first conclusion can be proved easily since the approximation strategy removes prevalent patterns from the cover set of a representative pattern. Therefore, for the same representative pattern P, $|set(P)| \geq |set'(P)|$.

Algorithm 5.3 gen_cover_set(F_r, F, dis)

```
Input: $F_r$: the current representative pattern; $F$: a sub-pattern; $dis$: accumulated distance
Output: $\mathcal{S}$: all prevalent patterns $\varepsilon$-covered by $F_r$
1: for all $P \subset F$ s.t. $|F| - |P| = 1$ & $PI(P) \geq minpi$ do
2:    $dis = dis$ + TableLookup($P$, $F$)
3:    if $dis \leq \varepsilon$ then
4:       Insert $P$ to $\mathcal{S}$
5:       gen_cover_set($F_r$, $P$, $dis$)
6:    else
7:       $dis = D(F_r, P)$
8:       if $dis \leq \varepsilon$ then
9:          Insert $P$ to $\mathcal{S}$
10:         gen_cover_set($F_r$, $P$, $dis$)
11: return $\mathcal{S}$
```

Consequently, if a representative pattern has non-empty approximate cover set (e.g., $|set'(P)| > 0$), its real cover set must be non-empty (e.g., $|set(P)| > 0$). That is, $\forall P \in \mathcal{P}'$, $P \in P$. However, the other way around may not be true. Thus, we have $\mathcal{P}' \subseteq \mathcal{P}$.

To prove the second conclusion, let us assume first $|\mathcal{R}'| < |\mathcal{R}|$. Since $\mathcal{R}' \subseteq \mathcal{P}'$ and $\mathcal{P}' \subseteq \mathcal{P}$, we must be able to find the same result set $\mathcal{R}'$ from $\mathcal{P}$. That is $\mathcal{R}' \subset \mathcal{P}$. In this case $\mathcal{R} = \mathcal{R}'$, which contradicts with $|\mathcal{R}'| < |\mathcal{R}|$. Hence the assumption is wrong. □

We will investigate the efficiency improvement gained by this approximate strategy and the incurred loss of compression rate in Section 5.5.

5.4.3 The *gen_cover_set()* Function

Integrating the optimization strategy and the approximation strategy discussed above, we present the function gen_cover_set() in Algorithm 5.3.

Given the input representative pattern F_r, Algorithm 5.3 visits its sub-patterns using a depth-first search. Line 1 finds all child prevalent co-location patterns of the current pattern F. Lines 2–4 implement the optimization strategy, when the co-location distance between F_r and a sub-pattern can be inferred to be smaller than ε. Otherwise, we have to compute the co-location distance (line 7). If the co-location distance is smaller than ε, we check further descendent sub-patterns (lines 9–10). If not, the depth-first search can be stopped according to the approximation strategy.

5.5 Experimental Study

We have conducted comprehensive experiments to evaluate the proposed algorithms from multiple perspectives on both synthetic and real data sets. All algorithms are implemented in Python 2.7.5. All experiments are run on a PC with Intel Core Xeon 2.9 GHz CPU and 8 GB memory.

5.5.1 Experiments on Synthetic Data

In this section, we first introduce the synthetic data generator used in our experiments. Then we present the evaluation results on three different synthetic data sets.

5.5.1.1 Synthetic Data Generator

Our synthetic data generation methodology is similar to the one used in [70] for co-location mining. Table 5.2 summarizes the parameters used in the generator. The data generation process begins with the generation of a set of N_{seed} seed co-locations. For each seed co-location, we randomly pick 2 features without replacement from a feature set. Next, we generate the instances of seed co-locations. The number of instances of a seed co-location is decided by a Poisson distribution with mean λ. To decide the positions of a seed co-location instances, we randomly pick a location (x_c, y_c) from the whole map $D_1 \times D_2$ as the center and set the radius as $r = \text{uniform}(0, \frac{\tau}{2})$, where $\text{uniform}(0, \frac{\tau}{2})$ selects a value from $(0, \frac{\tau}{2}]$ uniformly. Then we place instances within the circle. The coordinate of an instance is

$$(x_c + r \cdot \cos(\text{uniform}(0, 2\pi)), y_c + r \cdot \sin(\text{uniform}(0, 2\pi))).$$

After generating seed co-location instances, we generate auxiliary co-locations by growing each seed co-location with N_{aux} additional features. Particularly, for each seed co-location with N_c instances and its corresponding circles, we randomly select $\alpha^i \cdot N_c$ circles to insert an instance of the i^{th} additional feature, where $\alpha \in (0, 1]$ is the density ratio. A larger α leads to a denser data set. The final step is to generate noises, based on the two parameters of r_{noise}, the ratio of noise features, and n_{noise}, the number of noise instances per noise feature. Noise instances are placed randomly in the whole map.

Three synthetic data sets are generated with specific parameter values listed in Table 5.2. SynData_1 is a sparse data set while the other two are relatively dense. In particular, SynData_1 contains 37 features and $29,496$ events; SynData_2 consists 52 features and $291,520$ events, with more instances per co-location; and SynData_3 involves 525 features and $424,400$ events.

5.5.1.2 Compression Rate

We first evaluate the compression rate achieved by representative co-location pattern (RCP) mining, in comparison with closed co-location pattern (CCP) mining and maximal co-location pattern (MCP) mining. Specifically, we define *compression rate* as $(1 - \frac{N^*}{N_{PCP}}) \times 100\%$, where N^* equals to the number of compressed patterns and N_{PCP} refers to the number of prevalent co-location patterns (PCP).

Besides comparing with CCP and MCP, we also conduct experiments to investigate the compression rate of RCP without relaxation (RCP-NoRelax). As discussed in Section 5.2.3, we may either generate ε-clusters from the spatial data and return the prevalent centroid patterns as representatives, or relax the restriction to allow representative patterns to be prevalent w.r.t. $(1 - \varepsilon) \cdot minpi$ in order to achieve higher compression rate.

Figure 5.5 shows the compression rates of MCP, CCP, RCP, and RCP-NoRelax on the three synthetic data sets by varying the parameters $minpi$ and ε respectively. Overall, it can be observed that CCP has the lowest compression rate, while RCP achieves a higher compression rate than RCP-NoRelax. Regarding the comparison between RCP and MCP, we observe that MCP has a higher compression rate when ε is fixed at 0.2 (Figures 5.5.a, 5.5.c and 5.5.e). However, as ε is getting larger, the compression rate of RCP prevails (Figures 5.5.b, 5.5.d and 5.5.f). That is, by relaxing the condition on the co-location distance threshold ε, RCP can achieve a compression rate which is even higher than that of MCP. This is due to the definition of RCP, while the best compression rate of RCP-NoRelax is bounded by that of MCP.

Moreover, it can be observed that RCP obtains a high compression rate on a dense data set. For example, when $\varepsilon = 0.2$, the best compression rate of RCP on SynData_1 is 71.9% (Figure 5.5.a) while it is 89.9% and 85.0% on SynData_2 and SynData_3, respectively (Figure 5.5.c and Figure 5.5.e). This is because a representative pattern tends to cover more patterns on a dense data set. We also observe that when the prevalent threshold *minpi* gets smaller, which means more co-location patterns are generated, the compression rate of RCP is higher. When the requirement on preserving the prevalence information is relaxed (i.e., when the co-location distance threshold ε is increased), the compression rate of RCP also improves, which is consistent with the definition of ε-cover relationship.

5.5.1.3 *RCPFast* vs. *RCPMS*

In this section we conduct experiments to compare the two proposed algorithms from different perspectives. Note that, all experiments are run 5 times and the average performance results are reported.

Framwork Comparison. Firstly, we compare the *post-mining* framework and the *mine-and-summarize* framework by implementing them without any optimization. That is, we implement *RCPFast* without the optimization based on *Theorem* 2 and *RCPMS* without the optimization strategy stated in Section 5.4.1 and the approximation strategy in Section 5.4.2. Figure 5.6 shows the running time with respect to the variation of *minpi* and ε respectively. It can be seen that, on sparse data (SynData_1), the performance of the two frameworks is similar. This is because the *post-mining* framework performs quite fast on the sparse data set already. There is not much space for the *mine-and-summarize* framework to improve further. In contrast, when running on dense data sets (e.g., SynData_2 and SynData_3), the *post-mining* framework is relatively slower than the *mine-and-summarize* framework. The performance gap between the two frameworks enlarges on SynData_3 because SynData_3 is bigger than SynData_2, involving more features and events. Consequently, as we will show in Figure 5.8, there are more number of co-location distance computation required, which entangles the *post-mining* framework to spend more time on retrieving table instance information.

Computation Efficiency. We now compare the overall efficiency of the *RCPFast* algorithm and the *RCPMS* algorithm, both implemented with respective optimization strategies. In particular, we also implement a variation of *RCPMS*, called *RCPMS-NA*, which uses the optimization strategy in Section 5.4.1 only. Hence, by comparing *RCPMS* and *RCPMS-NA*, we can study the effectiveness of the approximation strategy in Section 5.4.2.

Figure 5.7 presents the running time on three synthetic data sets with respect to the variation of $minpi$ and ε respectively. It can be observed that *RCPMS* outperforms *RCPFast* in all situations. Comparing the results on the three datasets, we note that the performance advantage of *RCPMS* is not as obvious on sparse data (SynData_1) as on dense data (SynData_2 and SynData_3). This is because when the data is sparse, the size of table instance of a co-location is small, resulting in a short time for computing co-location distance. Consequently, even if *RCPMS* reduces more number of co-location distance computation, the effect of computation saving of *RCPMS* is not obvious. The reasons for *RCPMS* being more efficient on the two dense data sets are different. For SynData_2, the data is dense in terms of the number of co-location instances, which leads to larger size of table instances and longer time to compute co-location distances. Specifically, the time of co-location distance computation is around 60ms on SynData_2 while it is 2.3ms and 3.5ms on SynData_1 and SynData_3, respectively.[1] Therefore, by reducing a few more number of co-location distance computation, *RCPMS* can show efficiency improvement clearly. SynData_3 is dense in terms of the number of co-locations. For this type of dense data, *RCPMS* demonstrates efficiency advantage by directly reducing the number of co-location distance computation.

By comparing *RCPMS* and *RCPMS-NA*, it is obvious, especially on SynData_2 and SynData_3, that the approximation strategy contributes signifi-

[1]The results are obtained by a profile tool for Python, available at https://github.com/rkern/line_profiler.

cantly to the efficiency of the *RCPMS* algorithm.

Reduction of Co-location Distance Computation. The optimization strategies devised form both *RCPFast* and *RCPMS* aim to skip some co-location distance computation. To investigate the effectiveness of these strategies more thoroughly, we conduct experiments to record the number of co-location distance computations for *RCPFast*, *RCPMS* and *RCPMS-NA*, compared with the original number (baseline). Figure 5.8 presents the results by varying $minpi$ and ε respectively. It can be observed that all algorithms involves fewer co-location distance computations than the baseline does and the number of computations in *RCPMS* is the smallest.

In addition, by comparing *RCPFast* against the baseline, we notice that *Theorem 2* does reduce the number of co-location distance computations. However, when the reduced number is not great enough (e.g., Figures 5.8.a, 5.8.c, and 5.8.e), Theorem 5.2 cannot contribute much to the performance improvement. For example, the running times of *RCPFast* in Figures 5.7.a, 5.7.c, and 5.7.e is similar to those of the *post-mining* framework in Figures 5.6.a, 5.6.c, and 5.6.e. This is because it costs extra time for *RCPFast* to find all patterns that can be skipped according to Theorem 5.2.

Compression Rate. Although both Figures 5.7 and 5.8 show that the approximation strategy significantly improves the efficiency of *RCPMS*, as indicated by Lemma 5.2, more RCPs will be discovered by *RCPMS* than *RCPFast*. Hence, we further carry out experiments to evaluate how many more RCPs will be produced by *RCPMS*. We present the results using *compression rate difference*, which is calculated as $\frac{N_M - N_F}{N_{PCP}} \times 100\%$, where N_M and N_F refer to the numbers of patterns output by *RCPMS* and *RCPFast*, respectively. The results in Figure 5.9 show that the compression rate difference is less than 5% on all three datasets, regardless of the variation of parameters. Hence, *RCPMS* effectively improves the computation efficiency by sacrificing the compression rate very slightly.

5.5.2 Experiments on Real Data

Two real-world data sets are used in our experiments. The first one is from the EPA databases (http://www.epa.gov/), which contain environmental activities that affect air, land and water in United States. Different environmental interest types are used as spatial features and each facility represents a spatial event. In our experiment, we use the EPA data of Allen Counties in Indiana State, which consists of 23 features and 647 events in total. The second data set is the points of interest (POI) in California (http://www.usgs.gov/) which was used in [71]. There are 63 category types (e.g., dam, school, and bridge) and 104,770 data points. All the geographic coordinates are transformed to 2-dimensional Cartesian coordinates by Universal Transverse Mercator projection. The spatial distance threshold is 2000 by default (meaning 2km in real world).

We first investigate the compression rate of RCP mining. Figure 5.10 illustrates the compression rate of *RCPFast* and *RCPMS* on the two real data sets by varying *minpi* and ε respectively. We set the default values as $minpi = 0.4$ and $\varepsilon = 0.2$. Generally, the compression rates of the two algorithms are close to each other, except on the EPA dataset when ε is large (e.g., Figure 5.10.b where $\varepsilon = 0.5$). However, in that situation, we note *RCPMS* still can reach a compression rate as high as 75%, which is acceptable. Also it can be observed that the compression rates increase when *minpi* is decreased or ε is increased, which is consistent with the results obtained from the synthetic data sets.

Next, we study the efficiency of the proposed algorithms on real data sets. Figure 5.11 illustrates the running time of the two algorithms with respect to the variation of *minpi* and ε respectively. It shows that *RCPMS* outperforms *RCPFast* on the two real datasets, especially when the data is getting dense (e.g., when *minpi* is decreased) or the requirement of preserving prevalence information is relaxed (e.g., when ε is increased).

To examine whether the summarized representative co-location patterns are meaningful, we also inspect the discovered patterns. We show the experimental results on the POI data set when $minpi = 0.6$ and $\varepsilon = 0.2$ as an example. In this case, seven prevalent co-location patterns are discovered: BUILDING-PARK, CHURCH-PARK,PARK-SCHOOL, BUILDING-PO (*abbreviation of post office*), BUILDING-SCHOOL, BUILDING-CHURCH, and BUILDING-PARK-SCHOOL. Each pattern refers to a set of points of interests that are frequently located in proximity. By running the *RCPMS* algorithm on the data, we discover four representative co-location patterns: BUILDING-PARK-SCHOOL, BUILDING-PO, PARK-SCHOOL and BUILDING-CHURCH-PARK. It can be seen that it makes sense to use the compressed patterns to represent the original patterns. Moreover, among the four RCPs, only the pattern BUILDING-CHURCH-PARK is not a prevalent co-location. However, the PI value of BUILDING-CHURCH-PARK is 0.51, which is greater than $(1 - 0.2) \times 0.6 = 0.48$.

5.6 Related Work

The problem of prevalent co-location pattern mining was first introduced by Morimoto [72], where a *support* metric was defined as the number of instances of a co-location and was used to measure the prevalence of a co-location pattern. However, the metric does not possess the anti-monotonic property. Shekhar and Huang [13] proposed to use *participation ratio* and *minimum participation index* as the interestingness measures that are more statistically meaningful. Various algorithms have been developed to mine prevalent co-location patterns based on the two measures, such as the Apriori-like algorithm [70], the fast algorithm combining the discovery of neighbors with the mining process [16], the join-based algorithm [70], the partial-join algorithm [73] and the join-less algorithm [14]. Other interestingness measures have also been explored, such as mining confident co-locations using the *maximum participation ratio* [74] and mining co-locations based on statistic hypothesis [20].

Frequent pattern summarization has been studied extensively in traditional frequent itemset mining. A variety of concepts have been proposed to find a smaller set of patterns to represent the complete set of frequent patterns, such as maximal patterns [29], closed patterns [30] and non-derivable patterns [31]. One common generalization of closed patterns is the support distance-based approaches [23, 38] which measure the distance between two itemsets based on their supporting transactions and use a pattern to represent/cover its sub-patterns if their distance is no greater than a specified threshold. Although the framework achieves satisfactory compression rate, it cannot not be applied directly to summarize co-location patterns due to the lack of transaction concepts in co-location mining.

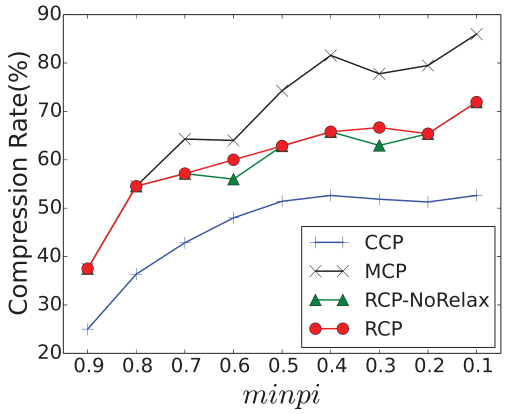
Some initial research efforts have been exerted to summarize prevalent co-location patterns. Mining maximal spatial co-location patterns from a large data set was studied in [67]. This work transforms the data into maximal cliques and then considers maximal cliques as transactions for data mining applications. However, the problem of extracting maximal cliques from a graph is known as NP-hard. Wang et al. used an *order-clique-based* approach to identify table instances and mine maximal co-locations [75]. Closed co-location pattern has been studied by Yoo et al. [14]. They presented a framework to mine top-*k* closed co-location patterns without using minimum prevalence threshold. To our knowledge, our work is the first distance-based approach to summarize co-location patterns using representative patterns, which preserve more prevalence information than maximal co-location patterns and enjoy higher compression rate than closed co-location patterns.

5.7 Conclusions

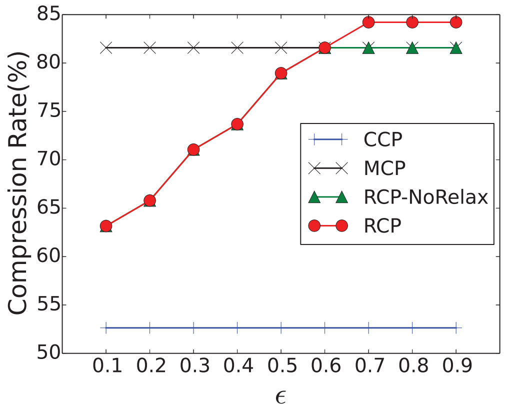
In this chapter, we study the problem of summarizing spatial co-locations using representative patterns. Addressing the missing of transactions in spatial data, a new measure is defined to appropriately quantify the prevalence distance between two co-location patterns, based on which the problem of representative co-location pattern mining is formulated. We propose two efficient algorithms for RCP mining: *RCPFast* and *RCPMS*. *RCPFast* follows existing approaches to adopt a *post-mining* framework that finds representative patterns from the set of discovered prevalent co-location patterns. *RCPMS* employs a novel *mine-and-summarize* paradigm to discover representative patterns directly from the spatial data set, thereby pushing pattern summarization into the co-location mining process. Experimental results show that RCP mining effectively summarizes prevalent co-location patterns, and *RCPMS* significantly improves over *RCPFast* on dense data sets by slightly sacrificing compression rate.

Parameters	Description	SynData_1	SynData_2	SynData_3
N_{seed}	The number of seed co-locations	5	5	50
λ	The parameter of Poisson distribution to define the number of instances of each seed co-location	1000	10000	1000
N_{aux}	The number of features added to construct auxiliary co-locations	3	5	5
α	The ratio that determines the number of instances of additional features when constructing auxiliary co-locations	0.7	0.9	0.9
r_{noise}	The ratio of the number of noise features over the number of features involved in seed and auxiliary co-locations	0.5	0.5	0.5
n_{noise}	The number of noise instances per noise feature	1000	1000	1000
$D_1 \times D_2$	The size of global spatial map	$10^5 \times 10^5$	$10^5 \times 10^5$	$10^5 \times 10^5$
τ	The spatial distance threshold	10	10	10
ε	The default value of co-location distance threshold	0.2	0.2	0.2
$minpi$	The default value of prevalence threshold	0.4	0.4	0.4

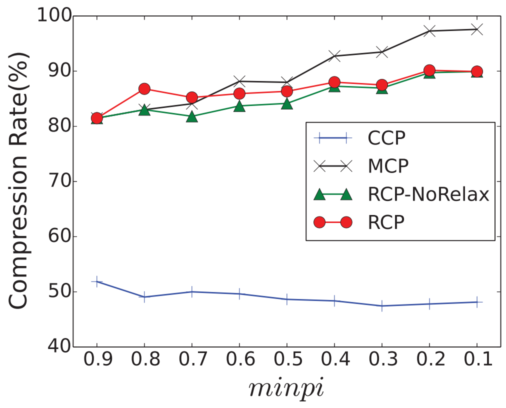
Table 5.2: Parameters used in synthetic data generation



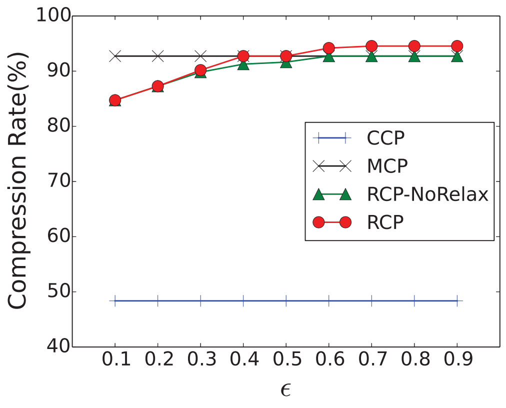
5.5.a SynData_1 (w.r.t. *minpi*)



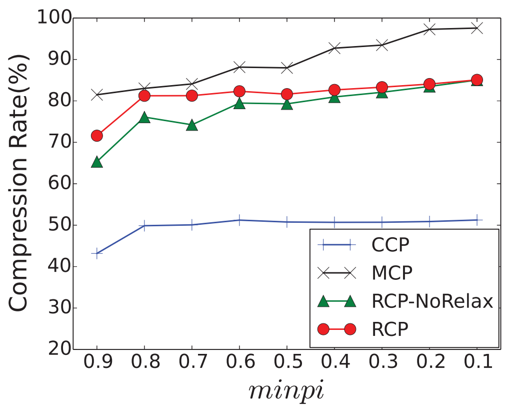
5.5.b SynData_1 (w.r.t. ε)



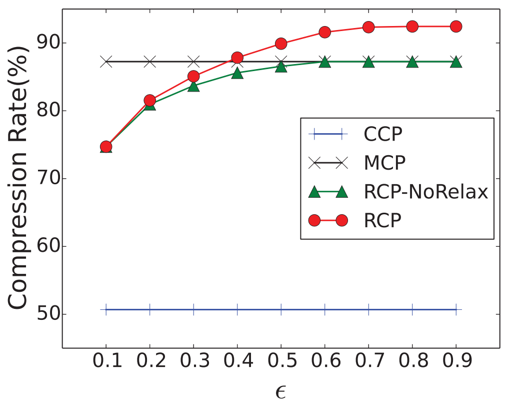
5.5.c SynData_2 (w.r.t. *minpi*)



5.5.d SynData_2 (w.r.t. ε)

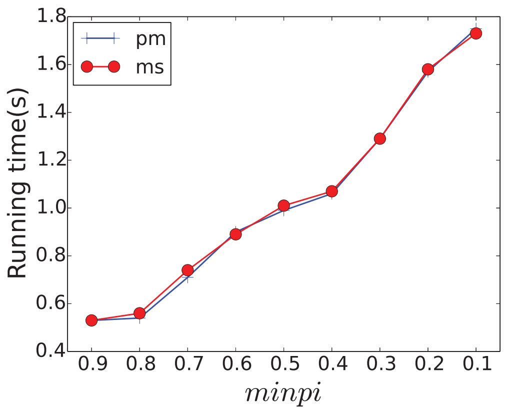


5.5.e SynData_3 (w.r.t. *minpi*)

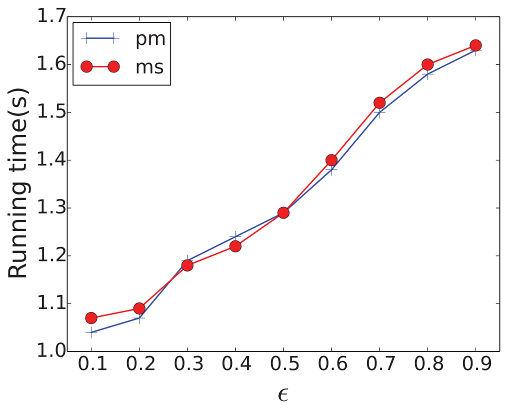


5.5.f SynData_3 (w.r.t. ε)

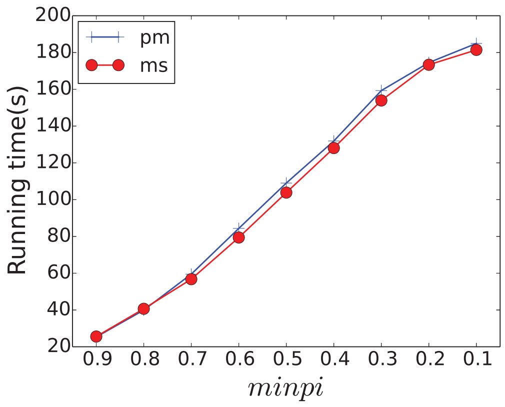
Figure 5.5: Compression rate tests on synthetic data sets.



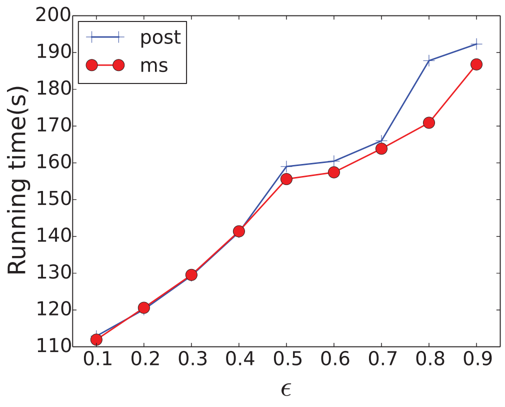
5.6.a SynData_1 (w.r.t. *minpi*)



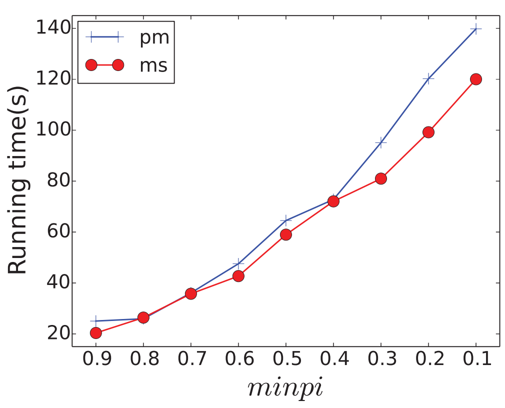
5.6.b SynData_1 (w.r.t. ε)



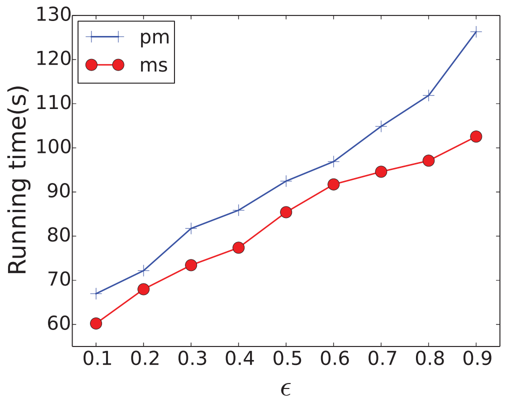
5.6.c SynData_2 (w.r.t. *minpi*)



5.6.d SynData_2 (w.r.t. ε)

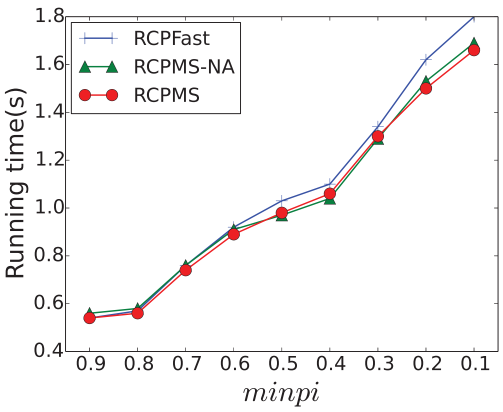


5.6.e SynData_3 (w.r.t. *minpi*)

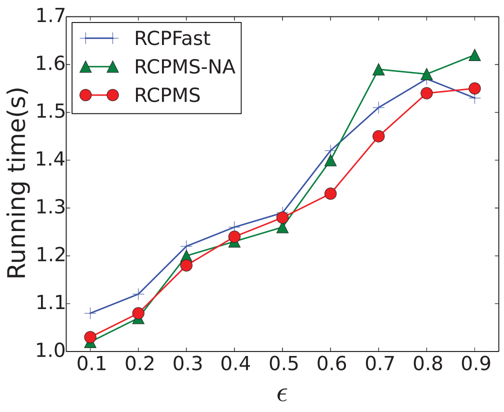


5.6.f SynData_3 (w.r.t. ε)

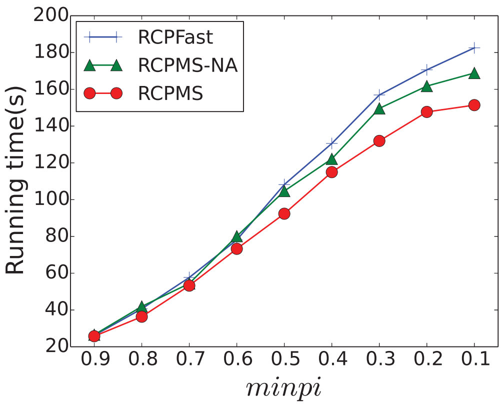
Figure 5.6: Framework comparison on synthetic data sets.



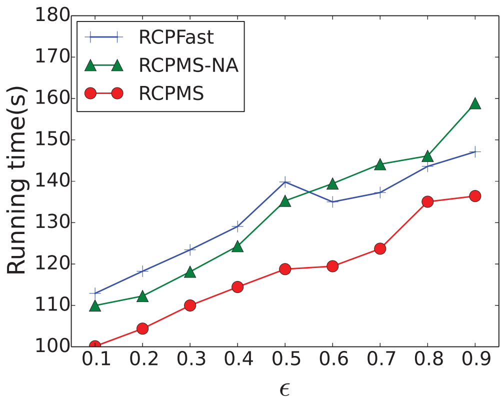
5.7.a SynData_1 (w.r.t. *minpi*)



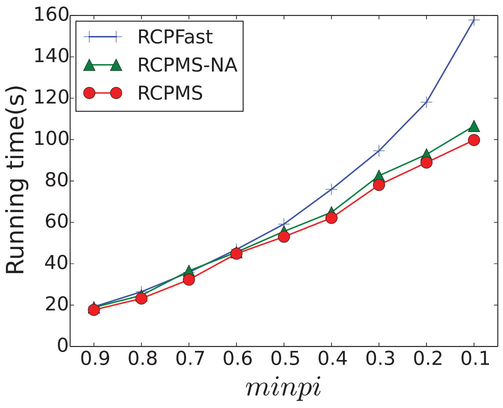
5.7.b SynData_1 (w.r.t. ε)



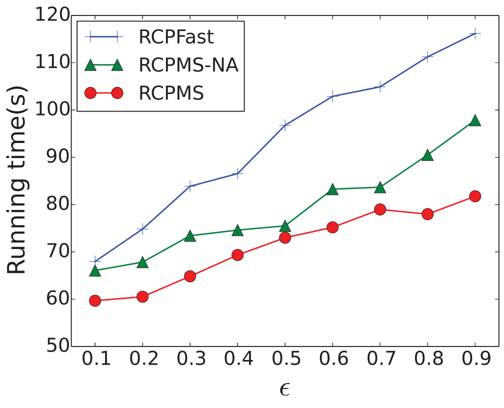
5.7.c SynData_2 (w.r.t. *minpi*)



5.7.d SynData_2 (w.r.t. ε)

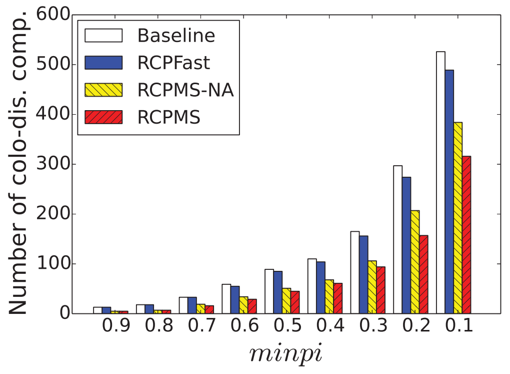


5.7.e SynData_3 (vs. *minpi*)

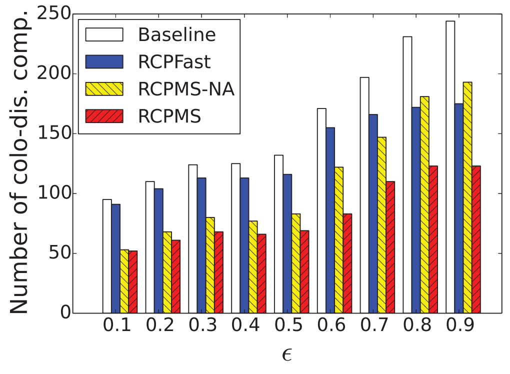


5.7.f SynData_3 (vs. ε)

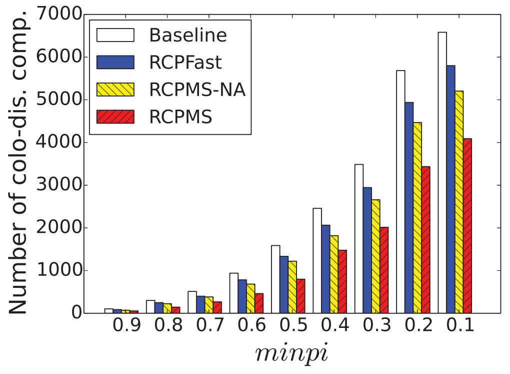
Figure 5.7: Performance tests with *minpi* and ε on synthetic data sets.



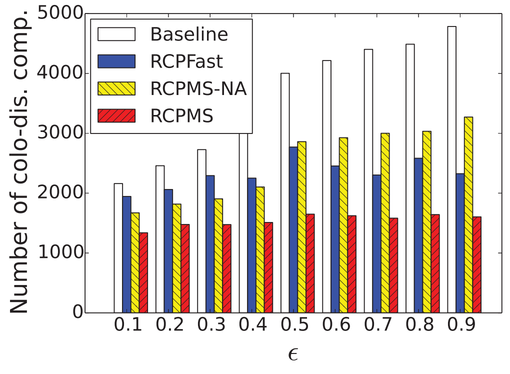
5.8.a SynData_1 (w.r.t. *minpi*)



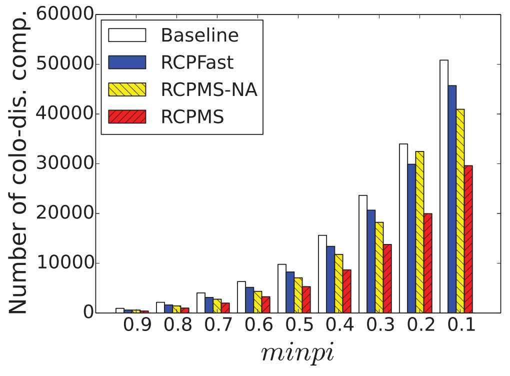
5.8.b SynData_1 (w.r.t. ε)



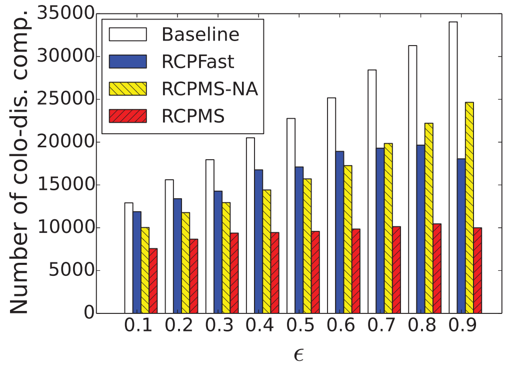
5.8.c SynData_2 (w.r.t. *minpi*)



5.8.d SynData_2 (w.r.t. ε)

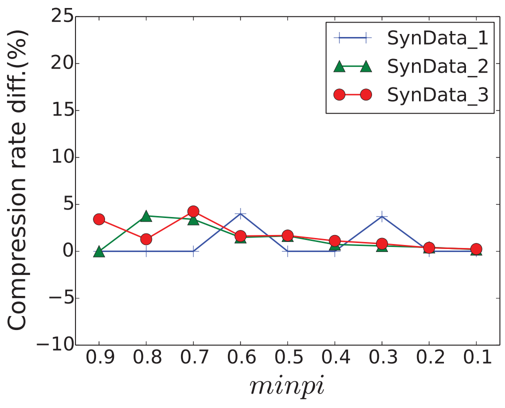


5.8.e SynData_3 (w.r.t. *minpi*)

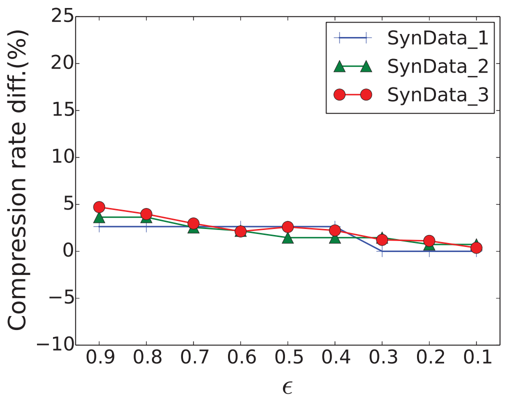


5.8.f SynData_3 (w.r.t. ε)

Figure 5.8: Co-location distance computation analysis on synthetic data sets.

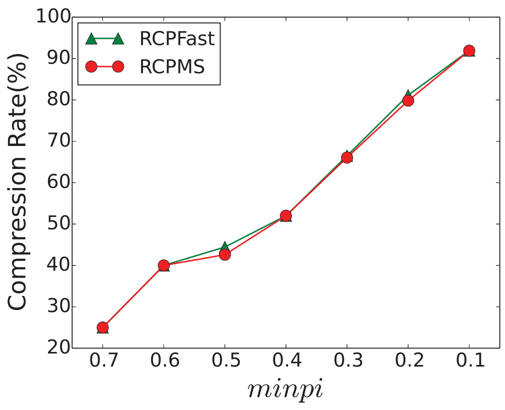


5.9.a w.r.t. *minpi*

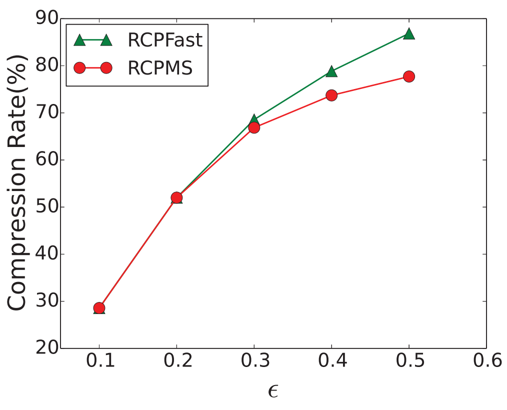


5.9.b w.r.t. ε

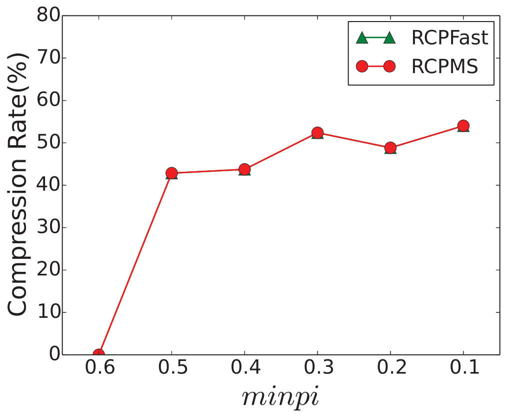
Figure 5.9: Compression rate differences between *RCPMS* and *RCPFast* on synthetic data sets.



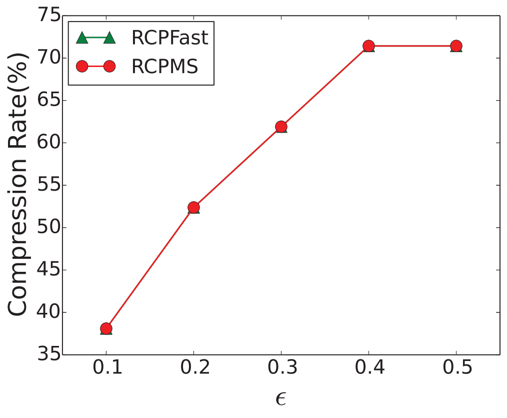
5.10.a EPA (w.r.t. *minpi*)



5.10.b EPA (w.r.t. ε)

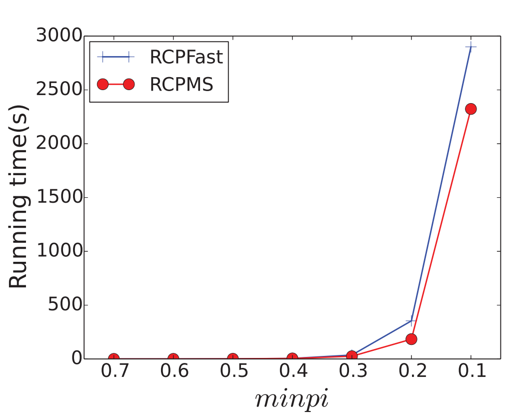


5.10.c POI (w.r.t. *minpi*)

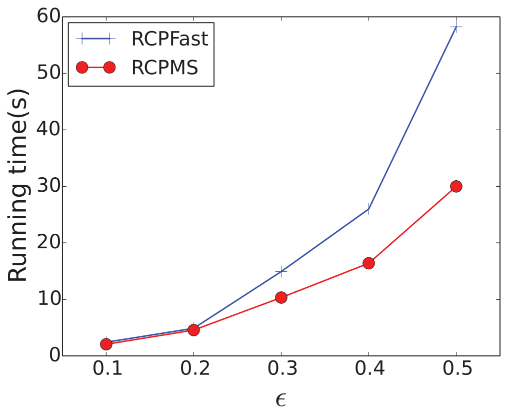


5.10.d POI (w.r.t. ε)

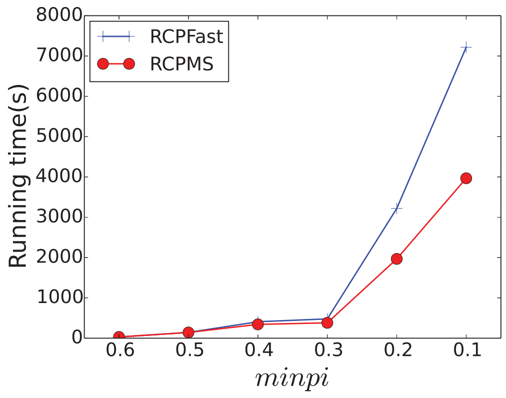
Figure 5.10: Compression rate tests on EPA and POI data sets.



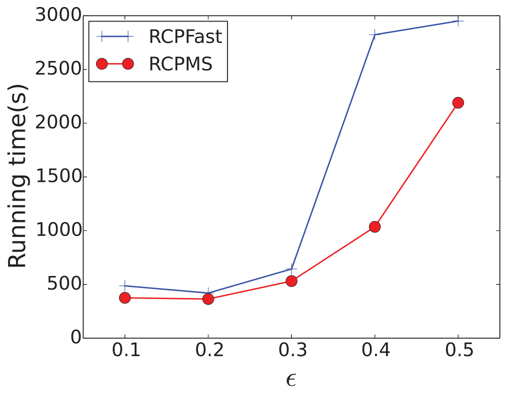
5.11.a EPA (w.r.t. *minpi*)



5.11.b EPA (w.r.t. ε)



5.11.c POI (w.r.t. *minpi*)



5.11.d POI (w.r.t. ε)

Figure 5.11: Performance on EPA and POI data sets.

Chapter 6

Conclusion

In this dissertation, we explored the problem of summarizing data with representative patterns in several scenarios, including summarizing uncertain databases with probabilistic representative frequent patterns, probabilistic tiles, and summarizing spatial databases with representative co-location patterns.

In Chapter 2, we propose the Probabilistic Representative Frequent Pattern (P-RFP) mining problem, which aims to find a small set of patterns to represent the complete set of probabilistic frequent patterns. To address the data uncertainty issue, we define the concept of probabilistic distance and an (ε, δ)-cover relationship between two patterns. P-RFPs are the minimal set of patterns that (ε, δ)-cover the complete set of probabilistic frequent patterns. We develop a P-RFP mining algorithm that uses a dynamic programming based scheme to efficiently check whether one pattern (ε, δ)-covers another. We also exploit effective optimization strategies to further improve the computation efficiency.

In Chapter 3, we propose the APM algorithm, which aims to efficiently and effectively find a small set of patterns to represent the complete set of probabilistic frequent patterns. To address the high computational complexity in examining the joint support probability, we introduce an approximation of the joint support probability with both theoretical and empirical proofs.

In Chapter 4, we study the problem of summarizing transaction data in uncertain databases and formulate the problem as Minimal Probabilistic

Tile Cover Mining. We propose the concept of Probabilistic Price Order and a two-step framework as a solution, including candidates generation and tile construction. We discuss the selection of candidates and devise an algorithm to determine the Probabilistic Price Order of two tiles. Several optimization techniques are further designed to improve the performance.

In Chapter 5, we study the problem of summarizing spatial co-locations using representative patterns. Addressing the missing of transactions in spatial data, a new measure is defined to appropriately quantify the prevalence distance between two co-location patterns, based on which the problem of representative co-location pattern mining is formulated. We propose two efficient algorithms for RCP mining: *RCPFast* and *RCPMS*. *RCPFast* follows existing approaches to adopt a *post-mining* framework that finds representative patterns from the set of discovered prevalent co-location patterns. *RCPMS* employs a novel *mine-and-summarize* paradigm to discover representative patterns directly from the spatial data set, thereby pushing pattern summarization into the co-location mining process.

Although the results obtained in this dissertation are encouraging, this is not the end of the road. The fast pace of technology advancement and the increasing significance of data summarization offers plentiful possible research directions in the future. Based on current works, we will study the summarization of other types of database, such as numeric data, time series data, and graph data. We will also explore the real-world applications of the representative pattern-based summarization approaches in various domains (for example, texts, images, and videos), which may yield influence on industry and society.

References

[1] Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques (Third Edition)*. Morgan Kaufmann Publishers, San Francisco, CA, USA, 3 edition, 2011.

[2] Yongxin Tong, Lei Chen, Yurong Cheng, and Philip S. Yu. Mining frequent itemsets over uncertain databases. *Very Large Data Base Endowment (VLDB)*, 5(11):1650–1661, 2012.

[3] Chun-Kit Chui, Ben Kao, and Edward Hung. Mining frequent itemsets from uncertain data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 47–58, 2007.

[4] Thomas Bernecker, Hans-Peter Kriegel, Matthias Renz, Florian Verhein, and Andreas Zuefle. Probabilistic frequent itemset mining in uncertain databases. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 119–128, 2009.

[5] Liwen Sun, Reynold Cheng, David W. Cheung, and Jiefeng Cheng. Mining uncertain data with probabilistic guarantees. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 273–282, 2010.

[6] Carson Kai-Sang Leung, Mark Anthony F. Mateo, and Dale A. Brajczuk. A tree-based approach for frequent pattern mining from uncertain data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 653–661, 2008.

[7] Charu C. Aggarwal, Yan Li, Jianyong Wang, and Jing Wang. Frequent pattern mining with uncertain data. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 29–38, 2009.

[8] Toon Calders, Calin Garboni, and Bart Goethals. Approximation of frequentness probability of itemsets in uncertain data. In *International Conference on Data Engineering (ICDE)*, pages 749–754, 2010.

[9] Carson Kai-Sang Leung. *Frequent Pattern Mining*, chapter Uncertain Frequent Pattern Mining, pages 339–367. Springer International Publishing, Cham, 2014.

[10] Erich A. Peterson and Peiyi Tang. Fast approximation of probabilistic frequent closed itemsets. In *ACM Annual Southeast Regional Conference (ASRC)*, pages 214–219, 2012.

[11] Peiyi Tang and Erich A. Peterson. Mining probabilistic frequent closed itemsets in uncertain databases. In *ACM Annual Southeast Regional Conference (ASRC)*, pages 86–91, 2011.

[12] Yongxin Tong, Lei Chen, and Bolin Ding. Discovering threshold-based frequent closed itemsets over probabilistic data. In *International Conference on Data Engineering (ICDE)*, pages 270–281, 2012.

[13] Shashi Shekhar and Yan Huang. Discovering spatial co-location patterns: a summary of results. *International Symposium on Spatial and Temporal Databases (SSTD)*, pages 236–256, 2001.

[14] Jin Soung Yoo and Shashi Shekhar. A joinless approach for mining spatial colocation patterns. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 18(10):1323–1337, 2006.

[15] Yan Huang, Jian Pei, and Hui Xiong. Mining co-location patterns with rare events from spatial data sets. *GeoInformatica*, 10(3):239–260, 2006.

[16] Xin Zhang, Nikos Mamoulis, David W. Cheung, and Yutao Shou. Fast mining of spatial collocations. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 384–393, 2004.

[17] Hui Xiong, Shashi Shekhar, Yan Huang, Vipin Kumar, Xiaobin Ma, and Jin Soung Yoo. A framework for discovering co-location patterns in data sets with extended spatial objects. In *SIAM International Conference on Data Mining (SDM)*, pages 78–89, 2004.

[18] Jin Soung Yoo, Shashi Shekhar, and Mete Celik. A join-less approach for co-location pattern mining: A summary of results. In *IEEE International Conference on Data Mining (ICDM)*, pages 813–816, 2005.

[19] Mete Celik, James M. Kang, and Shashi Shekhar. Zonal co-location pattern discovery with dynamic parameters. In *IEEE International Conference on Data Mining (ICDM)*, pages 433–438, 2007.

[20] Sajib Barua and Jörg Sander. Sscp: Mining statistically significant co-location patterns. In *International Symposium on Spatial and Temporal Databases (SSTD)*, pages 2–20, 2011.

[21] Feng Qian, Qinming He, Kevin Chiew, and Jiangfeng He. Spatial co-location pattern discovery without thresholds. *Knowledge Information System (KIS)*, 33(2):419–445, 2012.

[22] Charu C. Aggarwal and P.S. Yu. A survey of uncertain data algorithms and applications. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 21(5):609–623, 2009.

[23] Dong Xin, Jiawei Han, Xifeng Yan, and Hong Cheng. Mining compressed frequent-pattern sets. In *International Conference on Very Large Data Bases (VLDB)*, pages 709–720, 2005.

[24] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 207–216, New York, NY, USA, 1993. ACM.

[25] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *International Conference on Very Large Data Bases (VLDB)*, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[26] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In *ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 1–12, 2000.

[27] Chun-Kit Chui and Ben Kao. A decremental approach for mining frequent itemsets from uncertain data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 64–75, 2008.

[28] Liang Wang, Reynold Cheng, Sau Dan Lee, and David Cheung. Accelerating probabilistic frequent itemset mining: a model-based approach. In *ACM International Conference on Information and Knowledge Management (CIKM)*, pages 429–438, 2010.

[29] Roberto J. Bayardo Jr. Efficiently mining long patterns from databases. In *ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 85–93, 1998.

[30] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets for association rules. In *International Conference on Database Theory (ICDT)*, pages 398–416, 1999.

[31] Toon Calders and Bart Goethals. Mining all non-derivable frequent itemsets. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, pages 74–85, 2002.

[32] Jiawei Han, Jianyong Wang, Ying Lu, and Petre Tzvetkov. Mining top-k frequent closed patterns without minimum support. In *IEEE International Conference on Data Mining (ICDM)*, pages 211–218, 2002.

[33] Jianyong Wang, Jiawei Han, Ying Lu, and Petre Tzvetkov. Tfp: an efficient algorithm for mining top-k frequent closed itemsets. *IEEE Transactions on Knowledge and Data Engineering (ICDE)*, 17(5):652–663, May 2005.

[34] Dong Xin, Hong Cheng, Xifeng Yan, and Jiawei Han. Extracting redundancy-aware top-k patterns. In *ACM SIGKDD International Con-*

ference on Knowledge Discovery and Data Mining (KDD), pages 444–453, New York, NY, USA, 2006. ACM.

[35] Xifeng Yan, Hong Cheng, Jiawei Han, and Dong Xin. Summarizing itemset patterns: a profile-based approach. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 314–323, 2005.

[36] Ruoming Jin, Muad Abu-Ata, Yang Xiang, and Ning Ruan. Effective and efficient itemset pattern summarization: regression-based approaches. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 399–407, 2008.

[37] Ardian Kristanto Poernomo and Vivekanand Gopalkrishnan. Cp-summary: a concise representation for browsing frequent itemsets. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 687–696, 2009.

[38] Guimei Liu, Haojun Zhang, and Limsoon Wong. Finding minimum representative pattern sets. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 51–59, 2012.

[39] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 4(3):233–235, 1979.

[40] Chunyang Liu, Ling Chen, and Chengqi Zhang. Mining probabilistic representative frequent patterns from uncertain data. In *SIAM International Conference on Data Mining (SDM)*, pages 73–81, 2013.

[41] Jun Shao. *Mathematical Statistics*. Springer, Berlin, 2009.

[42] H. Cramér and H. Wold. Some theorems on distribution functions. *The Journal of the London Mathematical Society*, 11:290–295, 1936.

[43] D.R. Cox. The continuity correction. *Biometrika*, 57(1):217–219, 1970.

[44] Yang Xiang, Ruoming Jin, David Fuhry, and Feodor F. Dragan. Succinct summarization of transactional databases: An overlapped hyper-rectangle scheme. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 758–766, 2008.

[45] Floris Geerts, Bart Goethals, and Taneli Mielikainen. Tiling databases. In *Discovery Science*, volume 3245, pages 278–289, 2004.

[46] Yang Xiang, Ruoming Jin, David Fuhry, and Feodor F. Dragan. Summarizing transactional databases with overlapped hyperrectangles. *Data Mining and Knowledge Discovery (DMKD)*, 23(2):215–251, 2011.

[47] Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. A unifying framework for mining approximate top-k binary patterns. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 26(12):2900–2913, Dec 2014.

[48] Petteri Sevon, Lauri Eronen, Petteri Hintsanen, Kimmo Kulovesi, and Hannu Toivonen. Link discovery in graphs derived from biological databases. In *Data Integration in the Life Sciences (DILS)*, pages 35–49, 2006.

[49] Steven Finch. *Mathematical Constants*. Cambridge University Press, 2003.

[50] Chunyang Liu, Ling Chen, and Chengqi Zhang. Summarizing probabilistic frequent patterns: A fast approach. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 527–535, 2013.

[51] Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *ACM Symposium on Theory of Computing (STOC)*, pages 624–633, 2014.

[52] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[53] Pauli Miettinen, Taneli Mielikainen, Aristides Gionis, Gautam Das, and Heikki Mannila. The discrete basis problem. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 20(10):1348–1362, 2008.

[54] Nikolaj Tatti and Jilles Vreeken. Discovering descriptive tile trees. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, pages 9–24, 2012.

[55] Toon Calders, Calin Garboni, and Bart Goethals. Efficient pattern mining of uncertain data with sampling. In *PAKDD*, pages 480–487, 2010.

[56] Aristides Gionis, Heikki Mannila, and Jouni K. Seppanen. Geometric and combinatorial tiles in 0-1 data. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, volume 3202, pages 173–184. 2004.

[57] Ruoming Jin, Yang Xiang, and Lin Liu. Cartesian contour: A concise representation for a collection of frequent sets. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 417–426, 2009.

[58] Francesco Bonchi, Matthijs van Leeuwen, and Antti Ukkonen. Characterizing uncertain data using compression. In *SIAM International Conference on Data Mining (SDM)*, pages 534–545, 2011.

[59] Jian Pei, Guozhu Dong, Wei Zou, and Jiawei Han. Mining condensed frequent-pattern bases. *Knowledge and Information Systems (KIS)*, 6(5):570–594, 2004.

[60] Jian Pei, Guozhu Dong, Wei Zou, and Jiawei Han. On computing condensed frequent pattern bases. In *IEEE International Conference on Data Mining (ICDM)*, pages 378–385, 2002.

[61] Chao Wang and Srinivasan Parthasarathy. Summarizing itemset patterns using probabilistic models. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 730–735, 2006.

[62] Jilles Vreeken, Matthijs van Leeuwen, and Arno Siebes. Krimp: Mining itemsets that compress. *Data Mining and Knowledge Discovery (DMKD)*, 23(1):169–214, 2011.

[63] Jilles Vreeken, Matthijs van Leeuwen, and Arno Siebes. Characterising the difference. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 765–774, 2007.

[64] Lizhen Wang, Lihua Zhou, Joan Lu, and Jim Yip. An order-clique-based approach for mining maximal co-locations. *Information Science*, 179(19):3370–3382, September 2009.

[65] Jin Soung Yoo and Mark Bow. Mining top-k closed co-location patterns. In *IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM)*, pages 100–105, 2011.

[66] Ghazi Al-Naymat. Enumeration of maximal clique for mining spatial co-location patterns. In *ACS International Conference on Computer Systems and Applications (AICCSA)*, pages 126–133, 2008.

[67] Natwar Modani and Kuntal Dey. Large maximal cliques enumeration in sparse graphs. In *ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1377–1378, 2008.

[68] James Cheng, Linhong Zhu, Yiping Ke, and Shumo Chu. Fast algorithms for maximal clique enumeration with limited memory. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1240–1248, 2012.

[69] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms (Second Edition)*. MIT Press, 2001.

[70] Yan Huang, Shashi Shekhar, and Hui Xiong. Discovering colocation patterns from spatial data sets: A general approach. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(12):1472–1485, 2004.

[71] Feifei Li, Dihan Cheng, Marios Hadjieleftheriou, George Kollios, and Shang-Hua Teng. On trip planning queries in spatial databases. In *International Symposium on Spatial and Temporal Databases (SSTD)*, pages 273–290, 2005.

[72] Yasuhiko Morimoto. Mining frequent neighboring class sets in spatial databases. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 353–358, 2001.

[73] Jin Soung Yoo and Shashi Shekhar. A partial join approach for mining co-location patterns. In *International Conference on Advances in Geographic Information Systems (GIS)*, pages 241–249, 2004.

[74] Yan Huang, Hui Xiong, Shashi Shekhar, and Jian Pei. Mining confident colocation rules without a support threshold. In *ACM Symposium on Applied Computing (SAC)*, pages 497–501, 2003.

[75] Song Wang, Yan Huang, and Xiaoyang Sean Wang. Regional co-locations of arbitrary shapes. In *International Symposium on Spatial and Temporal Databases (SSTD)*, pages 19–37, 2013.