People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
Mohamed El Bachir El Ibrahimi University of Bordj Bou Arréridj
Faculty of Mathematics and Computer Science
Department of Computer Science

THESIS

In order to obtain the Doctorate degree in LMD (3rd cycle)
Branch : Computer science
Option : Decision Support Systems Engineering

Privacy preserving pattern mining from uncertain databases

By: Mira Lefkir

Publicly defended on: dd/mm/yyyy

In front of the jury composed of:

Pr. Abderraouf Bouziane	University of B.B.A	**President**
Pr. Farid Nouioua	University of B.B.A	**Supervisor**
Pr. Philippe Fournier-Viger	Shenzhen University (China)	**Co-Supervisor**
Pr. Mustapha Bourahla	University of M'Sila	**Examiner**
Dr. Chafia Kara Mohamed	University of Setif 1	**Examiner**
Dr. Abdelouahab Attia	University of B.B.A	**Examiner**

2024/2025

Dedication

I dedicate my dissertation work to my family my brother and my mother. A special feeling of gratitude to my father's soul, I also dedicate this dissertation to my husband who have supported me throughout the process. I will always appreciate all they have done,

I dedicate this work and give special thanks to my wonderful son Abderrahmane and Haithem for being there for me throughout the entire doctorate program. Both of you have been my best cheerleaders.

Acknowledgment

First and foremost, I am extremely grateful to my supervisors, Prof. Farid Nouioua and co-supervisor Prof. Phillipe Fournier-Viger for their invaluable advices, continuous support, and patience during my PhD studies. Their extensive knowledge and plentiful experience have encouraged me in my academic research and daily life. I would like to thank all the teachers from the faculty of computer sciences for their support, and also i would like to thank jury member for accepting to juge my works.

Abstract

With advancements in data analysis and processing techniques, the release of micro-data for research purposes, such as disease outbreak studies or economic pattern analysis, has become prevalent. However, these datasets, while valuable for researchers, often contain sensitive information that poses privacy risks to individuals. Privacy-preserving data mining (PPDM) has emerged as a critical field to address these concerns by concealing sensitive information while still enabling the extraction of useful insights. This task is NP-hard and involves the challenge of balancing the concealment of sensitive itemsets with the preservation of non-sensitive ones during data extraction. Numerous algorithms have been developed for deterministic databases, where information is binary (present or absent). This thesis explores a novel PPDM approach in the context of uncertain databases, where information is represented by probabilistic values. The sanitization process, aimed at hiding sensitive information, introduces side effects such as hiding failure, missing cost, artificial cost, and dissimilarity. These side effects are considered as objective functions to be minimized. To achieve the goal of PPDM, a Multi-Objective Optimization problem is formulated, and metaheuristic algorithms are employed. Specifically, the NSGA-II algorithm is applied, leading to the development of the NSGAII4ID algorithm for deterministic databases. This algorithm hides sensitive frequent itemsets by removing selected items from chosen transactions, representing a pioneering effort in privacy-preserving data mining. Furthermore, for uncertain databases, a novel algorithm named U-NSGAII4ID is proposed, addressing the multi-objective optimization problem by encoding a set of items to remove from selected transactions. Additionally, three heuristic approaches for PPDM in uncertain databases are introduced: the aggregate approach, which removes transactions; the disaggregate approach, which removes selected items from each transaction; and the hybrid approach, combining the two former approaches. Experimental evaluations compare these approaches, demonstrating their effectiveness in preserving sensitive itemsets in uncertain databases. *

Résumé

Les avancées en analyse de données ont conduit à une augmentation fréquente de la publication de micro-données à des fins de recherche, notamment dans les domaines tels que l'étude des épidémies et l'analyse des modèles économiques. Cependant, ces ensembles de données, bien utiles pour les chercheurs, présentent souvent des informations sensibles pouvant compromettre la vie privée individuelle. La préservation de la vie privée dans l'extraction de données (PPDM) est devenue cruciale pour résoudre ces préoccupations tout en permettant l'extraction d'informations utiles. Cette tâche complexe, NP-difficile, implique l'équilibre délicat entre la dissimulation des éléments sensibles et la préservation de ceux qui ne le sont pas lors de l'extraction de données. Dans le contexte des bases de données incertaines, où l'information est représentée par des valeurs probabilistes, cette thèse explore une nouvelle approche de PPDM. Le processus de désinfection, visant à masquer des informations sensibles, introduit des effets secondaires tels que l'échec de dissimulation, le coût manquant, le coût artificiel et la dissimilitude, considérés comme des fonctions objectives à minimiser. Pour atteindre l'objectif du PPDM, des algorithmes métaheuristiques sont utilisés, notamment l'algorithme NSGA-II. Cela a conduit au développement de l'algorithme NSGAII4ID pour masquer des ensembles d'éléments fréquents sensibles dans les bases de données déterministes. Dans le contexte des bases de données incertaines, un nouvel algorithme, U-NSGAII4ID, est proposé pour résoudre le problème d'optimisation multi-objectifs en encodant un ensemble d'éléments à supprimer de transactions sélectionnées. Trois approches heuristiques pour le PPDM dans les bases de données incertaines sont également introduites : l'approche agrégée, qui supprime des transactions ; l'approche désagrégée, qui supprime des éléments sélectionnés de chaque transaction ; et l'approche hybride, combinant les deux premières approches. Des études expérimentales comparent ces approches, démontrant leur efficacité dans la préservation des ensembles d'éléments sensibles dans les bases de données incertaines.

ملخص

ومع تطور تقنيات تحليل البيانات، أصبح إصدار البيانات الجزئية لأغراض البحث أكثر انتشاراً، مثل في دراسات تفشي الأمراض أو تحليل الأنماط الاقتصادية. ورغم قيمة هذه المجموعات للباحثين، إلا أنها غالباًا تحتوي على معلومات حساسة تهدد الخصوصية الفردية. برزت استخراج البيانات التي تحافظ على الخصوصية (حفظ الخصوصية للبيانات) كمجال حيوي لمعالجة هذه المخاوف بإخفاء المعلومات الحساسة مع السماح بالتحليلات المفيدة. تحدث هذه المهمة الصعبة والتي تصنف كصعبة عن التحدي في تحقيق توازن بين إخفاء العناصر الحساسة والحفاظ على تلك غير الحساسة خلال استخراج البيانات. تم تطوير العديد من الخوارزميات لقواعد البيانات الحتمية حيث تكون المعلومات ثنائية (موجودة أو غائبة). في سياق قواعد البيانات غير المؤكدة، حيث يتم تمثيل المعلومات بقيم احتمالية، تستكشف هذه الأطروحة نهجاًا جديداً لـ (حفظ الخصوصية للبيانات). تقدم عملية التعقيم التي تهدف إلى إخفاء المعلومات الحساسة، آثاراً جانبية مثل إخفاء الفشل وتكلفة الفقد والتكلفة الاصطناعية والاختلاف، وتُعَتبر هذه الآثار الجانبية وظائف موضوعية يتعين تقليلها. يتم استخدام مشكلة التحسين متعددة الأهداف والخوارزميات التلوية، وتحديداً خوارزمية (خوارزمية التصنيف غير المهيمن للتطور الوراثي النوع الثاني)، مما أدى إلى تطوير خوارزمية لقواعد البيانات الحتمية. تخفي هذه الخوارزمية مجموعات العناصر المتكررة الحساسة عن طريق إزالة العناصر المحددة من المعاملات المختارة، وهو جهد رائد في استخراج البيانات للحفاظ على الخصوصية. بالإضافة إلى ذلك، لقواعد البيانات غير المؤكدة، تم اقتراح خوارزمية جديدة لحل مشكلة التحسين متعددة الأهداف من خلال تشفير مجموعة من العناصر للحذف من المعاملات المحددة. علاوة على ذلك، تم تقديم ثلاثة أساليب إرشادية في قواعد البيانات غير المؤكدة: النهج الكلي الذي يزيل المعاملات، والنهج التفصيلي الذي يزيل العناصر المختارة من كل معاملة، والنهج المختلط الذي يجمع بين النهجين السابقين. تقارن الدراسات التجريبية هذه الأساليب، مما يظهر فعاليتها في الحفاظ على مجموعات العناصر الحساسة في قواعد البيانات غير المؤكدة.

Table of contents

List of Figures

List of Tables

Chapter 1: General Introduction

1.1 Context

Data mining and knowledge discovery in databases represent emerging research fields that focus on the automated extraction of previously undiscovered patterns from vast datasets. With the recent progress in data collection, dissemination, and associated technologies, a new research era has dawned, prompting a reevaluation of existing data mining algorithms from the perspective of privacy preservation. This paradigm shift recognizes the importance of safeguarding individual privacy in the era of expanding data availability and utilization.

In recent times, progress in hardware technology has resulted in a notable expansion of storage and recording capabilities for personal data related to consumers and individuals. This proliferation of data storage potential has generated apprehensions regarding the potential for personal data to be misused for various purposes. To address these concerns, a range of techniques has emerged, aiming to conduct data mining tasks while safeguarding privacy. These privacy-preserving data mining methods draw from a diverse range of fields, encompassing subjects like data mining, cryptography, and information concealment [2].

The issue of privacy-preserving data mining [3] has garnered substantial attention in recent years, driven by mounting concerns about the privacy of underlying data. This surge in interest has resulted in a proliferation of research papers in the field of privacy-preserving data mining, with topics explored by various communities and approached with different styles. Consequently, there is a growing need to organize these topics in a manner that acknowledges the relative importance of different research areas. Moreover, the domain of privacy-preserving data mining has been independently explored by distinct communities, including cryptography, databases, and statistical disclosure control. While there are instances where the lines of work

within these communities are quite similar, there exists a notable lack of integration, hindering the provision of a broader perspective. Despite data mining's successful advancements in fields such as machine learning, statistics, and artificial intelligence, it is often associated with extracting information that could compromise confidentiality. This aspect has fueled growing ethical concerns regarding the sharing of personal information for data mining activities [4].

Furthermore, it's important to emphasize that Privacy-Preserving Data Mining (PPDM) isn't solely concerned with protecting privacy during the mining phase. It extends its scope to encompass privacy considerations in various stages of the knowledge discovery process, including data pre-processing and post-processing [5]. This comprehensive approach addresses the challenges faced by organizations or individuals when sensitive information is at risk of being lost. [6] introduced a method to enhance privacy while preserving the statistical characteristics of data. Besides, [7] proposed a novel multiparty collaborative privacy-preserving mining technique. This approach securely combines various geometric perturbations, each preferred by different parties, utilizing the concept of keys. Other related research explores techniques like top-down specialization for privacy preservation [8] and workload-aware methods for data anonymization [9]. Furthermore, there is an ongoing discussion regarding privacy-preserving data mining in scenarios involving vertically or horizontally partitioned data [10].

In recent years, the prevalence of uncertain data has surged, primarily driven by advances in data collection technologies that produce measurements with inherent imprecision. This surge in uncertainty is observable across a wide array of real-life applications, including location-based services, sensor monitoring systems, medical analyses, and face recognition systems. Given that frequent itemset mining is a foundational concept in the field of data mining, the exploration of mining frequent itemsets over uncertain databases has garnered substantial attention [11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21]. For instance, the proliferation of wireless sensor networks has led to the accumulation of vast amounts of data. Consequently, the definition of frequent itemsets over uncertain databases becomes a pivotal concern.

In the realm of deterministic data, frequency evaluation is straightforward: an itemset is deemed frequent if and only if the support (frequency) of this itemset is not less than a specified minimum support threshold, often denoted as "min_sup" [22, 23, 17, 24]. However, in contrast to deterministic data, defining a frequent itemset over uncertain data presents two distinct semantic interpretations: the expected support-based frequent itemset [11, 16] and the

probabilistic frequent itemset [12]. Both of these interpretations treat the support of an itemset as a discrete random variable.

1.2 Objectives

A substantial portion of application data contains personal and sensitive information, including details about an individual's financial status, political affiliations, and medical history. The exposure of such personal data can significantly jeopardize individuals' privacy. In recent years, Privacy Preserving Data Mining (PPDM) has emerged as a crucial subfield of data mining and a prominent topic in the context of safeguarding privacy.

The primary objective of PPDM is to shield personal and sensitive information from being divulged to the public during the data mining process. This protection is achieved by employing a sanitization method, which is a process enabling the modification of the database. This modification can involve the removal or addition of items, or even the elimination of entire transactions, all aimed at concealing sensitive information effectively.

Privacy-preserving data mining primarily aims to devise techniques that protect specific sensitive information from being disclosed to unauthorized parties or intruders. In simpler terms, the core concern with privacy threats is the potential revelation of an individual's sensitive identity and personal information. Various types of privacy threats have the potential to unveil an individual's sensitive information:

- Identity disclosure [25]: In identity disclosure threat, intruder can get the individual identity from published data. This threat is affined to direct identifier attribute.
- Attribute disclosure [26]: In attribute disclosure threat, intruder can reveal individual's sensitive information. This threat is affined to sensitive attribute.
- Membership disclosure [27]: Any information concerning individual is disclosed from data set, known as membership disclosure. This may happen when data is not protected from identity disclosure.

The overarching aim of this study is to explore the realm of Privacy-Preserving Data Mining in the context of frequent itemset mining from both traditional and uncertain databases. The specific objectives of this research are as follows:

1. To formulate algorithms that can manipulate the original data in a manner that preserves the privacy of sensitive information, particularly sensitive frequent itemsets, even after the data mining process has occurred.

2. To achieve a balance where Privacy-Preserving Data Mining methods can modify the original data effectively, preserving the privacy of user data, while also allowing the mining models to be reconstructed from the modified data with a reasonable degree of accuracy.

In summary, Privacy-Preserving Data Mining methods aim to alter the original data in such a way that user data privacy is upheld, and simultaneously, the mining models can be reconstructed from the modified data with an acceptable level of accuracy. These methods are diverse and can be tailored to the specific data collection model and user privacy requirements, as reflected in the existing literature.

1.3 Methodology and results

To accomplish the primary objective of this thesis, the research adopts a multi-objective optimization approach. Within the framework of this thesis, we have structured our work into three distinct contributions:

This thesis makes a significant contribution by introducing an innovative Privacy-Preserving Data Mining (PPDM) algorithm named NSGAII4ID. Unlike conventional methods that eliminate entire transactions, this novel algorithm selectively removes specific items from transactions to conceal sensitive frequent itemsets. To our knowledge, this approach is unprecedented. The sanitization process is formulated as a multi-objective optimization problem, aiming to address four main side effects: hiding failure, missing cost, artificial cost, and database dissimilarity. Notably, the artificial cost equals always zero in our specific context. NSGAII4ID stands out for its dual-level approach to database sanitization. At the transaction level, a multi-objective optimization algorithm identifies a subset of candidate transactions for modification. Then, at the item level, NSGAII4ID determines an optimal subset of items to be removed from each candidate transaction. It is noteworthy that this problem aligns closely with the Set Cover Problem (SCP), which we tackle using a rapid and efficient greedy polynomial algorithm.

The second significant contribution of this research introduces a novel methodology known as U-NSGAII4ID, designed for the purpose of concealing sensitive expected frequent itemsets by selectively removing specific items from transactions within an uncertain database. This method, an extension of our previously proposed NSGAII4ID for traditional databases, leverages the Non-dominated Sorting Genetic Algorithm II (NSGA-II). Notably, three side effects, specifically hiding failure, missing cost, and database dissimilarity, have been incorporated as distinct objective functions to be minimized.

Similar to NSGAII4ID, U-NSGAII4ID takes on the challenge of database sanitization through a two-tiered optimization process. Initially, at the transaction level, it utilizes a multi-objective optimization algorithm to identify a subset of candidate transactions requiring modification. Subsequently, at the item level, U-NSGAII4ID aims to identify an optimal subset of items to be removed from each candidate transaction. Notably, this problem aligns precisely with the Weighted Set Cover Problem (WSCP), dictating the selection of optimal items for elimination from chosen transactions. This innovative method marks a significant advancement in privacy-preserving data mining, particularly for uncertain databases.

The third pivotal contribution of this thesis involves the presentation of three heuristic methodologies tailored for Privacy-Preserving Data Mining in the context of uncertain databases. These heuristic techniques have been crafted to effectively safeguard sensitive expected frequent itemsets while simultaneously mitigating undesirable side effects during the data mining process. The first heuristic, termed the aggregate approach, entails the selective removal of certain transactions from the uncertain database as a means of preserving privacy. The second, known as the disaggregate approach, focuses on the strategic elimination of specific items from individual transactions. The third, referred to as the hybrid approach, represents a versatile amalgamation of the two prior strategies, involving the removal of particular items from selected transactions. Together, these heuristics seek to strike a delicate balance between maintaining the confidentiality of sensitive information and reducing adverse consequences in the course of data mining over uncertain databases.

1.4 Thesis outline

This thesis consists of five chapters, in addition to this introductory chapter and a last conclusive chapter. The remaining chapters are organized as follows:

1. **Chapter 2: Pattern Mining** - This chapter provides a foundational understanding of data mining concepts and offers an overview of various methods for pattern extraction from databases.

2. **Chapter 3: Privacy Preserving Data Mining** - In this chapter, a comprehensive survey of the state-of-the-art in the domain of privacy-preserving data mining (PPDM) is presented. Additionally, it introduces the application of metaheuristic algorithms in this domain to safeguard sensitive itemsets from disclosure within certain databases.

3. **Chapter 4: Hiding Sensitive Frequent Itemsets via Items Removal** - This chapter presents a new method that we propose for preserving the privacy of sensitive itemsets in transaction databases. This method employs a two-level multi-objective optimization approach to solve the problem. Namely, we use NSGA-II (Non-dominated Sorting Genetic Algorithm II) at the transaction level and the set covering problem (SCP) at the items level. The effectiveness of this method is demonstrated through extensive experiments conducted on various large transaction databases.

4. **Chapter 5: Privacy Preserving Sensitive Expected Frequent Itemsets in Uncertain Databases** - In this chapter, an extended version of the NSGAII4ID algorithm, specifically designed for uncertain databases, is introduced to safeguard the privacy of sensitive expected frequent itemsets.

5. **Chapter 6: Heuristic Approaches for Concealing Sensitive Frequent Itemsets in Uncertain Databases** - This chapter outlines three heuristic approaches tailored for PPDM in uncertain databases, namely: U-Aggregate, U-Disaggregate, and U-Hybrid approaches. The results of these methods, obtained through several experiments, are also presented.

6. **Chapter 7: Conclusion** - The final chapter offers a global summary of the primary contributions made in this thesis and explores potential avenues for future research.

Chapter 2: Pattern Mining

2.1 Introduction

The exponential growth in the volume of data generated and stored across various domains necessitates the analysis and extraction of valuable insights from this data. Raw data, in its unprocessed form, often lacks immediate significance, and a thorough analysis is required to unearth the concealed knowledge it holds. This need has given rise to the field of knowledge discovery in databases (KDD) [24]. KDD is primarily concerned with developing methods and techniques for comprehending data and identifying patterns, which play a crucial role in this context. Patterns can encompass sub-sequences, substructures, or itemsets that reflect various forms of regularity and homogeneity within the data [28]. Consequently, patterns reveal intrinsic and significant properties of databases.

Put differently, data mining and knowledge discovery in databases are two burgeoning research fields focused on the automated revelation of previously undiscovered patterns from extensive datasets. Data mining focuses on extracting non-trivial, novel, and potentially valuable knowledge from extensive databases. Sequential data mining techniques have found successful applications in various domains, including customer relationship management, web mining, science, engineering, and medicine. However, the need for distributed and parallel data mining techniques has become evident in recent years.

Research in data mining is concerned with extracting potentially valuable information from large datasets with diverse applications, such as customer relationship management, market basket analysis, and bio-informatics. This information can take the form of patterns, clusters, or classification models. For instance, association rules in a supermarket can reveal the relationships between items frequently purchased together. Customers can be grouped into segments

to improve customer relationship management, and classification models can be constructed based on customer profiles and shopping behavior to facilitate targeted marketing.

In this chapter, we will recall the basics of data mining in Section 2.2. Subsequently, in Section 2.3, we will delve into various techniques of pattern mining. In section 2.4, we focus on pattern mining methods including frequent itemset mining, sequence mining, episode mining. In section 2.4.4, we introduce the concept of mining frequent itemsets in uncertain databases and we briefly describe three main approaches to dealing with uncertainty in this context. We conclude the chapter in Section 2.5

2.2 Data mining

Data mining, which is part of the process of knowledge discovery in databases (KDD), aims at revealing new correlations, patterns, and trends through the analysis of extensive data sets. This is accomplished by employing pattern recognition technologies, alongside other statistical and mathematical techniques.

- Data mining involves the examination of extensive observational datasets to discover new relationships among them. The goal is to reformulate these relationships to enhance their usability for the data owners [29].

- Data Mining is an interdisciplinary field that uses both techniques for automatic learning, pattern recognition, statistics, databases and visualization to determine the ways Extraction of information from very large databases [30].

- Data Mining is an inductive, iterative and interactive process whose objective is to discovery of valid, new, useful and understandable data models in Large Databases [31]

Therefore, Data mining involves a series of methods used for data analysis to extract valuable information and reveal underlying patterns within extensive datasets. This field has experienced rapid growth and emergence in recent years, offering substantial benefits to numerous services and organizations. For example, in the telecommunications industry, data mining tools assist operators in analyzing consumer behaviors to better cater to customer interests and develop new service packages. Similarly, commercial banks leverage data mining to identify investment and deposit patterns, enabling them to make adjustments to their interest rates. In

Figure 2.1, we illustrate the Data-Information-Knowledge-Wisdom (DIKW) framework, where data serves as the fundamental element that can contain valuable information through the data mining process.,

Figure 2.1: Data Information Knowledge pyramid

Furthermore, data mining plays a pivotal role in deciphering complex data, extracting explicit information, and facilitating informed decision-making. Beyond its contributions to business development, data mining finds applications in everyday scenarios. For instance, systems or websites leverage data mining tools to recommend applications and content to users based on their Internet data access and sharing behaviors.

In the realm of the Internet of Things (IoT), data mining is instrumental in offering intelligent services across a wide range of fields, including healthcare, anomaly detection, security and safety assurance, surveillance, and more. This technology enables pervasive and intelligent solutions that enhance various aspects of daily life.

Knowledge Discovery in Databases (KDD) is the comprehensive process of uncovering valuable insights within data, with a strong emphasis on the practical application of specific data mining methods. KDD is a multidisciplinary field that holds relevance for researchers in machine learning, pattern recognition, databases, statistics, artificial intelligence, knowledge acquisition for expert systems, and data visualization.

The overarching goal of the KDD process is to extract knowledge from data within the context of large databases. This is achieved through the utilization of data mining methods, including algorithms, to identify what qualifies as knowledge based on predefined measures and thresholds. The process involves working with a database, which may require pre-processing, sub-sampling, and various transformations. In Figure 2.2, we illustrate the KDD process, where the database undergoes several stages to extract knowledge, with data mining as the central step for evaluating and uncovering patterns. The KDD process involves the following tasks:

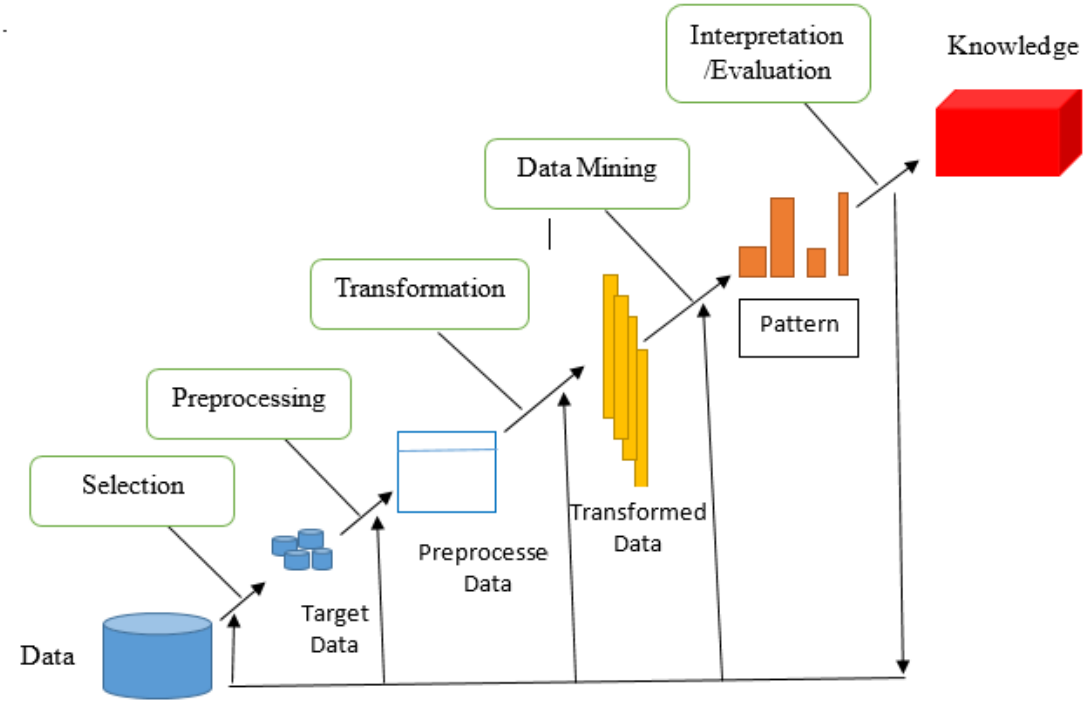

Figure 2.2: KDD process

- *Developing an Understanding:* Application Domain, relevant prior knowledge, goals of the End-User.
- *Creating a target dataset:* Selecting a data set or focusing on a subset of variables or data samples for discovery.
- *Data cleaning and pre-processing:* Removal of noise or outliers, collecting necessary information to model or account for noise, strategies for handling missing data.
- *Data reduction and projection:* Finding useful features to represent the data based on the task's goal, using dimensionality reduction or transformation methods to reduce variables or find invariant representations.

- *Choosing the data mining task:* Deciding whether the goal of the KDD process is classification, regression, clustering, etc.
- *Choosing data mining algorithm(s):* Selecting method(s) for searching patterns in data, deciding on models and parameters, matching a specific data mining method with the overall criteria of the KDD process.
- *Data Mining:* Searching for patterns of interest in a representational form (e.g., classification rules, trees, regression, clustering).
- *Interpreting mined patterns.*
- *Consolidating discovered knowledge.*

Since the mid-1980s, the field of database technology has undergone remarkable advancements, primarily fueled by the widespread adoption of relational technology and extensive research and development endeavors to create robust database systems. These developments have paved the way for sophisticated data models, including the extended relational model, object-oriented, object-relational, and deductive database systems. Furthermore, there has been a growing emphasis on distribution, diversification, and data sharing within the field.

The emergence of heterogeneous database systems and internet-based information systems, such as the World Wide Web (WWW), has played a pivotal role in the dissemination and management of information. These innovations have collectively transformed the landscape of database technology and data management.

Furthermore, data mining can be viewed as a natural progression in information technology. Starting from the 1960s, information and fundamental technology data systems have undergone systematic evolution, transitioning from basic file management to more robust and sophisticated databases. Ongoing research and development in the field of database systems since the 1970s has yielded significant advances, including the development of relational database systems, modeling tools, and techniques for efficient data indexing and organization.

2.3 Data mining techniques

Data mining techniques can be broadly categorized into two main forms: supervised (also referred to as predictive or directed) and unsupervised (also known as descriptive or undirected). Both of these categories encompass functions capable of uncovering various hidden patterns within large datasets.

While data analytics tools are increasingly emphasizing self-service capabilities, it remains valuable to understand which data mining approach suits your specific needs before embarking on a data mining operation.

2.3.1 Supervised data mining techniques

In supervised learning, the objective is to predict a specific target value related to data. These targets can encompass two or more possible outcomes, or even represent a continuous numeric value. To utilize these methods, it's ideal to possess a subset of data points for which the target value is already known. This data is used to construct a model that characterizes a typical data point in the context of its various target values. Subsequently, this model is applied to data where the target value is currently unknown. The algorithm's task is to identify the 'new' data points that best match the model of each target value.

2.3.1.1 Classification

Classification is one of the most commonly applied data mining techniques. It involves using a set of pre-classified examples to build a model capable of classifying a larger population of records. This technique is particularly well-suited for applications such as fraud detection and credit risk analysis. In practice, classification often relies on decision tree or neural network-based algorithms.

The data classification process consists of two key phases: learning and classification. During the learning phase, the classification algorithm analyzes the training data. In the classification phase, test data are used to assess the accuracy of the classification rules. If the accuracy meets acceptable standards, these rules can then be applied to new data tuples.

For instance, in a fraud detection application, this process would involve evaluating com-

plete records of both fraudulent and valid activities on a record-by-record basis. The classifier-training algorithm uses these pre-classified examples to determine the necessary parameters for accurate discrimination. Subsequently, these parameters are encoded into a model referred to as a classifier. Among the main classification methods, we can cite: Classification by decision trees, Bayesian classification, Neural Networks, Support Vector Machines (SVM), Classification Based on Associations, etc.

2.3.1.2 Prediction

Regression techniques are versatile tools that can be adapted for prediction in data mining. Regression analysis allows us to model the relationship between one or more independent variables and dependent variables. In the context of data mining, the independent variables represent attributes that are already known, while the response variables are what we aim to predict. In practice, many real-world problems involve more than simple prediction. Sales volumes, stock prices, and product failure rates, for example, can be challenging to predict due to complex interactions among multiple predictor variables. Therefore, the forecasting of future values may require more intricate techniques, such as logistic regression, decision trees, or neural networks.

Notably, some model types can serve both regression and classification purposes. For instance, the CART (Classification and Regression Trees) algorithm can be employed to create both classification trees (for categorical response variables) and regression trees (for continuous response variables). Similarly, neural networks can be used to build models for both classification and regression tasks. These are some regression methods: Linear regression, multivariate linear regression, nonlinear regression and multivariate nonlinear regression.

2.3.2 Unsupervised data mining techniques

Does not focus on predetermined attributes, nor does it predict a target value. Rather, unsupervised data mining finds hidden structure and relation among data.

2.3.2.1 Clustering

Clustering can be described as the process of identifying similar classes of objects. Through clustering techniques, we can discern dense and sparse regions within the object space, un-

cover overall distribution patterns, and explore correlations among data attributes. While the classification approach is also effective in distinguishing groups or classes of objects, it can be computationally expensive. In such cases, clustering can serve as a valuable pre-processing step for attribute subset selection and subsequent classification.

For instance, clustering can be employed to group customers based on their purchasing patterns, or to categorize genes with similar functionality. It provides a fundamental step in the analysis and organization of data, facilitating the exploration of underlying patterns and structures. Several clustering methods have been proposed, including: Partitioning Methods, hierarchical agglomerative (divisive) methods, density-based methods, grid-based methods and model-based methods.

2.3.2.2 Association rule mining

Association and correlation techniques are typically employed to discover frequent item sets within extensive datasets. These findings play a crucial role in assisting businesses with various decisions, including catalogue design, cross-marketing strategies, and the analysis of customer shopping behaviors. Association rule algorithms aim to generate rules with confidence values less than one. However, it is important to note that the number of possible association rules for a given dataset is often extremely large, and a significant proportion of these rules may have limited, if any, practical value. Different variants of association rules have been studied like: Multi-level, multi-dimensional and quantitative association rules.

2.4 Pattern mining

Pattern mining involves the discovery of interesting, valuable, and unexpected patterns within databases. This field of research gained prominence, as evidenced by [32], with the seminal paper by [33]. Pattern mining techniques hold significant appeal due to their capacity to unveil concealed patterns in extensive databases, which are not only interpretable by humans but also instrumental for understanding data and making informed decisions. For instance, a pattern like milk; chocolate cookies can provide insights into customer behavior and inform strategic decisions such as product co-promotions and discount offerings to boost sales.

While pattern mining has gained widespread popularity due to its applications across nu-

merous domains, it's important to note that many of these techniques, such as those for frequent itemset mining [33, 34, 35, 36, 37], and association rule mining [23], are primarily focused on analyzing data where the sequential ordering of events is not considered.

Since its inception [22], the task of pattern mining has generated considerable interest across various application fields, as evident in studies like [38], [39]. This growing interest has led to the definition of new types of patterns to meet the specific analysis needs of experts in different application domains.

2.4.1 Frequent itemset mining

One of the most popular tasks in knowledge discovery and data mining is mining frequent itemsets, which involves identifying frequently occurring combinations of items as well as the stong associations between frequent itemsets (association rules) from traditional databases where the transaction content, or items, is known with certainty. The extensive research dedicated to these tasks has resulted in the development of several advanced and efficient algorithms for discovering frequent itemsets. Some of the most well-known methods include Apriori, Eclat, and FP-Growth (Frequent Pattern Growth).

Frequent itemset mining (FIM) is a data analysis method originally designed for market basket analysis. It seeks to uncover purchasing patterns in the behavior of supermarket, mail-order company, and online shop customers. Specifically, it aims to identify sets of products that are frequently bought together. However, finding such patterns is not straightforward due to the exponential increase in computational complexity with the number of items in the data, as well as the substantial memory consumption required in the mining process. Therefore, the development of highly efficient solutions becomes essential. Figure 2.3 illustrates the various steps involved in frequent itemset mining from transactional databases.

The FIM (Frequent Itemset Mining) problem is considered the foundation of the pattern mining field [40], encompassing multiple tasks aimed at extracting itemsets in various forms and for different purposes [28]. A straightforward variation of FIM involves extracting itemsets that infrequently appear in data or were discarded due to the antimonotone property of support in FIM. This led to the definition of pattern mining as the task of mining sets of items that frequently or infrequently appear in data [41, 42, 43].

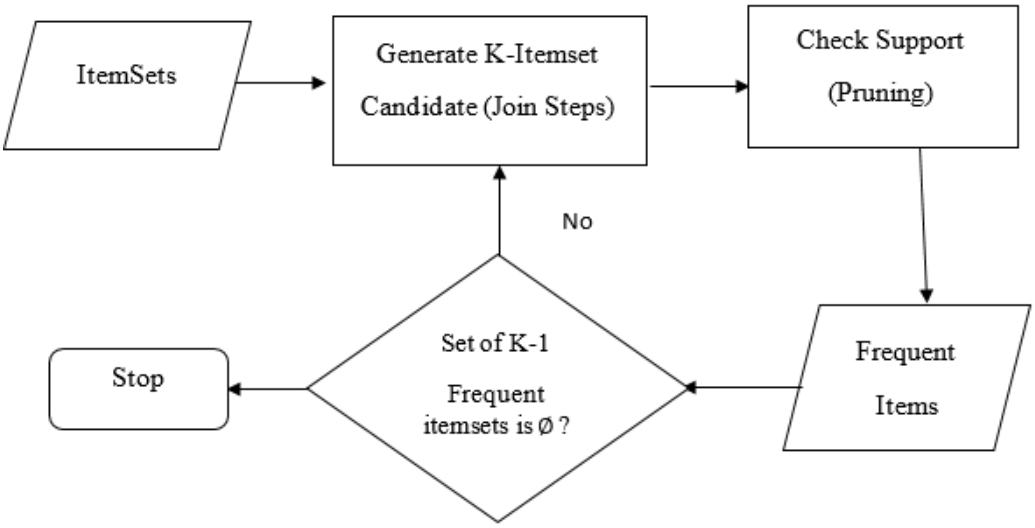

Figure 2.3: Mining frequent itemset

The significant interest in frequent itemset mining algorithms is primarily due to the computational challenges involved in the task. Even for moderately sized databases, the search space of FIM grows exponentially with the length of transactions, creating challenges for itemset generation when support levels are low. In practical scenarios, support levels for mining corresponding itemsets are often constrained by memory and computational limitations. Therefore, efficient space- and time-effective analysis is crucial. In the early years of research in this area, the primary focus was on developing FIM algorithms with improved computational efficiency.

Frequent Itemset Mining (FIM) finds applications in various domains, including spatio temporal data analysis, biological data analysis, and software bug detection [28]. It serves as a fundamental step in uncovering recurring patterns within a database, facilitating the generation of association rules for data analysis.

Frequent itemsets play a pivotal role in many data mining tasks, such as discovering association rules, correlations, sequences, classifiers, clusters, and more. Among these, the mining of association rules stands out as one of the most popular problems. The original motivation for searching association rules arose from the need to analyze supermarket transaction data, delving into customer behavior regarding purchased products. Association rules reveal how often items are bought together. For example, an association rule like "juice, chips (80%)" indicates that four out of five customers who purchased juice also bought chips. These rules find practical use in decisions regarding product pricing, promotions, store layout, and various other aspects.

Furthermore, FIM is a broad area of research, encompassing a wide range of topics, especially from an application-specific perspective. It has led to the development of numerous variations in frequent itemset mining tailored for advanced data types. These variations have been employed in a diverse set of tasks. Moreover, various data domains, such as graph data, tree-structured data, and streaming data, often demand specialized algorithms for frequent itemset mining. The issue of the interestingness of itemsets is also relevant in this context.

In general, algorithms for Frequent Itemset Mining (FIM) can be categorized into three main groups [28]: Join-Based, Tree-Based, and Pattern Growth, as depicted in Fig. 2.4. Join-Based algorithms employ a bottom-up approach to identify frequent itemsets in a database and expand them into larger itemsets as long as they meet a minimum threshold set by the user. Tree-Based algorithms utilize set-enumeration concepts to generate frequent itemsets. They construct a lexicographic tree that facilitates various mining methods, such as breadth-first or depth-first exploration. Pattern Growth algorithms implement a divide-and-conquer approach. They partition and project databases based on presently identified frequent patterns and expand them into longer patterns within the projected databases.

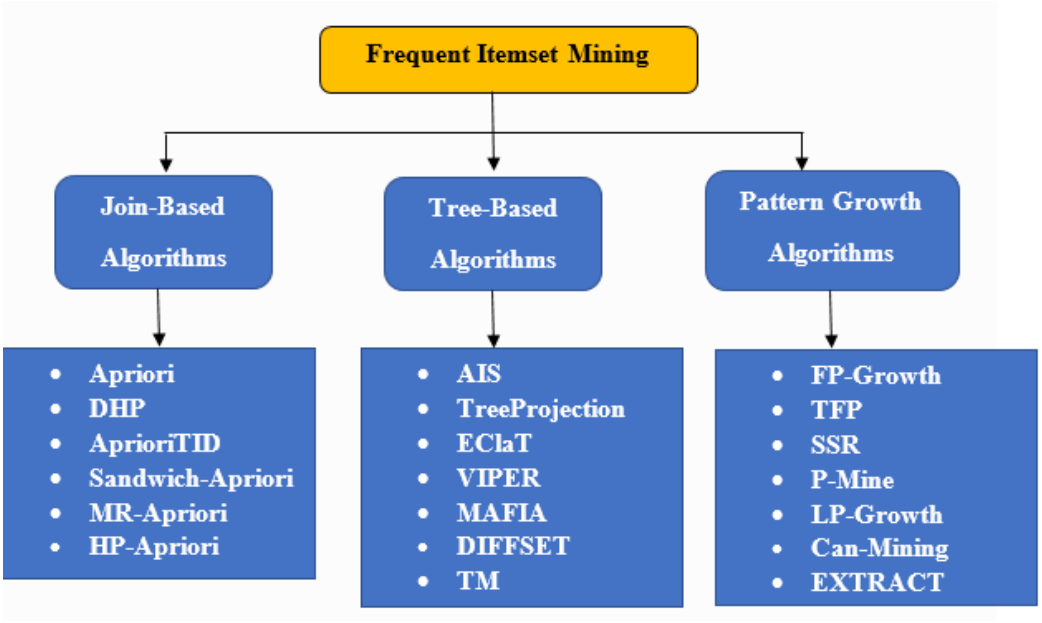

Figure 2.4: Classification of Frequent Pattern Mining algorithms

The Apriori algorithm is one of the first algorithms developed for frequent itemset mining. It was introduced in the early 1990s [22] and later refined in [23] and came to be known simply as Apriori. The Apriori algorithm follows a sequence of steps to identify the most frequent itemset within a given database. This data mining technique iteratively follows the 'join' and

Table 2.1: A traditional database

TID	List of item_IDs
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I5
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

'prune' steps until the most frequent itemset is discovered. The problem typically involves a minimum support threshold defined by the user or given as input. In the 'join' step, the algorithm generates (K+1)-itemsets from K-itemsets by pairing each item with itself. The 'prune' step involves scanning the count of each item in the database. Any candidate item that does not meet the minimum support is considered infrequent and is removed. This pruning step helps reduce the size of candidate itemsets.

A sample of transactional data, comprising product items purchased in different transactions, is illustrated in Table 2.1. In the first step of the Apriori algorithm, the database is scanned to identify all the frequent 1-itemsets by counting each of them and capturing those that meet the minimum support threshold. This process involves scanning the entire database to identify each frequent itemset. The minimum support threshold used, as shown in Fig. 2.5, is set at 2. Consequently, only the records with a minimum support count of 2 or more will proceed to the next cycle of algorithm processing.

Apriori algorithm offers significant performance improvements by effectively reducing the size of candidate itemsets in many cases. However, it still faces two critical limitations [44]. First, when the total count of a frequent k-itemsets increases, a substantial number of candidate itemsets may still need to be generated. Second, the algorithm requires repeated scanning of the entire database and verification of a large set of candidate items using the pattern matching technique.

Subsequent algorithms, like FP-Growth [24] and Eclat [45], took a different approach. FP-Growth uses a depth-first search procedure based on a prefix-tree-based main memory compressed representation of the input dataset. Eclat, on the other hand, employs a vertical trans-

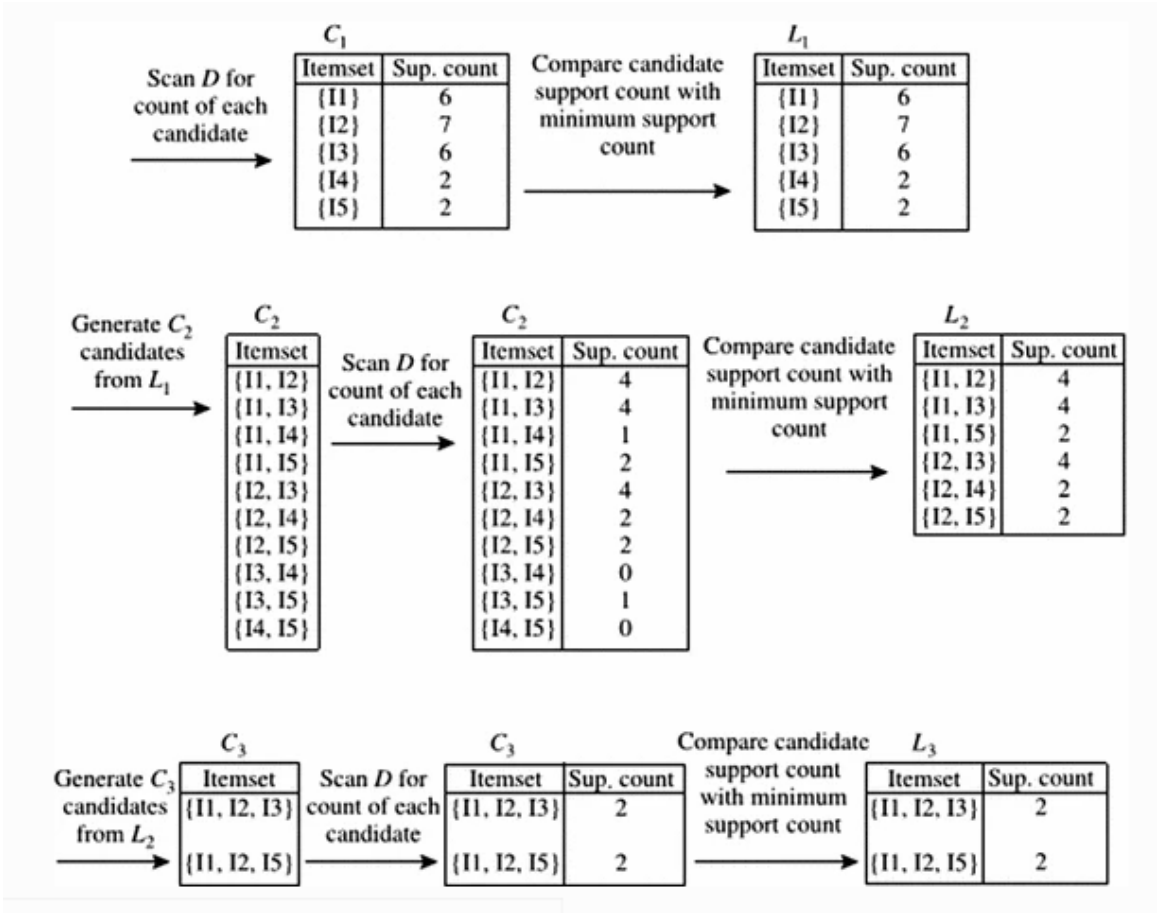

Figure 2.5: Generation of candidate itemsets and frequent itemsets

position of the dataset to work with transaction identifiers [40]. Considerable attention has been dedicated to assessing the performance of these novel algorithms in the field [46], [47]. In recent years, extremely large databases can be analyzed within seconds, thanks not only to advancements in hardware and architectures but also to the proposed algorithmic solutions.

Efficient algorithms can now be found in the literature, addressing various issues related to frequent itemset mining, including memory requirements, exponential time complexity, data dimensionality, and search space pruning [28]. In many situations, such algorithms are sufficiently capable of tackling these problems.

A significant challenge in frequent itemset mining is that the rules discovered may not always be very interesting when using quantification metrics such as support and confidence. This is because such quantification does not normalize for the original frequency of the underlying items. For instance, an item that rarely appears in the underlying database is also likely to appear in itemsets with lower frequency. Therefore, the absolute frequency often provides

limited insight into the likelihood of items co-occurring due to biases associated with individual item frequencies.

2.4.2 Sequence mining

Sequential pattern analysis is a data mining technique that focuses on discovering similar itemsets or patterns within transactional data over a defined business period. These patterns are subsequently leveraged for further business analysis to uncover relationships within the data.

Sequential itemset mining, a key component of sequential pattern analysis, entails identifying sub-sequences that appear in a sequence database with a frequency equal to or exceeding a user-specified threshold. A sequence database comprises a collection of records, each representing an ordered sequence of events, with or without explicit timestamps. These sequences can range from retail customer transactions to DNA sequences and web log data. For example, if a sub-sequence, such as a customer purchasing a PC, followed by a digital camera and then a memory card, occurs frequently in a customer transaction database, it is considered a (frequent) sequential pattern.

Sequential itemset mining has gained prominence as a distinct theme in data mining research. In recent years, several surveys on sequential itemset mining have been published, providing valuable resources [48, 49, 50, 51, 52].

Two common types of sequential data used in data mining are time-series and sequences [53]. Time-series represent ordered lists of numerical values, such as stock prices, temperature readings, and electricity consumption, as shown in Fig. 2.6 (left). On the other hand, sequences represent ordered lists of nominal values or symbols, as depicted in Fig. 2.6 (right). These sequences find applications in various domains, such as representing sentences in texts, items purchased by customers in retail stores, and web pages visited by users.

Sequential itemset mining aims to reveal relationships between sequential events by identifying specific orderings of occurrences. It involves the discovery of frequently occurring sequences, which can be used to describe the data, predict future data, or uncover periodic patterns. A sequential pattern is defined as a sequence of itemsets that frequently occur in a specific order, with all items within the same itemsets having the same transaction-time value or falling within a specified time gap [54].

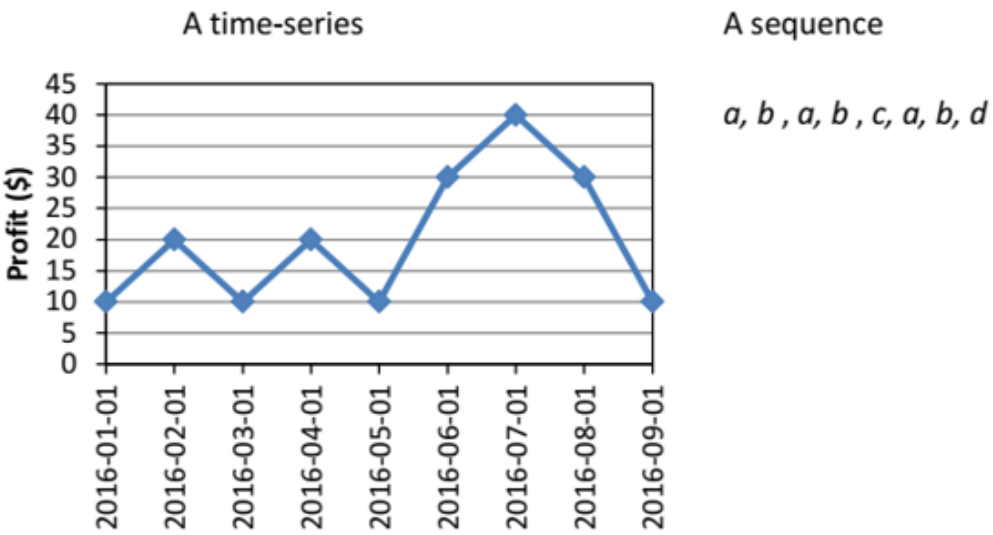

Figure 2.6: A time-series (left) and a sequence (right)

Table 2.2: A transaction database

TID	transactions
10	a, b, d
20	a, c, d
30	a, d, e
40	b, e, f

Table 2.2 shows an example of a transaction database, when each transaction has a subset of items, dislike, in table 2.3, a sequence database consists of order of elements or events.

In table 2.3, A database of sequences, where each sequence consists of a list of transactions ordered by transaction time and each transaction is a set of items, sequential pattern mining is to discover all sequential patterns with a user-specified minimum support, where the support of a pattern is the number of data sequences that contain the pattern.

Table 2.3: A Sequence database

SID	sequences
10	$\langle a\ (a\ b\ c)\ (a\ c)\ d\ (c\ f)\rangle$
20	$\langle (a\ d)\ c\ (b\ c)(a\ e)\rangle$
30	$\langle (e\ f)\ (a\ b)\ (d\ f)\ c\ b\rangle$
40	$\langle a\ (a\ b\ c)\ (a\ c)\ d\ (c\ f)\rangle$

Given a set of sequences and support threshold, find the complete set of frequent sub sequences A sequence $\langle (e\ f)\ (a\ b)\ (d\ f)\ c\ b \rangle$. An element may contain a set of items. Items within an element are unordered and we list them alphabetically. $\langle a\ (b\ c)\ d\ c \rangle$ is a sub-sequence of $\langle a\ (a\ b\ c)\ (a\ c)\ d\ (c\ f) \rangle$. Given support threshold $minsup = 2$, $\langle (a\ b)\ c \rangle$ is a sequential pattern.

Sequential itemset mining is a subfield of data mining that focuses on extracting frequent and meaningful patterns from sequential data. These patterns can provide valuable insights into the temporal relationships and dependencies present in the data. Here are some additional details about sequential itemset mining:

1. Types of Sequential Patterns: Sequential itemset mining can identify different types of patterns, including sequential patterns, episode patterns, periodic patterns, and closed sequential patterns. Each type represents different characteristics and temporal dependencies within the data.

2. Algorithms: Various algorithms have been developed for sequential itemset mining, such as Apriori-based algorithms, GSP (Generalized Sequential Pattern) algorithm, PrefixSpan, SPADE (Sequential Pattern Discovery using Equivalence classes), and many more. These algorithms employ different strategies to efficiently discover frequent patterns in large sequence databases[55].

3. Support Measure: The support measure is used to determine the frequency of a pattern in the dataset. It represents the proportion of sequences in the database that contain the given pattern. A user-defined minimum support threshold is typically set to filter out infrequent patterns and focus on more meaningful and significant patterns[24].

4. Applications: Sequential itemset mining has numerous applications in various domains. It can be applied to analyze customer purchasing behavior, website browsing patterns, DNA sequencing, process mining, analyzing sensor data in Internet of Things (IoT) applications, and many more. By discovering frequent patterns, it enables businesses to make data-driven decisions, optimize processes, improve recommendations, and understand temporal dynamics in different contexts [56],[24].

5. Challenges: Sequential itemset mining poses several challenges, including scalability, handling large and high-dimensional datasets, dealing with noise and variations in se-

quences, managing variable sequence lengths, and incorporating constraints on patterns such as time gaps, item constraints, and length constraints.

Overall, sequential itemset mining plays a vital role in extracting meaningful patterns from sequential data, enabling valuable insights and decision-making in various domains.

2.4.3 Episode mining

An episode comprises a collection of events or states that co-occur within a designated time window. Frequent episode mining is focused on identifying these repeated episodes that surpass a predetermined threshold of occurrence frequency.

The mining process of historical data can be time-intensive, lasting for hours or even days. Most existing Frequent Episode Mining (FEM) solutions face two key characteristics that contribute to this time consumption:

1. The anti-monotonicity property, crucial for episode frequency, may not hold true [57]. This means that the frequency of a sub-episode might be lower than that of the super-episode, particularly if a minimal occurrence is used to determine episode frequency.

2. Testing whether an episode occurs in a sequence poses an NP-complete problem [58]. This complexity contributes to the time-consuming nature of the mining process.

Episode mining is focused on uncovering noteworthy patterns within lengthy sequences of symbols or events. Sequence data, prevalent across various domains, encapsulates information such as a series of alarms generated by a computer system, a sequence of words within a text, or even a sequence of purchases made by a customer. Through episode mining, patterns within these sequences can be unveiled, aiding in the comprehension of data and offering support for decision-making processes. A visual representation of this sequence is shown in figure 2.7 the running example of event sequence :

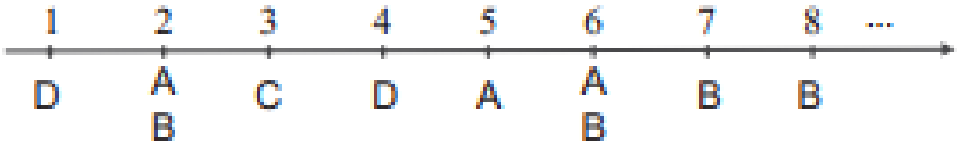

Figure 2.7: The running example of event sequence

Therefore, Frequent episode mining serves as a widely adopted framework for unveiling sequential patterns from sequence data. Typically, prior studies in this domain handle data offline in a batch mode. However, in scenarios with rapidly evolving sequence data, previously discovered episodes might become outdated as new, valuable episodes continually emerge. The goal of episode mining is to find frequent episodes. An episode E is a sequence of event sets of the form $E = \langle X_1, X_2, \ldots, X_p \rangle$. This notation indicates that a set of events X_1 occurred, was followed by another set of events X_2, and so on, and finally followed by a set of events X_p.

Figure 2.7 shows an event sequence:

$$\vec{S} = \langle (\{D\},1), (\{A,B\},2), (\{C\},3), (\{D\},4), (\{A\},5), (\{A,B\},6), (\{B\},7), (\{B\},8) \rangle.$$

An episode (also known as a serial episode) α is defined as a non-empty totally ordered set of events of the form $e_1 \rightarrow \ldots e_j \rightarrow \ldots e_k$ where $e_i \in E$ for all $i \in [1,k]$ and the event e_i occurs before the event e_j for all $1 \leq i \leq j \leq k$. The length of an episode is defined as the number of events in the episode. An episode α of length k is called a $k-episode$. For example, $\alpha = D \rightarrow A \rightarrow C$ is a $3-episode$. An event A can also be viewed as a $1-episode$.

Frequent Episode Mining (FEM) techniques are widely applied in various domains such as telecommunication [59, 60], manufacturing [61, 62], finance [63, 64], biology [65, 66], system log analysis [67, 64], and news analysis [68].

In this context, an episode, also known as a serial episode, is typically defined as a completely ordered set of events. The frequency of an episode serves as a metric that measures how frequently it occurs within a sequence. FEM is geared towards identifying all the frequent episodes whose frequencies exceed a user-defined threshold.

The original definition of mining frequent episodes was set within a context of "a sequence of events," sampled regularly, as proposed by [61]. In a broader sense, an episode is an assembly of events that occur together, typically in a partially ordered manner [69].

2.4.4 Mining frequent itemsets over uncertain transaction databases

The identification of frequent itemsets within traditional transactional databases, where transaction content is completely known and certain, stands as a significant task in knowledge discovery and data mining. However, the landscape has evolved in recent years, with

the prevalence of uncertain data due to inherent imprecision in modern data collection technologies. Frequent pattern mining holds a pivotal role in various real-world applications, spanning banking, bio-informatics, environmental modeling, epidemiology, finance, marketing, medical diagnosis, and meteorological data analysis [11]. Many of these applications grapple with uncertainty rooted in diverse factors: limitations in human perception or understanding, constraints in observation tools, and restricted resources for data collection, storage, transformation, or analysis. Additionally, inherent biases contribute to uncertainty in the data. Sensors used in environmental surveillance, security, and manufacturing systems, encompassing acoustic, chemical, electromagnetic, mechanical, optical radiation, and thermal sensors, can introduce noise and uncertainty [11]. Notably, techniques for mining frequent itemsets in uncertain databases significantly diverge from traditional methods applied in certain transaction databases. Researchers have adapted conventional techniques like Apriori and tree mining to suit the requirements of uncertain transaction databases [11].

Data uncertainty is notably prevalent in various contexts, notably exemplified within the medical domain. In medical datasets, tuples encapsulate diverse symptoms or illness indicators observed in individual patients. However, these values may not be entirely certain; instead, they are often associated with a confidence degree, typically represented by a probability value. Additionally, satellite imagery analysis offers another example. Image processing techniques are employed to extract features indicating the presence or absence of specific target objects, such as identifying bunkers, yet with a certain level of uncertainty inherent in the analysis.

The uncertainty inherent in spatial data, arising from noise and limited resolution, is frequently quantified and expressed through probability measures [70, 11]. This has notably spurred active research in the field of data mining involving uncertain data. For instance, [71] introduced efficient clustering algorithms specifically crafted for uncertain objects. Moreover, research such as that by [72, 73] has explored naive Bayes and decision tree classifiers tailored explicitly for uncertain data. For a comprehensive overview of techniques in mining uncertain data, [74] provides an extensive resource.

Mining frequent itemsets within uncertain databases presents the challenge of evaluating the confidence in the resulting mining outcomes. These uncertain database models commonly categorize into two types: attribute uncertainty, where each item in a transaction is associated with an existential probability, and tuple uncertainty, where each tuple is linked to a probabil-

ity signifying its existence. While the research in this field has been somewhat limited, certain strategies for pruning have been suggested to expedite algorithms, particularly in uncertain scenarios. Yet, an unexplored area in this domain involves dealing with data uncertainty within privacy-preserving pattern mining. This necessitates hiding sensitive data while extracting frequent patterns from uncertain databases.

The representation of uncertain data relies significantly on the model employed to encapsulate its complexity. Possible worlds, as proposed by [48], serves as a comprehensive model for uncertain data, striving to encompass all conceivable database states in line with a specified schema. Handling uncertain data is notably more intricate compared to traditional databases, demanding a representation of uncertainty that is easily processed and queried. Information with lower uncertainty generally holds greater importance than information with higher uncertainty. However, calculating the complexity of these algorithms frequently involves a computation cost of at least O (N log N) for each itemset.

The investigation of mining frequent itemsets within uncertain databases has become a central focus within data mining communities, signifying a fundamental challenge in the field. Numerous researchers have dedicated their efforts to identify the most effective strategies for mining frequent itemsets within uncertain databases.

The algorithms for mining frequent itemsets are commonly classified into three primary groups: Expected support-based frequent itemsets, Exact probabilistic frequent algorithms, and Approximate probabilistic frequent algorithms [19].

2.4.4.1 Expected support-based frequent itemset mining algorithms

Given an uncertain database D with N transactions and a minimum expected support ratio $minesup$, an itemset X is an expected support-based frequent itemset if and only if $ExpSup(X)/|D| \geq minesup$ where $ExpSup(X)$ is the expected support of X given by the following formula:

$$ExpSup(X) = \sum_{t \in T} Pr(X \subseteq t)/|D| = \sum_{t \in T} (\prod_{x \in X} Pr(x \in t))/|D| \tag{2.1}$$

The complexity of computing the expected support-based frequent itemset is $O(N)$. Among the main expected support-based frequent itemsets algorithms proposed in the literature, we

can cite : U-Apriori[16], UFP-Growth[17] and UH-Mine algorithm [11] that extend the well-known APriori, FP-Growth and H-Mine algorithms respectively to the case of probabilistic transactional databases.

2.4.4.2 Exact Probabilistic Frequent itemsets algorithms

The representation of uncertain data relies on the model used for capturing the nature of complexity. The possible worlds [75] is the general model for uncertain data, which tries to capture all the possible states of a database which are consistent with a given schema. Furthermore, the process of managing uncertain data is much more complicated than that for traditional databases, that is why the uncertainty information needs to be represented in a form which is easy to process and query. The complexity computation of these algorithms requires spending at least $O(N\log^2 N)$ computation cost for each itemset.

In the exact probabilistic frequent itemset mining algorithm, an itemset X is frequent if and only if the probability that the support of X is more than or equal *minsup* is greater or equal to a probability threshold *minprob*.

An algorithm based on dynamic programming is proposed in [15] to calculate exact probabilistic frequent itemsets. Another exact algorithm based on the divide and conquer (DC) principle is proposed in [18]. Its main idea is to divide uncertain database into two sub-databases: $UDB1$ and $UDB2$. After that, DC algorithm calls itself recursively to divide the database until only one transaction remains. Finally, the complete probability distribution of itemset support is obtained when the algorithm finishes.

2.4.4.3 Approximate approach for mining probabilistic frequent itemsets

The support of an itemset is considered as a random variable following Poisson-Binomial distribution. Thus, the random variable or the support of an itemset can be approximated by the Poisson or Normal distribution effectively when uncertain databases are large enough. The computation cost is $O(N)$, and return the complete probability information when uncertain database is huge. Three approximate algorithms was designed for mining frequent probabilistic itemsets over uncertain database: The Poisson distribution based U-Apriori called PDU-Apriori algorithm proposed in [16]. Besides, according to the Lyapunov Central Limit Theory (see e.g. [76]), Poisson-Binomial distribution converges to the Normal distribution with high

probability [77]. Hence, based on this theorem, the Normal Distribution-based U-Apriori called NDU-Apriori algorithm is proposed in [13]. Likewise, the Normal distribution-based UH-Mine algorithm is proposed in [78]. The UH-Mine algorithm is in these steps:

1. Scan the uncertain database and discover all expected support-based frequent itemsets.
2. Build a head table that contains all expected support-based frequent itemsets.
3. Insert all transactions in to the data structure, UH-Struct.
4. Use the depth-first strategy to build the head table.
5. The algorithm recursively builds the head tables where different itemsets are prefix.

2.5 Conclusion

The field of data mining is a powerful methodology and technology that plays a crucial role in generating information essential for decision-making. As future developments unfold, data mining is expected to become even more powerful and useful. However, with this advancement comes growing concern over individual privacy and the protection of sensitive data. The increasing volume of personal data raises alarms about potential misuse without proper consent.

While data mining effectively extracts valuable insights, it also risks intruding on individual privacy by revealing implicit private details that individuals may not willingly disclose. Addressing these concerns is critical, making the integration of privacy-preserving features into the data mining process essential. The next chapter will explore the domain of privacy-preserving data mining, discussing its core concepts that not only address these issues but also engage researchers in the field.

Chapter 3: Privacy Preserving Data Mining

3.1 Introduction

Government agencies, businesses, and non-profit organizations actively seek ways to collect, analyze, and utilize data related to individuals, households, or enterprises. This effort enhances their capabilities for immediate and long-term planning. The increased capacity for data storage and improved data accessibility has led to the widespread adoption of data mining methods. These techniques serve as a means to extract valuable insights across various sectors such as marketing, medical diagnosis, weather forecasting, and national security. However, exploring knowledge from specific data types may pose challenges that demand careful attention to safeguard the privacy rights of data owners. Data mining results take various forms, from patterns and clusters to classification models. For instance, in a retail context, association rules can reveal relationships between concurrently purchased items.

In recent years, data mining has evolved into an essential technology, primarily due to its capability to uncover concealed insights, identify patterns, and detect trends within vast data repositories. This versatile technique offers numerous advantages and finds applications across diverse domains, including businesses, marketing, healthcare, manufacturing, sales, scientific research, and technology innovation. However, despite its merits, data mining introduces a significant concern: the potential jeopardy it poses to data privacy. Through data mining methods, sensitive information, including personal details and patterns, can be deduced from seemingly non-sensitive or unclassified data, potentially encroaching upon individual privacy. Safeguarding such information is crucial, and all data mining activities should prioritize secure execu-

tion. This scenario has prompted an urgent need for the development of appropriate privacy-preserving data mining (PPDM) techniques.

The widespread practice of sharing and publishing data presents numerous opportunities. However, the processes involved in data collection and dissemination can inadvertently lead to information disclosure, posing a significant threat to privacy. The powerful capabilities of data mining tools to extract hidden insights from extensive data collections have encouraged increased data gathering efforts by both companies and government agencies, thereby raising concerns about privacy. In response to these concerns, data mining researchers have developed specialized techniques focused on safeguarding privacy, forming the field of privacy-preserving data mining (PPDM). Many organizations gather data about individuals for specific purposes but must ensure the preservation of individual privacy and the confidentiality of sensitive data. Hence, a diverse array of PPDM techniques becomes indispensable.

The reminder of the chapter is structured into three sections. Section 3.2 furnishes a definition of privacy-preserving data mining. Section 3.3 presents a review of privacy-preserving algorithms while Section 3.4 focuses on metaheuristic-based algorithms for PPDM. Finally, the conclusions are summarized in Section 3.5.

3.2 Privacy preserving data mining

Privacy Preserving Data Mining (PPDM) has been a significant area of research since 1991, finding applications in both public and private sectors. It encompasses data mining techniques specifically designed to protect sensitive information from unauthorized disclosure. The primary aim is to extract valuable knowledge from extensive databases while upholding privacy principles. Many privacy computation methods rely on data transformation. However, these techniques often result in a reduction in representation granularity, striking a delicate balance between information preservation and privacy. This reduction in granularity may potentially compromise the effectiveness of data management and mining algorithms.

In recent years, PPDM has gained significant importance [79, 80, 81, 82, 83, 84] in hiding sensitive information while enabling the extraction of useful knowledge. PPDM techniques find application in modern fields such as cloud computing [85, 86], Internet of Things [87], blockchain technology [88], and deep learning [86, 84]. This article provides a concise review

of recent work in three main domains: classification, data exchange within heterogeneous and interconnected sources, and mining interesting patterns in databases. For a more comprehensive classification and evaluation of PPDM methods, see, for example, [89].

The primary goal of privacy-preserving data mining is to develop algorithms that modify the original data in a manner that ensures the protection of private data and knowledge even after the mining process. Various approaches have been proposed in the literature, differing in their assumptions about data collection models and user privacy requirements. The aim of PPDM is to sanitize a database by concealing and securing personal, confidential, or sensitive information of participants while still allowing for data analysis.

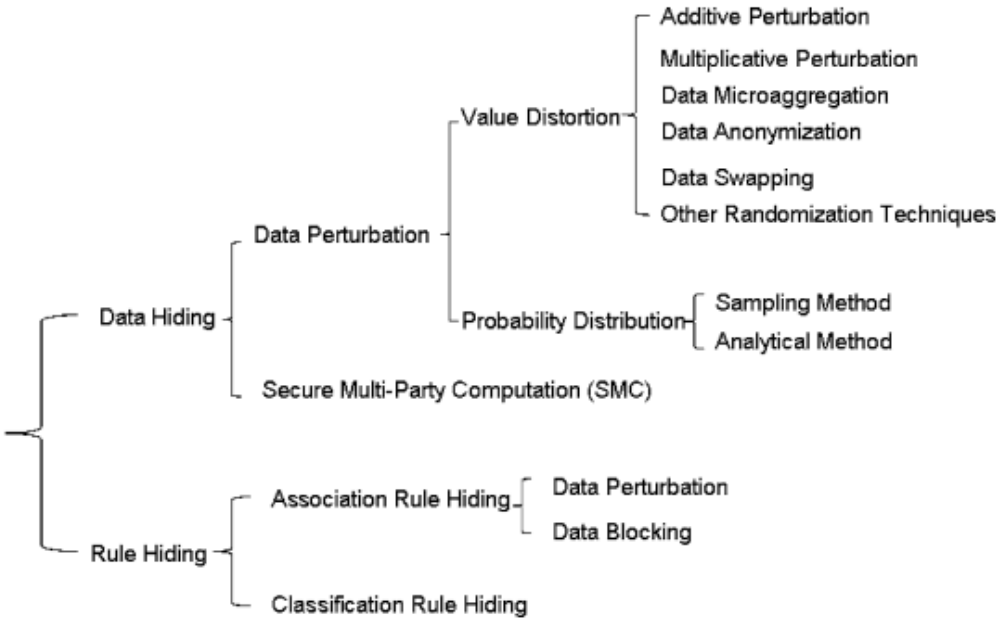

Figure 3.1: A brief overview of PPDM techniques

PPDM techniques are divided into several categories that are based on different assumptions or domain knowledge which are categorized in [90], and shown in Figure 3.1.

Another approach within Privacy-Preserving Data Mining (PPDM) involves the use of cryptographic tools to construct data-mining models. This approach treats PPDM as a specialized form of secure multi-party computation (SMC), focusing not only on safeguarding individual privacy but also on preventing any information leakage aside from the final outcome. However, this approach comes with the drawback of significant increases in communication and computation costs as the number of involved parties grows.

Rule hiding represents a technique that transforms a database, disguising only sensitive

rules while still allowing other valuable information to be revealed.

Furthermore, PPDM represents a distinct set of data mining activities comprising techniques developed to safeguard data privacy, enabling the knowledge discovery process to proceed unimpeded. The primary objective of PPDM is to extract pertinent knowledge from extensive databases, delivering accurate data mining outcomes while preventing the disclosure or inference of sensitive information. In PPDM, novel techniques are devised to ensure privacy for the knowledge extracted during data mining, all while ensuring that privacy concerns do not impede the knowledge discovery process.

Dealing with the excess of personal data poses a significant challenge in privacy-preserving data mining. While various technologies support appropriate data handling, there is still considerable work to be done, and several obstacles must be addressed for their effective deployment.

3.3 Review of privacy preserving data mining algorithms

Numerous studies have delved into privacy-preserving data mining (PPDM) across various contexts. For instance, [91] explored the performance of classification algorithms—such as C4.5, Naïve-Bayes, and Random Tree—in a privacy-preserving classification scenario. To safeguard data privacy, they employed input perturbation using differential privacy, analyzing the relationship between the 'e' parameter and the classifiers' accuracy. Another study by [92]introduced a rule-based classifier utilizing the ant bee colony optimization algorithm and differential privacy's input perturbation for privacy-preserving classification. Additionally, [93] addressed the classification problem in a distributed environment, proposing the Privacy-Preserving Distributed ERT approach, adapting the Extremely Randomized Trees algorithm for settings where structured data is distributed across multiple sources.

In a quest to eliminate sensitive information through decomposition and dimensionality reduction, [94] devised a methodology leveraging the singular value decomposition (SVD) machine learning algorithm and 3D rotation data perturbation (RDP) for data privacy preservation.

For data exchange among varied sources, [95] introduced a novel technique called Remodeling. This technique, in tandem with the k-anonymity and k-means algorithm, aims to minimize data loss, enhance privacy preservation, and maintain data diversity. Another inno-

vation in privacy-preserving data mining architecture, proposed by [96], is a distributed model based on Multi-Party Computation (MPC) designed to manage vast and multi-source data, considering potential data inaccessibility during aggregation due to sporadic network connectivity or abrupt user departures. In healthcare, [97] scrutinized PPDM, analyzing data privacy and utility requirements for healthcare process data, assessing the suitability of privacy-preserving data transformation methods to anonymize healthcare data. Finally, [98] introduced a multi-objective sanitization model to secure 6G Internet of Things (IoT) networks.

Recent research has explored diverse aspects of privacy-preserving data mining (PPDM) in varied contexts. For instance, [99] employed an ant colony optimization (ACO) approach with multiple objectives and a transaction deletion mechanism to safeguard confidential information. In the IoT domain, [99] proposed a machine learning model for data sanitization based on data privacy. Additionally, [100] applied PPDM to open government data from heterogeneous sources, suggesting a method that permits users to adjust the privacy threshold level for balancing privacy risk with data utility, even in the absence of identifiers.

To conceal sensitive itemsets, [101] introduced a greedy approach involving the insertion of new transactions to reduce the support of sensitive itemsets. They used a statistical method to determine the items for insertion. Meanwhile, [102] devised a greedy algorithm named SIFIDF, which deletes transactions from a database rather than adding synthetic data, utilizing the TF-IDF measure (term frequency-inverse document frequency) to select transactions to be eliminated, reducing side effects.

Moreover, [103] presents an evolutionary computation-based model for PPDM under a multi-threshold constraint. In another study, [104] addressed the critical issue of setting minimum support thresholds, proposing a modified minimal support concept to establish a stricter threshold for objects containing multiple items according to a given threshold function.

3.4 Meta-heuristic algorithms for PPDM

Meta-heuristic algorithms play a significant role in privacy-preserving data mining (PPDM) by addressing optimization challenges effectively. These algorithms offer near-optimal solutions within reasonable timeframes, particularly beneficial when handling extensive datasets.

Belonging to a class of general algorithmic methods, meta-heuristics are designed to tackle intricate optimization problems. Evolutionary computation (EC) is a notable meta-heuristic approach, inspired by natural evolution processes and first emerging in the late 1960s. EC algorithms are proficient at resolving complex search and optimization challenges, expanding the horizons of optimization to unattainable areas previously. Computer scientists have developed numerous algorithms mimicking natural evolution, especially to solve NP-hard problems. Moreover, many of these evolutionary algorithms have been extended to address multi-objective optimization problems, giving rise to a substantial body of work in this field (e.g., [105, 106, 86, 107, 108, 109]).

In the domain of PPDM, particularly for concealing sensitive itemsets, single-objective evolutionary computation algorithms optimize a single objective function, usually a weighted sum of various side effects. On the contrary, multi-objective evolutionary computation algorithms aim to concurrently optimize multiple conflicting objectives without merging them into a single function. The solutions derived from multi-objective approaches are non-dominated, indicating that no other solution in the search space is superior in all objectives simultaneously. By presenting a set of non-dominated solutions, multi-objective methods offer decision-makers greater flexibility in choosing the most appropriate solution according to their preferences.

3.4.1 Single-objective metaheuristic algorithms for PPDM

Several significant studies have been conducted in using single-objective optimization evolutionary algorithms for PPDM.

[110] introduces the EHS algorithm, employing an evolutionary algorithm driven by a single-objective optimization method to eliminate sensitive itemsets from transaction databases. It involves two techniques: one for adjusting itemset support and the other for modifying transaction structures to obscure sensitive itemsets.

[111] proposes the EPIS algorithm for privacy-preserving frequent itemset mining, employing single-objective optimization to minimize information loss while concealing sensitive data. The approach involves modifying the database by adding transactions that either support or do not support sensitive itemsets.

[112] introduces the GHI genetic algorithm aimed at concealing sensitive itemsets in transactional databases. Using a single-objective optimization function based on a fitness assessment considering the number of sensitive itemsets and the database's size after modification.

[113] presents a differential evolution algorithm that utilizes single-objective optimization to perform privacy-preserving frequent itemset mining. The algorithm focuses on reducing the support of sensitive itemsets by introducing new transactions into the database.

Furthermore, studies by [114, 115] introduce the sGA2DT, pGA2DT, and cpGA2DT algorithms, aiming to conceal sensitive itemsets within large databases. These algorithms select an appropriate set of transactions to delete, considering each chromosome as a potential solution for transaction deletion. A fitness function incorporates three side effects, each weighted differently as per user-defined values. These algorithms offer variations in approach to optimize transaction deletion for hiding sensitive itemsets, considering population size and database scan cost reduction.

Additionally, [116] introduced the PSO2DT algorithm, leveraging a discrete PSO-based approach to identify transactions to remove while hiding sensitive itemsets and minimizing side effects via a single objective function.

Finally, [117] proposed the ACS2DT algorithm based on Ant Colony System to delete transactions. This bio-inspired algorithm aims to hide sensitive itemsets while minimizing three side effects through a fitness function that calculates a weighted sum of these effects to evaluate solutions.

3.4.2 Multi-objective metaheuristic algorithms for PPDM

Multi-objective optimization has been significantly applied in privacy-preserving data mining to address the task of hiding sensitive itemsets. Several notable studies illustrate the utilization of multi-objective optimization for this purpose:

[118] introduces a multi-objective genetic algorithm (MOGA) to conceal sensitive itemsets within transaction databases. It aims to minimize the number of deleted transactions, the count of modified transactions, and the maximum number of sensitive itemsets in each modified transaction.

[119] presents an algorithm based on NSGA-II (Non-dominated Sorting Genetic Algorithm II) focusing on three objectives: the count of deleted and modified transactions and the utility loss due to modifications to conceal sensitive itemsets.

[120] develops a multi-objective algorithm utilizing the bat algorithm to hide sensitive itemsets while minimizing the number of deleted and modified transactions and the loss of utility.

[121] proposes a multi-objective genetic algorithm for privacy-preserving frequent itemset mining. It aims to minimize the number of deleted and modified transactions and the utility loss caused by modifications.

Moreover, [122] designs the NSGA2DT algorithm, derived from the NSGA-II algorithm, to identify optimal transactions for deletion to conceal sensitive itemsets. This algorithm incorporates database dissimilarity as a fourth factor, striving for a better balance between multiple side effects.

In another approach, [123] presents the GMPSO algorithm, a multi-objective PSO-based technique using a grid-based method. GMPSO targets Pareto optimal solutions for data sanitization, utilizing strategies for global and local solutions and adapting the Pre-large concept to expedite the evolutionary process, reducing the frequency of multiple database scans.

These studies collectively demonstrate the efficacy of multi-objective optimization in privacy-preserving data mining, offering valuable insights for future research in this domain.

3.5 Conclusion

In summary, privacy-preserving data mining represents a vital research domain aimed at safeguarding sensitive data while extracting valuable patterns. The methodologies and strategies employed in PPDM strive to find a delicate equilibrium between data privacy and data utility by employing techniques like data perturbation, anonymization, and encryption. This area finds particular significance in domains where privacy-sensitive data is central, such as healthcare, finance, and e-commerce. Yet, while PPDM techniques offer protection for sensitive data, it's imperative to evaluate their efficacy to ensure they don't compromise individuals' privacy. Hence, continuous research and advancements in PPDM are necessary to address emerging privacy concerns.

Moreover, ethical and legal considerations play a pivotal role when deploying PPDM techniques, given their substantial impact on individuals and society. Implementing privacy-preserving measures should involve the consent and awareness of data owners and users while aligning with pertinent laws and regulations. With evolving data privacy concerns owing to technological advancements, PPDM researchers and practitioners must stay abreast of the latest trends and consistently enhance their techniques to mitigate emergent privacy risks. Overall, PPDM serves a crucial function by safeguarding the privacy of individuals and organizations, enabling them to glean insights from their data.

In this thesis, we focus on the development of new methods in privacy preserving frequent itemset mining. For that purposes, the proposed methods bring new contributions to this field in two perspectives: First, contrary to most of the existing approaches which sanitize databases by deleting complete transactions, our proposed methods operate at the finer level of items which leads to less update in the original database. Moreover, we address in this thesis the PPDM problem in the context of uncertain databases. To the best of our knowledge, this issue has not been tackled previously.

Chapter 4: Hiding Sensitive Frequent Itemset via Items Removal

4.1 Introduction

The identification of relevant patterns and association rules is a fundamental task in data mining, employing a variety of techniques and methods [47]. Among these methods, Apriori, introduced by [23], stands out as the first and most widely used algorithm for discovering association rules in transactional databases. Another prominent algorithm in this domain is FP-Growth, proposed by [124]. While these techniques are effective in extracting implicit information from large databases, a major concern arises regarding their potential misuse, leading to the exposure of private and sensitive data such as credit card numbers, personal identification numbers, and telephone numbers. This concern is exemplified in a scenario presented in [125], where banks sharing their databases could result in a compromise of sensitive information. To address this issue, it is crucial to incorporate privacy-preserving techniques, including data perturbation and encryption, to mitigate risks and ensure the confidentiality of sensitive data.

The depicted scenario underscores a significant privacy concern, as databases often encompass sensitive details about customers, their accounts, and their families. While sharing certain data can be mutually advantageous, banks need to safeguard their strategic information and customer trends. Consequently, it becomes crucial to transform the data intended for sharing, concealing sensitive patterns to prevent their exposure through data mining algorithms. To achieve this, the adoption of appropriate privacy-preserving techniques is imperative. Algorithms like Apriori and FP-Growth, while effective in extracting implicit knowledge from large databases, also pose the risk of malicious use to unveil private and confidential information,

such as credit card numbers and personal identification numbers. Thus, the implementation of robust privacy measures becomes essential to shield individuals and organizations from the potential misuse of data mining techniques.

PPDM techniques are tailored to hide sensitive frequent itemsets, which represent sets of values frequently found in transactions. Database sanitization is a common method for securing sensitive itemsets; however, this process can lead to undesired side effects such as hiding failure, missing cost, and artificial cost. Selecting optimal data modifications to minimize these side effects becomes a challenging NP-hard optimization problem. Thus, designing PPDM algorithms that strike a balance between preventing sensitive information disclosure and preserving non-sensitive yet important information is a key challenge. While existing PPDM techniques typically delete entire transactions to hide sensitive itemsets, this approach may result in significant loss of non-sensitive information. To address this concern, this chapter introduces a novel algorithm called Non-dominated Sorting Genetic Algorithm II for Item Deletion (NSGAII4ID). NSGAII4ID treats the task of hiding sensitive itemsets as a multi-objective optimization problem, achieving concealment by deleting individual items in transactions rather than entire transactions. This approach minimizes the overall impact on the database.This is done by performing optimizations at two levels with the aim of minimizing side-effects:

1. At the external level (transactions), NSGAII4ID adapts the NSGA-II algorithm [126] for multi-objective optimization to use three objective functions representing the side-effects. Optimization is done at the transaction level to find a set of candidate transactions from which some items will then be selected and removed to minimize the objective functions.

2. Then, at the evaluation level (items), for a given combination of candidate transactions, the problem of determining the items that should be removed from each transaction is shown to be nothing but a Set cover Problem (SCP) which is a well-known NP-Hard combinatorial optimization problem. To solve the SCP, a fast greedy algorithm is applied which provides a good approximate solution in polynomial time. The resulting optimal subset of removed items determines the values returned by the objective functions, i.e., the quality of a selected subset of candidate transactions. To our best knowledge, this chapter is the first to explore this two level optimization approach for PPDM, which remove items rather than whole transactions to reduce the amount of change. We call this novel approach two-level multi-objective optimization for PPDM.

The key differences between NSGAII4ID and previous approaches are:

- Unlike many PPDM algorithms, NSGAII4ID applies a multi-objective approach to take into account the multiple side-effects in the process of finding a solution.
- Instead of removing entire transactions, as most algorithms for hiding sensitive itemsets do, NSGAII4ID acts at a lower granularity level and removes subsets of items from transactions. This leads to a more refined change operator and is more in line with the minimality principle, which suggests that changes made to a database to hide sensitive information should be as limited as possible.
- NSGAII4ID integrates the dissimilarity criterion as an objective to be minimized in addition to other side effects. Therefore, the goal is to search for solutions that also minimize the amount of change made to the original database during sanitization.

This chapter is organized as follows. Preliminary definitions are presented in Section 4.2. Then, Section 4.3 presents some formal results that simplify the computation of side effects, and then formally defines the task of hiding sensitive itemsets as a multi-objective optimization problem. Section 4.4 is devoted to the presentation and explanation of the proposed NSGAII4ID sanitization algorithm. In section 4.5, we give a detailled example illustrating our proposed method. Thereafter, experimental results are reported and discussed in Section 4.6. Finally, Section 4.7 Concludes the chapter.

4.2 Preliminaries

In this section, we present preliminaries and some basic notions about the problem of hiding sensitive frequent itemsets with minimal side effects. We also present the Set Cover Problem (SCP) which is a well-known combinatorial optimization problem that will be used later in our method to detect an optimal subset of items from every victim transaction.

4.2.1 Basic definitions

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a finite set of items. $D = \{T_1, T_2, \ldots, T_n\}$ is a transactional database which contains a set of transactions, where each transaction $T_q \in D$ is a subset of I and has a unique identifier q called TID. MST denotes a minimum support threshold given by the user.

An itemset X is a subset of items ($X \subseteq I$). Its support in the database D denoted $Support_D(X)$ is the number of transactions in D that include X. An itemset X is frequent (in D) if and only if $Support_{D'}(X)$ is greater or equal to the threshold MST. The set of frequent itemsets is denoted by $L = \{l_1, l_2, \dots\}$.

Definition 4.2.1 (Sensitive itemsets) . *We denote by $SI \subseteq L$ the set of sensitive frequent itemsets. We notice that the set of sensitive frequent itemsets is specified by the user or an expert. The set of sensitive frequent itemsets is given by $SI = \{s_1, s_2, \dots\}$.*

The main goal of PPDM is to hide sensitive information from original database so that this kind of information cannot be disclosed by applying data mining techniques. For hiding sensitive itemsets, some candidate sensitive transactions that contain sensitive itemsets have to be modified by deleting some victim sensitive items from each candidate transaction. The challenge is to look for the set of candidate transactions and the way to choose victim items that minimize the possible undesired side effects. Actually, there are three major side effects to be taken into account, namely the "hiding failure", the 'missing cost' and the "artificial cost".

We denote by D' the database which results from the original database D after the sanitization process and by L' the set of frequent itemsets in D'. The three side effects mentioned above are formally defined as follows:

Definition 4.2.2 (Hiding failure) . *The "hiding failure" (S_F_H) is the number of sensitive itemsets that the sanitization process fails to hide. It is is given by: $S\_F\_H = |SI \cap L'|$. In other words, S_F_H is the number of sensitive items of SI that are frequent in L' : $S\_F\_H = |\{X \in SI | Support_{D'}(X) \geq MST\}|$.*

Definition 4.2.3 (Missing cost) . *The "missing cost" (N_S_L) is the number of non-sensitive frequent itemsets that are hidden (become infrequent) after the sanitization process. This number is given by: $N\_S\_L = |NS - L'| = |(L - SI) - L'|$. In other words, N_S_L is the number of frequent non sensitive itemsets in D that are infrequent in D' : $N\_S\_L = |\{X \in NS | Support_{D'}(X) < MST\}|$.*

Definition 4.2.4 (Artificial cost) . *The "artificial cost" (S_F_G) is the number of non-frequent itemsets that become frequent after the sanitization process. This number is given by $S\_F\_G = |L' - L|$.*

The relationship between these three side effects is depicted in Fig 1.4.1

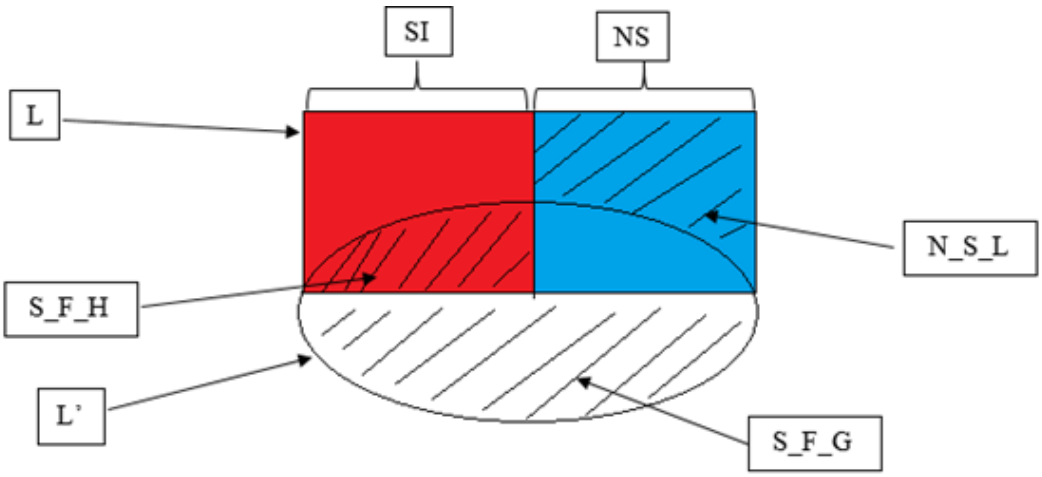

Figure 4.1: Relationships between the three side effects of data sanitization in PPDM

In addition to minimizing the previously mentioned factors for an effective sanitization process, another important criterion to minimize is the dissimilarity between the original and resulting databases. A desirable solution should aim to minimize the alterations made to the original database during the sanitization process. A straightforward method to quantify this dissimilarity is by counting the total number of victim items removed from the original database to derive the sanitized one.

Definition 4.2.5 (Dissimilarity measure) . *The dissimilarity measure (Dis) between the original database D and the modified database D' obtained by sanitization is the number of victim items removed from D to obtain D'. It is calculated as follows: $Dis = \|D\| - \|D'\|$, where $\|D\|$ is the total number of items present in all the transactions of $\|D\|$, i.e. $\|D\| = \sum_{T \in D} |T|$.*

Definition 4.2.6 (Sensitive transaction) . *A sensitive transaction is a transaction that contains at least one sensitive itemset.*

Definition 4.2.7 (Projected database) . *Let D be a transactional database and SI be the set of sensitive itemsets. We define the projected database D^* containing the sensitive transactions of D as follows : $D^* = \{t | t \in D, \exists s \in SI \text{ such that } s \subseteq t\}$.*

4.2.2 Set Cover Problem

The Set Cover problem (SCP) is a well-known NP-hard combinatorial optimization problem, with its associated decision problem being NP-Complete (refer, for instance, to [127]). It has demonstrated successful applications in modeling various industrial problems, including scheduling, manufacturing, service planning, and location problems.

It is noteworthy that the Set Cover Problem (SCP) can be approximated in polynomial time using a greedy algorithm. In our approach, we integrate SCP at the evaluation level for items. Specifically, given a set of transactions encoded in a solution (a chromosome of our genetic algorithm), SCP is employed to formulate the problem of identifying the minimal subset of items to be removed from a transaction to conceal all its sensitive itemsets. The resulting solution is then evaluated based on our defined objectives.

The formal definition of the SCP is as follows:

Let $U = \{e_1, e_2, \ldots, e_m\}$ be a universe of elements and $F = \{F_1, F_2, \ldots, F_n\}$ be a collection of subsets such that $F_j \subseteq U$ for each $j \in \{1, \ldots, n\}$ and $\bigcup_{j \in \{1,\ldots,n\}} F_j = U$. Each set F_j covers at least one element of U. The goal of SCP is to find a minimum sub-collection of sets $X \subseteq F$ that covers all elements in U, i.e. a sub-collection $X \subseteq F$ that satisfies :

- $\bigcup_{s \in X} s = U$ (Covering all elements of U).
- For all $Y \subseteq F$ such that $|Y| < |X|$, $\bigcup_{s \in Y} s \neq U$ (Minimality), where $|X|$ (resp. $|Y|$) denotes the cardinal number of X (resp. Y).

A straightforward greedy approximate algorithm for the Set Cover Problem (SCP) constructs the solution in multiple steps, making a locally optimal decision at each step. The approach involves selecting, at each step, a set from the collection F that has the maximum intersection with the elements of the universe U not yet covered.

Consider the example illustrated in Figure 4.2, adapted from [1]. We have a set of 12 elements (the universe U) and a collection $F = F_1, F_2, F_3, F_4, F_5, F_6$ consisting of six subsets whose union forms the universe. According to the greedy algorithm, the first chosen set is F_1 since it has the largest intersection with the set of elements not yet covered (here, U itself, as all elements are initially uncovered). The greedy logic is based on the principle that the larger the

set, the more uncovered elements it covers. The algorithm continues with F_4, covering three more elements, and then F_5, covering two elements. Towards the end, the choice is between F_6 or F_3, as both cover the same number of uncovered elements (one element in this case). With this approximate approach, the cardinality of the cover set is 4. It is important to note that for this example, the optimal solution has only three elements: F_3, F_4, and F_5.

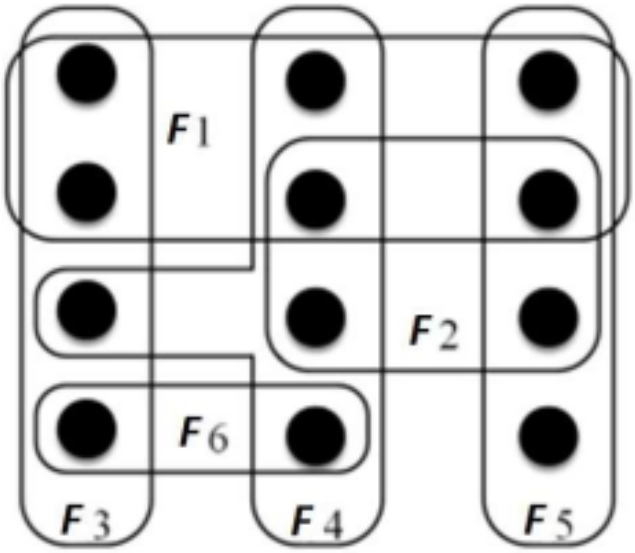

Figure 4.2: An example of SCP with 12 elements and a collection of 6 subsets, by [1]

Now, once the main concepts are defined, the subsequent section show how the PPDM problem is formally modelled as a multi-objective optimisation problem.

4.3 PPDM as a multi-objective optimization problem

Before presenting a formal definition of Privacy-Preserving Data Mining (PPDM) as a multi-objective optimization problem, let us revisit the three side effects that require minimization. In particular, we aim to devise an efficient method for computing hiding failure and missing cost in the sanitized database without the need for an additional scan. Additionally, we'll illustrate that the artificial cost side effect is not relevant in our specific context.

The determination of S_F_H and N_S_L, corresponding to hiding failure and missing cost, respectively, inherently involves the calculation of support in the adapted database D' for each frequent itemset X (denoted as $Support_D(X)$). Concurrently, the computation of S_F_G for artificial cost needs the calculation of the set L' containing frequent itemsets in D'.

Given that these three values are calculated for each evaluation of a given chromosome, conducting this computation by scanning the entire database D' for *S_F_H* and *N_S_L*, and then explicitly identifying the set L' of frequent itemsets in D' for *S_F_G*, would result in significant computational costs. Fortunately, we demonstrate, under a reasonable hypothesis, that it is feasible to compute the values for hiding failure and missing cost without the need to scan the entire database D'. Furthermore, we establish that the third value for artificial cost consistently remains zero, making it irrelevant for the optimization task.

Hypothesis 1 *We assume that no transaction in the original database exclusively consists of items belonging all to sensitive itemsets.*

This assumption posits that every transaction in the input database must include at least one item not belonging to any sensitive itemset. This assumption is generally valid in practical scenarios, as evidenced by all benchmark databases utilized in the experimental study detailed in Section 6. Typically, the proportion of items associated with sensitive itemsets is limited compared to the entire set of items. Consequently, encountering transactions exclusively composed of items from sensitive itemsets is highly improbable. An immediate implication of this assumption is that the number of transactions remains unchanged before and after the sanitization process, i.e., $|D| = |D'|$. As a result, the following outcomes are derived:

proposition 1 . *Consider two transactional databases, D and D', where D' is obtained from D through a sanitization process. Let D_1 represent the subset of transactions in D that has undergone modifications to yield the set of modified transactions D'_1 in D'. Each $t' \in D'_1$ is obtained by removing from a corresponding transaction $t \in D_1$ the set of items $t - t'$. Let X be an itemset. Under Hypothesis 1, the support of X in D' is related to its support in D as follows:*

$$Support'_D(X) = Support_D(X) - \frac{|\{t \in D_1 | (t - t') \cap X \neq \varnothing\}|}{|D|} \tag{4.1}$$

proof 1 *First, notice that the sets $D' - D'_1$ and $D - D_1$ are identical and both of them contains the transactions that have not been modified. It follows that:*

$$|\{t' \in D' - D'_1 | X \subseteq t'\}| = |\{t \in D - D_1 | X \subseteq t\}| | \tag{4.2}$$

Let t' be a transaction from D'_1, and let t be the corresponding transaction in D_1 (where t' is

obtained by removing $(t-t')$ from t). Then, $X \subseteq t'$ if and only if $X \subseteq t$ and $(t-t') \cap X = \emptyset$. This condition implies that t' still contains X after removing $(t-t')$ from t if and only if t already contains X, and the removed set $(t-t')$ does not include any item from X. It follows that:

$$|\{t' \in D'_1 | X \subseteq t'\}| = |\{t \in D_1 | X \subseteq t\} - \{t \in D_1 | (t-t') \cap X \neq \emptyset\}| \tag{4.3}$$

Now, by the definition of support, we have:

$$Support_{D'}(X) = \frac{|\{t' \in D' | X \subseteq t'\}|}{|D'|} = \frac{|\{t' \in D' - D'_1 | X \subseteq t'\}|}{|D'|} + \frac{|\{t' \in D'_1 | X \subseteq t'\}|}{|D'|}$$

From Equations 4.2 and 4.3 and since $|D| = |D'|$, we have:

$$Support_{D'}(X) = \left(\frac{|\{t \in D - D_1 | X \subseteq t\}|}{|D|} + \frac{|\{t \in D_1 | X \subseteq t\}|}{|D|}\right) - \frac{|\{t \in D_1 | (t-t') \cap X \neq \emptyset\}|}{|D|}$$

Then, we obtain:

$$Support_{D'}(X) = Support_D(X) - \frac{|\{t \in D_1 | (t-t') \cap X \neq \emptyset\}|}{|D|} \blacksquare$$

Proposition 1 allows one to iteratively compute $Support_{D'}(X)$, starting with $Support_D(X)$ as initial value. For each transaction t in D_1 such that the corresponding t' in D'_1 is generated by removing at least one element of X, $Support_{D'}(X)$ is decremented by $1/|D|$ (see Algorithm 2).

This is pivotal as it offers an efficient method to calculate the updated support of an itemset in the sanitized database D', and consequently, evaluate the side effects of hiding failure and missing cost without the necessity for an additional scan of D'.

proposition 2 . *Consider transactional databases D and D', where D' is obtained from D through sanitization. Let D_1 represent the subset of transactions in D that has been modified to yield the set of modified transactions D'_1 in D', where each $t' \in D'_1$ is obtained by removing from a corresponding transaction $t \in D_1$ the set of items $t - t'$. Let X be an itemset. Under Hypothesis 1, it is always true that $S\_F\_G = 0$.*

proof 2 . *From Definition 4.2.4, we can express S_F_G as $|L' - L|$. By employing Equation 4.1 and considering that $\frac{|t \in D_1 | (t-t') \cap X \neq \emptyset|}{|D|} \geq 0$ for each itemset X, we conclude that $Support_{D'}(X) \leq Support_D(X)$. Consequently, if $Support_{D'}(X) \geq MST$ (i.e., X is frequent in D'), then it follows*

that $Support_D(X) \geq MST$ *(i.e., X is frequent in D). This implies* $L' \subseteq L$*, meaning that* $L' - L = \emptyset$*, and thus* $|L' - L| = 0$. ■

Proposition 2 affirms that the artificial cost side effect is always equal to zero in our case. This observation is crucial as it streamlines the problem, allowing us to concentrate on the minimization of only three objective functions instead of four. Figure 4.3 illustrates the adjusted relationships between side effects, incorporating the insights from Proposition 2.

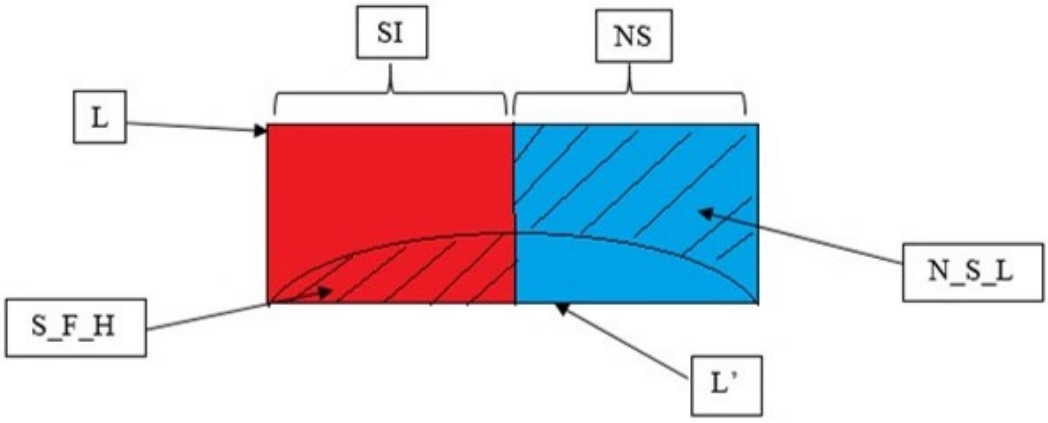

Figure 4.3: New Relationships between the side effects of data sanitization in PPDM.

Given a transaction database D, the Privacy-Preserving Data Mining (PPDM) problem with item deletion aims to identify an optimal set of sensitive transactions from D. For each of these transactions, the objective is to determine an optimal set of sensitive items for deletion. An optimal solution should yield a sanitized database D' where the side effects S_F_H and N_S_L, along with the dissimilarity measure Dis, are simultaneously minimized.

Now, let us give a formal definition of our optimization problem. Recall first some notations: $I = i_1, i_2, \ldots, i_m$ denotes the set of items, and $D = T_1, T_2, \ldots, T_n$ is a transactional database with n transactions. Additionally, SI (resp. NS) represents the set of sensitive (resp. non-sensitive) frequent itemsets. The objective is to hide the itemsets of SI and not those of NS.

Consider n binary decision variables $x_1, \ldots, x_n$, and define the vector $\vec{x} = (x_1, x_2, \ldots, x_n)$, where each x_i is associated with the transaction T_i. Specifically, $x_i = 1$ if T_i is a candidate transaction (from which some items should be removed), and $x_i = 0$ otherwise. Thus, a vector $\vec{x}$ encodes a potential solution or a chromosome, i.e., a subset of candidate transactions to be modified to obtain a sanitized database D' from the original database D. The SCP is solved to determine the optimal subset of items to remove from each candidate transaction T_i with $x_i = 1$.

Let scp be the function that takes a vector $\vec{x}$ and returns the sanitized database D' obtained from D by considering the transactions in $\vec{x}$ as candidate transactions to be updated. Therefore, for a given $\vec{x}$, we have: $scp(\vec{x}) = D'$.

Based on Definitions 4.2.2, 4.2.3, and 4.2.5, the three side effects can be defined as functions of the decision variable vector $\vec{x}$ using the following equations:

- Hiding failure is given by the function :

$$sfh(\vec{x}) = |\{X \in SI | Support_{scp(\vec{x})}(X) \geq MST\}| \tag{4.4}$$

- Missing cost is given by the function :

$$nsl(\vec{x}) = |\{X \in NS | Support_{scp(\vec{x})}(X) < MST\}| \tag{4.5}$$

- Dissimilarity is given by the function :

$$dis(\vec{x}) = \|D\| - \|scp(\vec{x})\| = \sum_{T \in D} |T| - \sum_{T \in scp(\vec{x})} |T| \tag{4.6}$$

We have no particular constraints except the fact that each variables x_i $(i = 1..n)$ takes binary values. Now, the PPDM problem is formally defined in equation 4.7 as follows:

$$\begin{aligned} &Minimise : \{sfh(\vec{x}), nsl(\vec{x}), dis(\vec{x})\} \\ &Subject\ to : \ x_i \in \{0,1\}\ for\ i = 1..n \end{aligned} \tag{4.7}$$

4.4 Description of NSGAII4ID algorithm

A new sanitization algorithm is proposed, named NSGAII4ID (Non-dominated sorted genetic algorithm II for item deletion). The NSGAII4ID algorithm is designed as a two-level optimization algorithm. In the first level, we use an adapted version of the NSGA-II multi-objective genetic algorithm introduced in [126]. The primary objective at this level is to identify an optimal combination of candidate transactions. These transactions are potential candidates for modification. This modification of some selected transaction constitute the second optimiza-

tion level where the goal is to select an optimal subset of victim items in each transaction for removal. Thus, the task of determining the items to be removed from each candidate transaction is treated at a separate optimization level and modeled by the Set Cover Problem (SCP).

The rationale behind the choice of victim items lies in the observation that removing at least one item from a sensitive itemset X in a sensitive transaction t results in X no longer being included in t, leading to a decrease in the support of X. The primary objective is to attain, for each sensitive itemset, a support value below the minimum threshold set by the user. Consequently, only sensitive transactions that contain at least one sensitive itemset (elements of D^*) are considered as possible candidate transactions for modification.

The algorithm oversees a population of N chromosomes, generated randomly during the initialization step. The user defines the population size N as a parameter of the algorithm. Each chromosome serves as a potential solution to the problem and has a fixed length determined by the user. The chromosome comprises a sequence of positive integer values, with each integer representing the TID of a candidate transaction if it is non-zero. A value of 0 signifies the absence of a specified transaction in the respective gene. This chromosome structure, incorporating zero values, provides flexibility, enabling the encoded candidate transactions' count to vary across different chromosomes. To illustrate, consider this example of chromosome:

10	0	452	5	0	100	6

This chromosome encodes the subset of candidate transactions whose TIDs are : 10, 452, 5, 100 and 6. Based on the previous definitions, a flowchart of the proposed algorithm is depicted in Figure 4.4. Algorithm 1 presents the pseudo-code for the NSGAII4ID algorithm, which incorporates the resolution of the Set Cover Problem (SCP) within the chromosome evaluation, adapting the general procedure of the NSGA-II algorithm.

NSGAII4ID takes several elements as input: The original database D, the set of frequent itemsets L obtained in D with a user-defined minimum support threshold MST, a set SI comprising sensitive itemsets, a population size N, chromosome size $nVar$, and a maximum number of iterations. The algorithm produces a set of optimal solutions. The frequent itemsets can be identified using any frequent itemset mining algorithm. In our implementation, the Apriori algorithm is used, considering an itemset frequent if its support is equal to or exceeds MST.

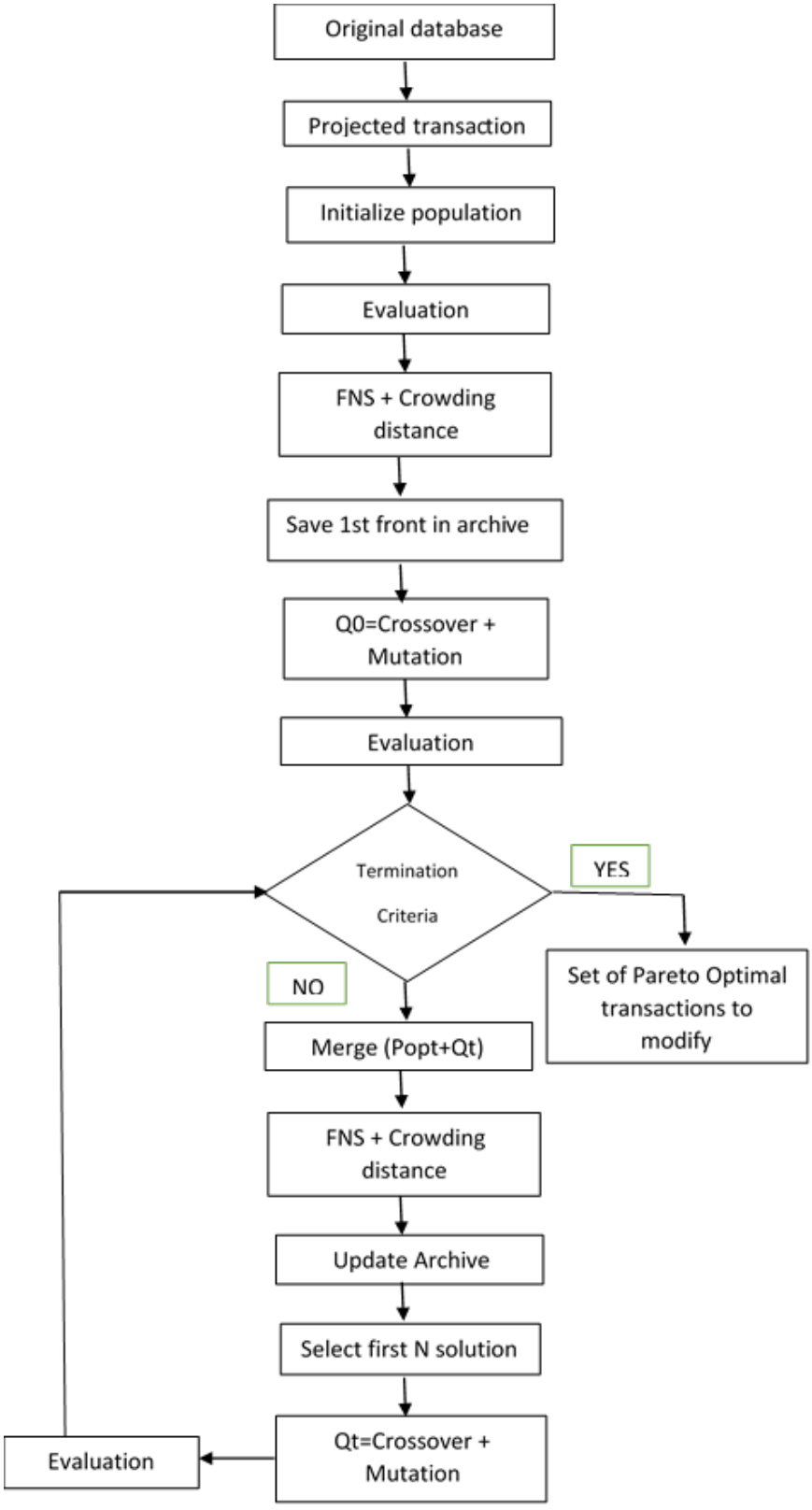

Figure 4.4: Flowchart of the proposed algorithm

The algorithm begins by projecting the database D and collecting the set D^* of sensitive transactions (lines 2-8). A sensitive transaction is one from the original database D that contains

at least one sensitive itemset.

Following this, a population of N chromosomes, representing potential solutions, is initialized (lines 9-12). Each chromosome is a set of genes, with each gene representing the Transaction ID (TID) of a sensitive transaction from the projected database D^*. The initialization employs random selection, using the Roulette wheel technique to increase the likelihood of choosing transactions with multiple sensitive itemsets. The chromosome size $nVar$ is a user-specified algorithm parameter.

Next, multi-objective optimization functions are evaluated for each potential solution (lines 13-15), with three fitness values associated with each chromosome representing the three objective functions. Further details about the evaluation function are provided in Algorithm 2.

Fast Non-Dominated Sorting and Crowding distance mechanisms, introduced by ([126]), are applied to organize solutions into fronts within the population, with the best solutions residing in the first front. The initial front is preserved in an additional population referred to as the archive, which has an unlimited size (line 19).

Crossover and mutation operations are then applied to generate an offspring population (lines 20-21). The crossover operation involves randomly selecting a gene from each chromosome and exchanging them. If redundancy arises in the new chromosome, the redundant gene is replaced with a new value drawn randomly from D^*. The mutation operation randomly selects a gene from a chromosome (a potential solution obtained after applying a crossover operation) and replaces it with another random solution from D^* not already present in the chromosome. Subsequently, each chromosome in the offspring population is evaluated (lines 22-24).

The evolution process iterates, with the current population and the offspring population merged in each iteration, resulting in twice the number of potential solutions ($2 \times N$). Fast Non-Dominated Sorting and Crowding distance strategies are applied to sort solutions into fronts (lines 30-31), and the best solutions are stored in the archive (lines 32-33). This archival process ensures that solutions included in the archive are regularly updated throughout iterations. Following this, the new population is formed by selecting the first N best potential solutions from the merged population (line 34). Subsequently, crossover and mutation operations are applied (lines 35-36) to generate a new offspring population. Each chromosome in the offspring population is evaluated according to the three objective functions (lines 37-39). The

actions conducted between lines 29 and 40 are repeated until a specified condition is met.

In the process of evaluating each chromosome, we introduce an assessment mechanism that identifies an optimal set of items to be removed from each transaction encoded in the chromosome. Subsequently, it computes three objective functions based on this optimal item removal. The pseudo-code for this evaluation function is outlined in Algorithm 2.

Initiating the algorithm (line 1), Algorithm 2 begins by isolating the set of sensitive transactions encoded in the chromosome c for evaluation. For each transaction t in this set, the algorithm constructs the necessary components to formulate the problem of determining the optimal set of items to remove from t, treating it as a Set Cover Problem (SCP). Specifically, the universe U is defined as the set of all sensitive itemsets included in transaction t (line 5), while $Sitems$ encompasses all items associated with the sensitive itemsets of U (line 6).

Subsequently, the algorithm employs the function *set_cover_problem* (line 7), detailed in Algorithm 3, to obtain the set of items, denoted as $Victitems_t$, that should be removed from transaction t. Once $Victitems_t$ is determined, the algorithm proceeds to update:

1. D' is obtained by eliminating the items in $Victitems_t$ from transaction t (as indicated in line 8);

2. The support in D' of each itemset, initially frequent in D and present in transaction t before removing the items of $Victitems_t$, is computed (lines 9-13).

3. The dissimilarity between D and D' is calculated by incorporating the number of items present in $Victitems_t$ (as indicated in line 14).

Upon completing the processing for all transactions, we obtain the updated support values in the modified database D' for all itemsets that were frequent in the original database D. Additionally, we calculate the final dissimilarity value between D and D', representing the third objective function. Subsequently, the value of the first objective function S_F_H (hiding failure) is determined by counting the number of sensitive itemsets that were frequent in D and remain frequent, in D' (lines 19-21). Similarly, the second objective function N_S_L (missing cost) is computed by counting the instances of non-sensitive itemsets that were frequent in D but become non-frequent in D' (lines 22-24).

Algorithm 1 NSGAII4ID pseudo-algorithm

Require: D: Original Database; L: Set of frequent itemsets; MST: Minimum support threshold; SI: Set of sensitive itemsets; N: Population size; $nVar$: Chromosome size; Max_iter: Maximum number of iterations.

Ensure: Set of optimal solutions to be modified (Archive).

{Construction of the projected database containing transactions including at least one sensitive itemset}

$D^* \leftarrow \emptyset$;
for (each transaction T_q in D) **do**
  for (each sensitive itemset s in SI) **do**
    if ($s \subseteq T_q$) **then**
      $D^* \leftarrow D^* \cup \{T_q\}$;
    end if
  end for
end for

{Initialization of the population}

for (each $i = 1 : N$) **do**
  $Vector \leftarrow Roulette\_Wheel(D^*)$;
  $Pop_0[i] \leftarrow rand(Vector, N, nVar, null)$;
end for

{Evaluation of each chromosome : S_F_H, N_S_L, Dis, are calculated within the evaluation function and each of them has a discrete value}

for (each chromosome *chrom* of Pop_0) **do**
  $Evaluate(chrom, D^*, SI, D, NS)$;
end for
$[Pop_0, F] \leftarrow Fast\_Non-Dominated\_Sort(Pop_0)$;
$Pop_0 \leftarrow Crowding\_distance(Pop_0, F)$;
$Pop_0 \leftarrow Sort\_Population(Pop_0)$;
$Archive \leftarrow Pop_0(F[1])$; {the first Front is saved in an extra population : Archive}
$Q0 \leftarrow cross_op(Pop_0)$;
$Q0 \leftarrow mutate_op(Q_0, Vector)$; {Evaluate each chromosome}
for (each chromosome of Q_0) **do**
  $Evaluate(chrom, D^*, SI, D, NS)$;
end for
$Iter \leftarrow 1$;
$Pop_t \leftarrow Pop_0$;
$Q_t \leftarrow Q_0$;
while ($Iter \leq Max\_iter$) **do**
  $R_t \leftarrow Pop_t \cup Q_t$; {Merge the current and offspring populations}
  $[Popt, F] \leftarrow Fast\_Non-Dominated\_Sort(Pop_t)$;
  $Pop_t \leftarrow Crowding\_distance(Pop_t, F)$;
  $Pop_t \leftarrow Sort\_Population(Pop_t)$;
  $Archive \leftarrow Pop_t(F[1])$; {Update the archive with the new first front obtained.}
  $Pop_t \leftarrow select(R_t, N)$; {Select N first solutions from R_t}
  $Q_t \leftarrow cross\_op(Pop_t)$;
  $Q_t \leftarrow mutate\_op(Q_t, Vector)$;
  for (each chromosome of Q_t) **do**
    $Evaluate(chrom, D^*, SI, D, NS)$;
  end for
  $Iter \leftarrow Iter + 1$;
end while

Algorithm 2 Evaluation function

Require: c: chromosome; SI: sensitive itemsets; D^*: projected database; D: original database; NS set of non-sensitive frequent itemsets.

Ensure: S_F_H, N_S_L, Dis.

1: $Trans_c \leftarrow \{T_{id} \in D^* | Id$ is an identifier of a transaction coded in $c\}$;
2: $D' \leftarrow D$;
3: $Dis \leftarrow 0$;
4: **for** (each transaction t in $Trans_c$) **do**
5: $U \leftarrow \{s | s \subseteq t$ and $s \in SI\}$;
6: $Sitems \leftarrow \bigcup_{s \in U} s$;
{Determine the set of items to be removed from the transaction by applying the SCP function}
7: $Victitems_t \leftarrow set\_cover\_problem(U, Sitems)$;
{modify the database D' by removing the items of the set $victitems_t$ from transaction t.}
8: $D' \leftarrow Modif\_database(Victitems_t, D')$;
{Update the support value in D' of each frequent itemset}
9: **for** (each X in L) **do**
10: **if** $((X \subseteq t)$ and $(X \cap victitems_t \neq \emptyset))$ **then**
11: $Support_X \leftarrow Support_X - \left(\frac{1}{|D|}\right)$;
12: **end if**
13: **end for**
{Update the value of the dissimilarity between D and D'}
14: $Dis \leftarrow Dis + |Victitems_t|$;
15: **end for**
16: $S\_F\_H \leftarrow 0$;
17: $N\_S\_L \leftarrow 0$;
18: **for** (each X in L) **do**
19: **if** $(X \in Si$ and $Support_X \geq MST)$ **then**
20: $S\_F\_H \leftarrow S\_F\_H + 1$; {One more sensitive frequent itemset in D which remains frequent in D'}
21: **end if**
22: **if** $(X \notin Si$ and $Support_X < MST)$ **then**
23: $N\_S\_L \leftarrow N\_S\_L + 1$; {One more non-sensitive frequent itemset in D which becomes non-frequent in D'}
24: **end if**
25: **end for**
26: Return S_F_H, N_S_L, Dis;

Algorithm 3 Greedy Set Cover Algorithm

Require: U : a finite set of sensitive itemsets; $Sitems$: the set of items appearing in U.
Ensure: $Victimitems$: the set of victim items to be removed.

1: **for** (each item i in $Sitems$) **do**
2: $F_i \leftarrow \{s \in U | i \in s\}$;
3: **end for**
4: $Victimitems \leftarrow \emptyset$;
5: $W \leftarrow U$;
6: **while** (W is not empty) **do**
7: Select item i from $Sitems$ which maximizes $|F_i \cap W|$;
8: $Victimitems \leftarrow Victimitems \cup \{i\}$;
9: $Sitems \leftarrow Sitems - \{i\}$;
10: $W \leftarrow W - F_i$;
11: **end while**
12: Return $Victimitems$;

Now, let's explore the task of identifying the smallest subset of items, referred to as victim items, to eliminate from a specified sensitive transaction. The objective is to ensure that at least one item is removed from every sensitive itemset within this transaction. Essentially, after the removal of the victim items, the transaction should no longer contain any sensitive itemset. Remarkably, this problem perfectly aligns with the Set Cover Problem (SCP), a well-known NP-hard combinatorial challenge. Fortunately, a polynomial-time greedy algorithm, denoted as GSCA (Greedy Set S-cover Algorithm), proves to be effective in providing good approximate solutions for the SCP.

Our instance of the Set Cover Problem (SCP) involves a universe of objects denoted as U, which corresponds to the set of all sensitive itemsets included in transaction t. The set of collections, denoted as F, encompasses a collection F_i for each item i that is present in at least one sensitive itemset included in t. The elements of F_i consist of the sensitive itemsets included in t and containing the item i. The pseudo code for the Greedy Set S-cover Algorithm (GSCA) is presented in Algorithm 3.

In Algorithm 3, the procedure begins by establishing an association for each item i in $Sitems$ with a collection F_i, which contains the sensitive itemsets of U that include i (lines 1-3). In our context, the entire set of collections F_i for all values of i in $Sitems$ forms the set F of collections, as discussed in Section 4.2.2 regarding the general definition of the Set Cover Problem (SCP). In our case, the collections of F are indexed by items in $Sitems$, indicating that the presence of a collection F_i in the SCP solution implies that i is one of the victim items to be removed.

The set $Victimitems$, initially set to an empty set (line 4), is designed to hold the solution. In other words, the minimal set of items to be removed, represented by the indexes of the collections in F that constitute the solution to the SCP. The set W is initialized with the set U, encompassing all the sensitive itemsets to be covered (line 5). Subsequently, while there are still sensitive itemsets in W that have not been covered yet (line 6), the GSCA algorithm iteratively performs the following steps:

1. Select an item i from $Sitems$ in a way that the corresponding collection F_i covers the maximum number of elements in W that have not been covered yet (ine 7).

2. Include i in the set $Victimitems$ (line 8) and eliminate i from $Sitems$; additionally, remove F_i from W (lines 9-10).

Once all elements of U are covered (resulting in an empty W), the algorithm returns the set of acquired victim items (as indicated in line 12).

4.4.1 Computational complexity

The proposed NSGAII4ID algorithm follows the same general process as NSGAII [126] but integrates a greedy algorithm to address the SCP at the evaluation level. Let's denote the population size by N, the number of objectives by M, the size of a chromosome by C, the number of distinct items by T, and the number of sensitive itemsets by S.

During each iteration of the NSGAII4ID algorithm, the main operations involve the three primary operations considered in NSGAII. However, there is also a fourth operation that entails evaluating the N chromosomes of population Q_t (lines 37-39 in Algorithm 1). As demonstrated by [126], the worst-case complexity of the first three operations is:

- Fast_Non-dominated_Sort (line 30 of Algorithm 1) : $O(M \times N^2)$
- Crowding_distance (line 31 of Algorithm 1) : $O(M \times N \times log(N))$
- Sort_Population (line 32 of Algorithm 1) : $O(N \times log(N))$

The fourth operation evaluates N chromosomes, with each chromosome encoding at most C transactions subjected to the greedy SCP algorithm. It has been established (refer to [1]) that for a universe U and a collection F of subsets, the complexity of the greedy SCP algorithm is

$O(|U| \times |F| \times \min(|U|, |F|))$. In our context, the universe comprises sensitive itemsets, with its size capped at S, and each subset of the collection F is indexed by an item. Consequently, in the worst case, the size of the collection F is T. Thus, the complexity of one application of the greedy SCP algorithm on one transaction is $O(S \times T \times \min(S,T))$, and the complexity of evaluating the entire population is then $O(N \times C \times S \times T \times \min(S,T))$.

Therefore, the overall complexity is $O(M \times N^2 + N \times C \times S \times T \times \min(S,T))$. If we assume that $S < T$ (the number of sensitive itemsets is less than the total number of items), the complexity can be expressed as $O(M \times N^2 + N \times C \times T \times S^2)$, signifying that the complexity of NSGAII4ID is:

- Quadratic with respect to the population size N and the number of sensitive itemsets S;
- Linear with respect to the number of objective functions M, the size of a chromosome C, and the number of items T.

4.5 Illustrative example

To better explain how the evaluation algorithm is applied, this Section presents a detailed example.

Table 4.1: The original database D

TID	Items
1	$\{i_1, i_3, i_4\}$
2	$\{i_2, i_3, i_5\}$
3	$\{i_1, i_3, i_5\}$
4	$\{i_2, i_5\}$
5	$\{i_1, i_2, i_3, i_5\}$

Table 4.1 shows a database D containing five transactions. Suppose that the set of sensitive frequent itemsets is $SI = \{s_1, s_2\}$ where $s_1 = \{i_1, i_3\}$ and $s_2 = \{i_2, i_3\}$. Table 4.2 shows the projected database D^* which contains all transactions of D that include at least one sensitive itemset from SI. Assume that the user sets the minimum support threshold MST to 20%. Table 4.3 then shows the set L of frequent itemsets extracted from the original database D.

The Evaluation algorithm is applied on each chromosome (potential solution) of the popu-

Table 4.2: The projected database D^*

TID	Items
1	$\{i_1, i_3, i_4\}$
2	$\{i_2, i_3, i_5\}$
3	$\{i_1, i_3, i_5\}$
5	$\{i_1, i_2, i_3, i_5\}$

Table 4.3: Frequent itemsets in D and their supports

1-Itemsets	Supp	2-Itemsets	Supp	3-Itemsets	Supp	4-Itemsets	Supp
$\{i_1\}$	60%	$\{i_1, i_2\}$	20%	$\{i_1, i_2, i_3\}$	20%	$\{i_1, i_2, i_3, i_5\}$	20%
$\{i_2\}$	60%	$\{i_1, i_3\}$	60%	$\{i_1, i_2, i_5\}$	20%		
$\{i_3\}$	80%	$\{i_1, i_4\}$	20%	$\{i_1, i_3, i_4\}$	20%		
$\{i_4\}$	20%	$\{i_1, i_5\}$	60%	$\{i_1, i_3, i_5\}$	40%		
$\{i_5\}$	80%	$\{i_2, i_3\}$	40%	$\{i_2, i_3, i_5\}$	40%		
		$\{i_2, i_5\}$	60%				
		$\{i_3, i_4\}$	20%				
		$\{i_3, i_5\}$	60%				

lation generated randomly from D^*, where each gene of a chromosome represents a transaction from D^*. Here is an example of a potential solution of the population.

Chromosome :	5	0	0	2	0	1

We see that there are three candidate transactions 5, 2 and 1 encoded in this chromosome. From each of these three transactions, we have to find a minimal set of victim items to be removed. Then, the three objective functions can be calculated based on the removed items. Let us consider the evaluation process step by step:

Step 1. For each TID in the chromosome, we identify the corresponding transaction from D^*. Then we determine the set U of its sensitive itemsets. As shown in Table 4.4, transaction 5 contains the two sensitive itemsets while transaction 2 and transaction 1 each contain one sensitive itemset.

Table 4.4: Transactions represented in the solutions

TID	Transaction	Sensitive itemsets	Victim items
5	i_1, i_2, i_3, i_5	$\{i_1, i_3\}$, $\{i_2, i_3\}$	$\{i_3\}$
2	i_2, i_3, i_5	$\{i_2, i_3\}$	$\{i_2\}$
1	i_1, i_3, i_4	$\{i_1, i_3\}$	$\{i_1\}$

Step 2. For each transaction t of the chromosome, the designed algorithm performs the following actions:

1. Apply the greedy SCP algorithm to find the set of victim items to be removed. Let us take for instance the transaction having TID = 5:

   - The universe is $U = \{s_1, s_2\}$ where $s_1 = \{i_1, i_3\}$ and $s_2 = \{i_2, i_3\}$.
   - $Sitems = \{i_1, i_2, i_3\}$ is the union of items included in the subsets of U.
   - U and $Sitems$ are given as input to the greedy SCP function (Algorithm 3). We can verify that the returned set of victim items is $\{i_3\}$.

   Table 4.4 shows the sets of victim items for each transaction of the chromosome [1].

2. Update the modified database D' by removing the found victim items from the current transaction t, update the supports in D' of itemsets that are frequent in D and update the dissimilarity Dis between D and D' by adding the number of victim items of the current transaction t.

At the end of this step, the modified database D' is obtained (see Table 4.5) by removing from D, the set of victim items $\{i_1\}$ (resp. $\{i_2\}$, $\{i_3\}$) from the transaction having TID 1 (resp. 2, 5).

The final value for the dissimilarity measure is obtained by adding the values of the cardinals of the victim items sets obtained for the three transactions having the TIDs 1, 2 and 5. Thus, we find that $Dis = 3$.

[1] For transactions having TID = 2 and TID = 5, the GSCA could have returned another minimal set of victim items which is $\{i_3\}$.

Table 4.5: The modified database D'

TID	Items
1	$\{i_3, i_4\}$
2	$\{i_3, i_5\}$
3	$\{i_1, i_3, i_5\}$
4	$\{i_2, i_5\}$
5	$\{i_1, i_2, i_5\}$

Step 3. Next, the algorithm calculates the values of the two other objective functions (S_F_H and N_S_L) :

1. S_F_H is the number of sensitive frequent itemsets (in D) that remain frequent in D'. We have:

$$Support_{D'}(\{i_1, i_3\}) = 20\% \text{ and } Support_{D'}(\{i_2, i_3\}) = 0\%$$

   We see that only $\{i_1, i_3\}$ remains frequent in D' and hence, $S\_F\_H = 1$.

2. N_S_L is the number of non-sensitive frequent itemsets (in D) that become non frequent in D'. We can check that there are 6 such itemsets (see Table 4.6) and hence $N\_S\_L = 6$.

Table 4.6: Non-sensitive itemsets that are frequent in D but not in D'

Itemset	Support in D'
$\{i_1, i_4\}$	0%
$\{i_2, i_3\}$	0%
$\{i_1, i_2, i_3\}$	0%
$\{i_1, i_3, i_4\}$	0%
$\{i_2, i_3, i_5\}$	0%
$\{i_1, i_2, i_3, i_5\}$	0%

In summary, the values of the three objective functions for the chromosome are: $S\_F\_H = 1$, $N\_S\_L = 6$ and $Dis = 3$.

4.6 Experiment Results

To evaluate the effectiveness and solution quality of the proposed Non-dominating Sorting Genetic Algorithm II for item deletion (NSGAII4ID), we conducted experiments comparing its performance with four algorithms known for delivering superior results in concealing sensitive itemsets in previous studies.

Among these algorithms, NSGA2DT and GMPSO stand out as prominent evolutionary multi-objective optimization techniques utilized for hiding sensitive itemsets. Both algorithms accomplish their objective by removing entire transactions from the dataset.

- The NSGA2DT algorithm, introduced by [122], is based on the multi-objective NSGA-II algorithm. However, in contrast to our proposed algorithm, NSGA2DT conceals sensitive itemsets by eliminating entire transactions.
- The GMPSO algorithm, presented by [123], utilizes the grid multi-objective Particle Swarm Optimization (PSO) technique to minimize four side effects. Similar to NSGA2DT, GMPSO conceals sensitive itemsets by removing entire transactions from the database.

It is crucial to subject the proposed NSGAII4ID algorithm to a comprehensive evaluation, particularly in comparison with other cutting-edge algorithms that share a similar multi-objective approach. By analyzing runtime, memory costs, and various side effects—such as hiding failure, missing cost, and dissimilarity—you can holistically assess the performance of NSGAII4ID across different dimensions. This thorough evaluation serves to uncover the strengths and weaknesses of each algorithm, offering valuable insights into their applicability and efficiency in diverse scenarios.

In this experimental study, NSGAII4ID was compared with two other algorithms, namely PSO2DT [116] and sGA2DT [115]. Distinguished by a single-objective optimization approach, both PSO2DT and sGA2DT integrate three objectives into a fitness function, aggregated using three parameters (α, β, and γ). These algorithms frame the problem as a single-objective optimization task and employ evolutionary algorithms to search for a solution, with PSO2DT utilizing a PSO algorithm and sGA2DT employing a genetic algorithm [116, 115]. This comparison facilitates a comprehensive investigation into the performance disparities between multi-objective and single-objective optimization strategies, offering valuable insights into their re-

spective efficiencies and effectiveness in the realm of privacy-preserving data mining.

We conducted a comprehensive comparative analysis, evaluating the runtime, memory consumption, and the dissimilarity side effect—quantified by the number of removed items—across our proposed algorithm, NSGAII4ID, and two single-objective algorithms, PSO2DT [116] and sGA2DT [115]. While it is commonly anticipated that single-objective algorithms generally demonstrate lower time and memory costs than multi-objective ones, our evaluation extended beyond these metrics. Significantly, we found it imperative to assess NSGAII4ID against single-objective algorithms concerning the degree of modifications made to the original database during the sanitization process. This emphasis aligns with the fundamental objective of our contribution, which revolves around minimizing changes to the original database. Importantly, comparing the original and sanitized databases after the execution of the single-objective algorithms provides a direct assessment of the dissimilarity side effect.

The NSGAII4ID algorithm has been implemented in Matlab R2017a, and the source code is available on GitHub at the following address: https://github.com/mira34/NSGAII4ID-algorithm. The experiments were conducted on a PC equipped with an Intel Core i3-4005U processor and 4 GB of RAM, running under the 64-bit Microsoft Windows 10 operating system.

For our experiments, we utilized four databases, three of which are real-world databases obtained from the SPMF library [128]: Mushroom, Chess, and Foodmart. It is noteworthy that the Foodmart dataset contains information not only about items in transactions but also about their purchase quantities. However, for the purposes of our experiments, quantities were disregarded as they are not necessary for the algorithms. Additionally, we employed a synthetic dataset called T10I4D100K, generated using the IBM database generator[2]. These databases were selected due to their popularity, with most recent studies on PPDM utilizing them for validation purposes. Table 4.7 presents the features of each dataset, where $\#|D|$ denotes the number of transactions, $\#|I|$ is the number of distinct items, *AvgLen* is the average transaction length, *MaxLen* is the maximal transaction length and *Type* is the type of dataset (dense or sparse).

Table 4.8 (resp. Table 4.9) gives the parameters of NSGAII4ID, NSGA2DT and GMPSO (resp. PSO2DT and sGA2DT) that have been used in the experiments.

[2]The generator may be downloaded at: https://sourceforge.net/projects/ibmquestdatagen/

Table 4.7: Features of used databases

Dataset	$\#\|D\|$	$\#\|I\|$	*AvgLen*	*MaxLen*	*Type*
chess	3196	74	37	37	Dense
mushroom	8124	119	23	23	Dense
foodmart	21556	1559	4	11	Sparse
T10I4D100K	100000	870	10.1	29	Sparse

Table 4.8: The parameters of the three multi-objective algorithms used in the experiments

Algorithms			
Parameter	NSGAII4ID	NSGA2DT	GMPSO
Max Iteration	10	10	10
Population size	10	10	10
Dimension of chromosome	defined by user (1000)	calculated by m	calculated by m
Run	10	10	10
Crossover probability	0.95	0.95	-
Mutation probability	0.05	0.05	-
nGrid	-	-	4

Table 4.9: Parameter of PSO2DT and sGA2DT

Algorithms		
Parameter	PSO2DT	sGA2DT
Max Iteration	10	10
Population size	10	10
Run	10	10
Crossover probability	-	0.95
Mutation probability	-	0.05

4.6.1 Runtime

In this experiment, we conducted a comparative analysis of the execution time of the proposed NSGAII4ID algorithm against two multi-objective algorithms, NSGA2DT and GMPSO, as well as two single-objective algorithms, PSO2DT and sGA2DT. The comparison was performed for each of the four databases, and the results are presented in Figure 4.5.

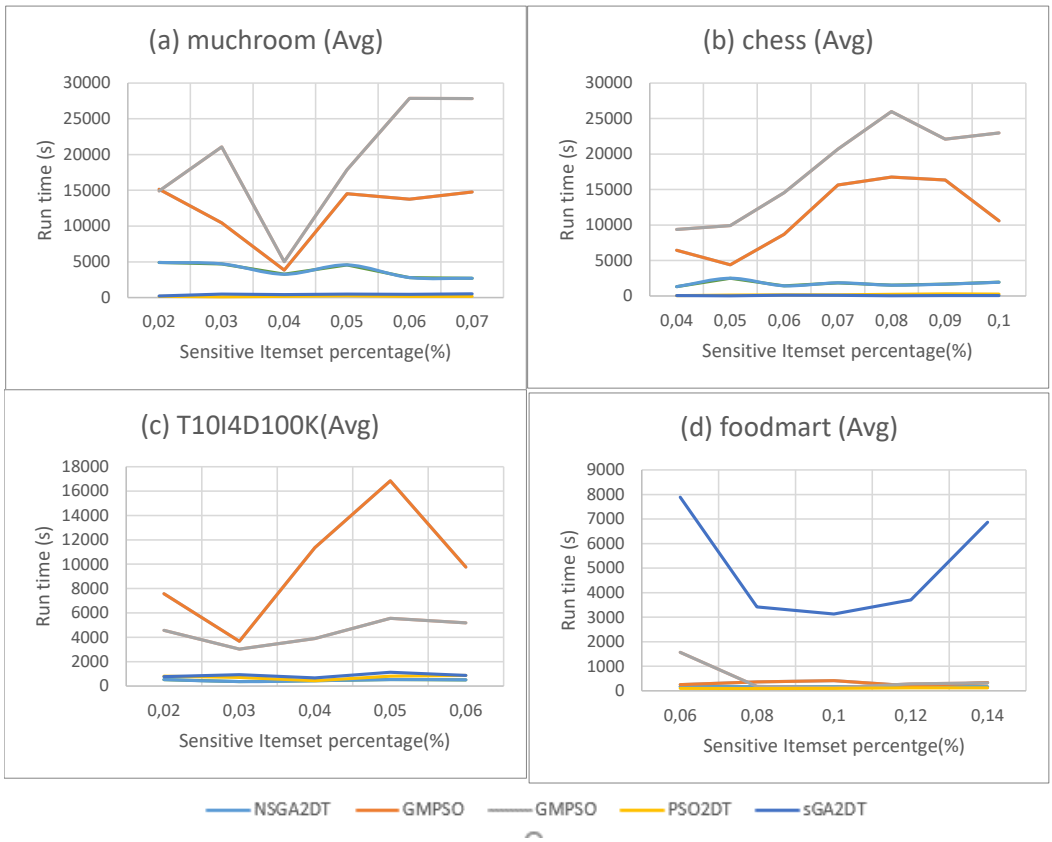

Figure 4.5: Runtime of five algorithms for four databases

Figure 4.5(a) illustrates the average runtime for two executions with different minimum support (MSP) thresholds (40% and 55%) on the Mushroom database. Five algorithms, namely NSGAII4ID, NSGA2DT, GMPSO, PSO2DT, and sGA2DT, were employed for the comparison. As anticipated, the single-objective algorithms, PSO2DT and sGA2DT, exhibit faster runtimes than the multi-objective algorithms. Notably, NSGAII4ID outperforms NSGA2DT and GMPSO in terms of runtime, with execution times consistently below 5000 seconds. In contrast, GMPSO records the highest execution time, peaking at 27000 seconds for a sensitive itemset percentage of 7%. NSGA2DT falls in between the other algorithms, with a maximum execution time of 15000 seconds.

Figure 4.5(b) presents the average runtime for the Chess database, considering two MSP threshold values (90% and 95%). Once again, NSGAII4ID consistently outperforms the other multi-objective algorithms, maintaining an execution time around 2000 seconds for various sensitive itemset percentages. However, the single-objective algorithms achieve the best runtimes. In the case of multi-objective algorithms, a similar pattern emerges as in the Mushroom database experiment: GMPSO exhibits the highest execution time (ranging between 10000

and 26000 seconds), with NSGA2DT situated in the middle, showing execution times between 4000 and 16000 seconds.

Figure 4.5(c) and 4.5(d) display the average runtime for two sparse databases, T100I4D100K (with two MSP threshold values: 2.40% and 2.50%) and Foodmart (with two MSP threshold values: 0.30% and 0.32%), respectively. Notably, for these databases as well, NSGAII4ID consistently outperforms the other multi-objective algorithms and demonstrates competitiveness even compared to single-objective algorithms. This suggests that our algorithm is particularly efficient for sparse databases.

In summary, the results depicted in Figure 4.5 demonstrate that, in terms of runtime, the proposed NSGAII4ID algorithm clearly outperforms the other compared multi-objective algorithms, NSGA2DT and GMPSO. Moreover, it is competitive with the single-objective algorithms, PSO2DT and sGA2DT, especially for sparse databases. This result is very interesting and encouraging. Indeed, since NSGAII4ID operates at the item level, which is finer than the transaction level, one might expect that this could require more runtime than the other multi-objective algorithms operating at the transaction level. The fact that NSGAII4ID is faster indicates the efficiency of using SCP to carefully choose the items to delete, accelerating the convergence towards the solution.

4.6.2 Memory Cost

Figure 4.6 depicts the average memory costs of three multi-objective algorithms (NSGAII4ID, NSGA2DT, and GMPSO) and two single-objective algorithms (PSO2DT and sGA2DT) across four different databases.

Initially, it's apparent that the two single-objective algorithms exhibit significantly lower memory consumption compared to the three multi-objective algorithms. This observation aligns with expectations, given that multi-objective algorithms typically employ more sophisticated techniques and complex data structures. Although the memory costs for NSGAII4ID and the other multi-objective algorithms remain reasonable, the key advantage of NSGAII4ID lies in its capacity to deliver effective solutions with minimal alterations to the original dataset.

Further examination of the multi-objective algorithms' behavior regarding memory costs follows.

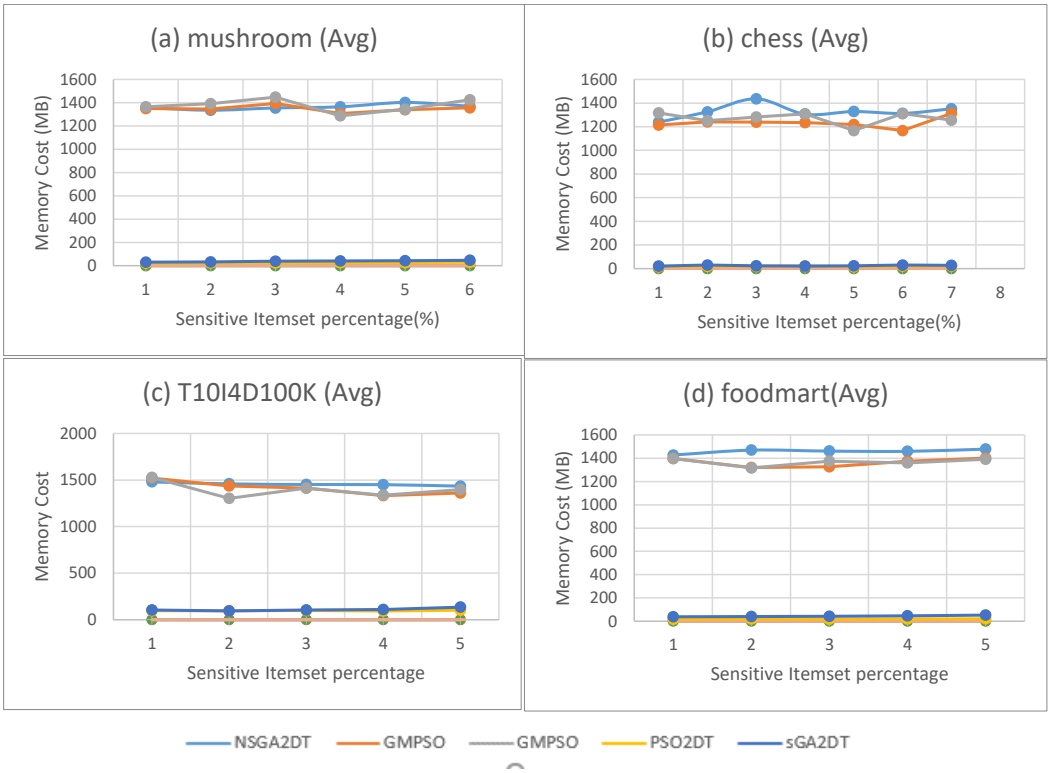

Figure 4.6: Memory Cost of five algorithms for the four databases

In Figure 4.6(a), the average memory costs in megabytes for the Mushroom database are displayed. Notably, the memory costs of NSGAII4ID remain relatively stable across different sensitive itemset percentages (between 1340 and 1400 MB). It is lower than the memory costs of the two other algorithms for sensitive itemset percentages between 2% and 4.5%, and higher for the remaining interval between 4.5% and 7%. Overall, the three algorithms demonstrate competitiveness in terms of memory usage for the Mushroom database.

Figure 4.6(b) illustrates the average memory costs for the Chess database, where the three algorithms (NSGAII4ID, NSGA2DT, and GMPSO) exhibit closely comparable memory costs, ranging between 1200 MB and 1400 MB. Thus, for the Chess database, the three algorithms demonstrate competitive performance in terms of memory usage.

Finally, Figures 4.6(c) and 4.6(d) depict the results for the T10I4D100K and Foodmart databases, respectively. In these cases, the average memory cost of NSGAII4ID is slightly higher than that of NSGA2DT and GMPSO. Overall, for dense databases, the three algorithms demonstrate competitive memory costs. However, for sparse datasets, NSGAII4ID tends to use slightly more memory than the other two algorithms. This is compensated by the superior runtime efficiency of NSGAII4ID for sparse databases, as demonstrated in the previous section.

4.6.3 Side effects

In this section, our focus centers on the comprehensive examination of three critical side effects: hiding failure, missing cost, and dissimilarity. We initiate the exploration with an experimental investigation into the internal behavior of these side effects concerning their variations in the proposed NSGAII4ID algorithm, detailed in Section 4.6.3.1. Subsequently, we delve into a meticulous comparative analysis between NSGAII4ID and other state-of-the-art algorithms, dissecting each side effect individually in the subsequent sections.

4.6.3.1 Internal behavior

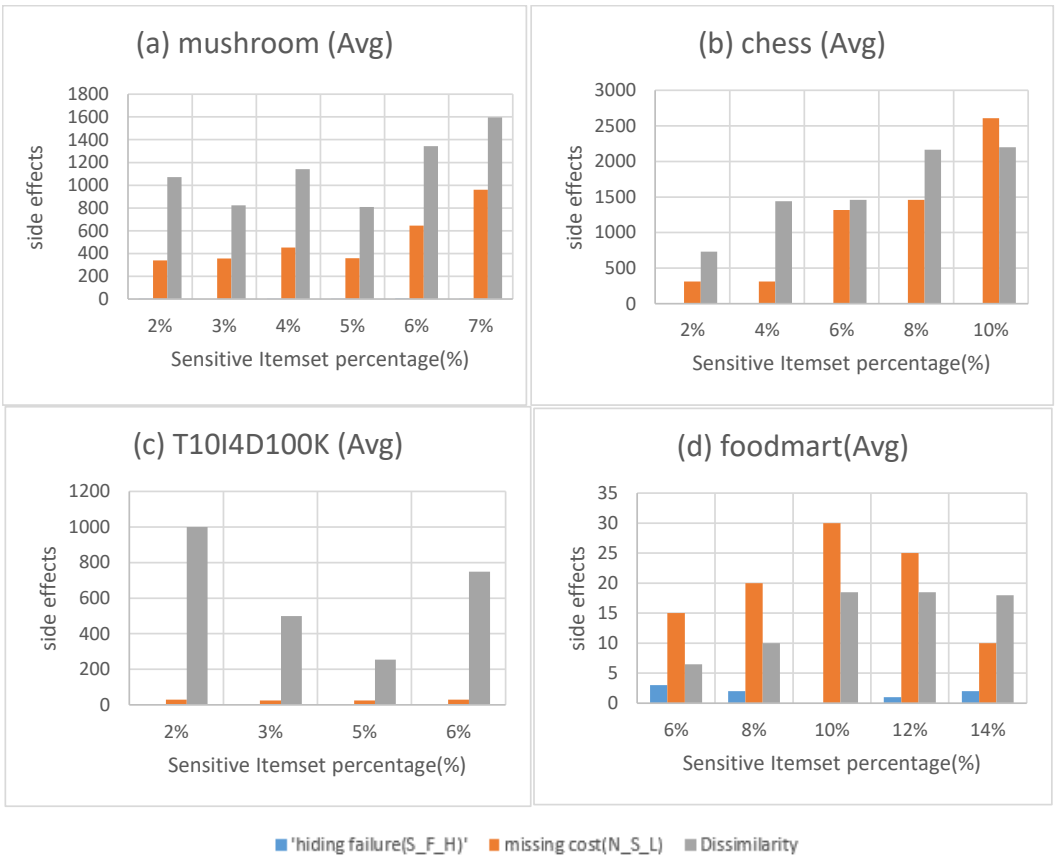

Figure 4.7: Internal behavior of NSGAII4ID algorithm for four databases

In this experiment, we evaluate the three side effects employed in the NSGAII4ID algorithm across four diverse databases. For each database, we aggregate average results for the side effects, namely hiding failure (F_T_H), missing cost (N_S_L), and dissimilarity (Dis), derived from two executions with varying minimum support thresholds (MST). The MST values used are 40% and 45% for the mushroom database, 90% and 95% for the chess database, 2.4% and 2.5% for the T10I4D100K database, and 0.30% and 0.32% for the foodmart database.

Given that multi-objective algorithms generate multiple Pareto optimal solutions, we meticulously assess each solution, extract values for the aforementioned side effects, and identify the best solution containing the minimal value for a particular side effect. The outcomes are depicted in Figure 4.7.

In the mushroom database (Figure 4.7(a)), notably low values for hiding failure are observed, indicating the algorithm's efficacy in concealing nearly all sensitive itemsets. Conversely, values for missing cost (reflecting the number of non-sensitive frequent itemsets hidden by the sanitization process) and dissimilarity (representing the total number of items removed from the original database during sanitization) exhibit similar variations and are more pronounced than hiding failure. A comparable trend is evident in Figure 4.7(b) for the chess database. This similarity in behavior can be attributed to the dense nature of both the mushroom and chess databases.

For T10I4D100K and foodmart (Figures 4.7(c) and 4.7(d), respectively), very low values for hiding failure are once again apparent, particularly for the T10I4D100K database. However, the dissimilarity values are larger in these two databases. Additionally, it is noteworthy that the missing cost is substantially lower in T10I4D100K compared to foodmart.

Overall, NSGAII4ID consistently attains outstanding results in hiding sensitive itemsets across all cases, surpassing the two other side effects (missing cost and dissimilarity). This success is attributed to the algorithm's emphasis on minimizing sensitive itemsets during the sanitization process by strategically selecting victim items to hide all sensitive itemsets from a given candidate transaction. Subsequent experiments aim to compare the behavior of each side effect between NSGAII4ID and other existing algorithms.

4.6.3.2 Comparison with respect to hiding failure(S_F_H)

Figure 4.8 presents a comparison of the average value of the hiding failure objective function (S_F_H) found by the three algorithms NSGAII4ID, NSGA2DT, GMPSO on the four databases (mushroom, chess, T10I4D100K and foodmart) for various sensitive itemset percentages. Recall that S_F_H denotes the count of sensitive itemsets in the original database that the algorithm fails to conceal in the sanitized database.

It is obvious across all considered databases that the proposed NSGAII4ID algorithm con-

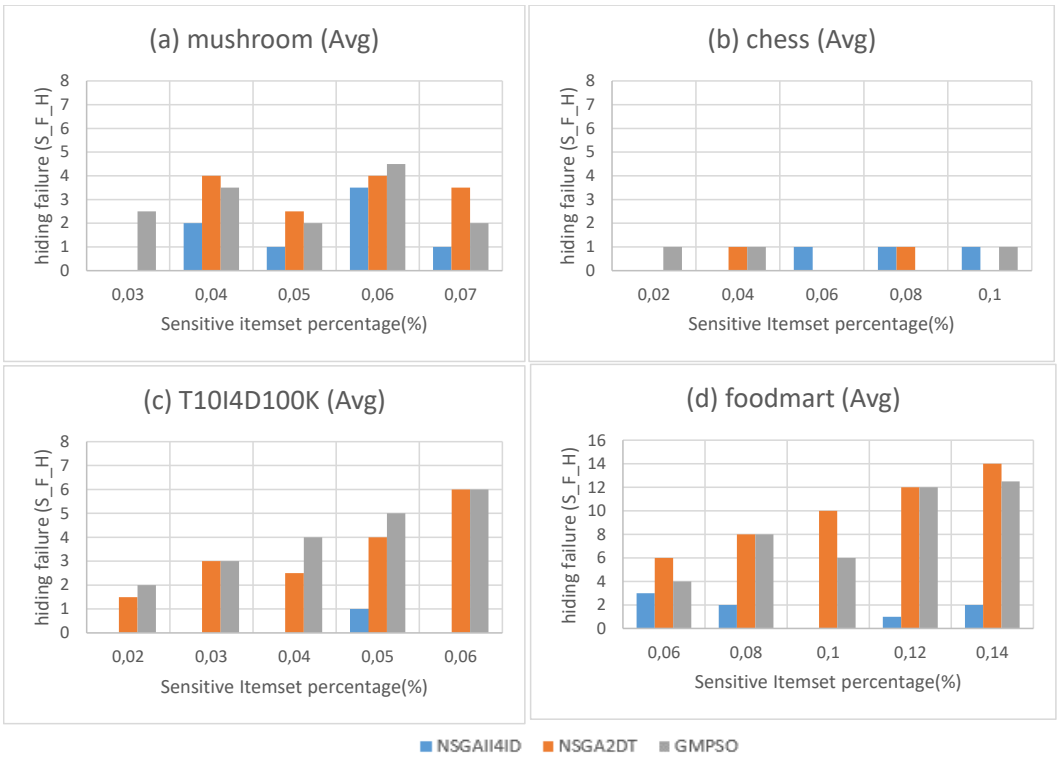

Figure 4.8: Hiding failure of three algorithms for four databases

sistently outperforms NSGA2DT and GMPSO, indicating a superior ability to hide more sensitive itemsets than the other compared algorithms. Notably, the exception is the chess database, where all algorithms exhibit similar performance. This observation underscores the effectiveness of the Set Cover Problem (SCP) in facilitating precise change operations that successfully conceal a maximum number of sensitive frequent itemsets.

Furthermore, it is noteworthy that the margin between NSGAII4ID and the other two algorithms becomes more pronounced for sparse databases, as illustrated in Figures 4.8(c) and 4.8(d). This emphasizes the algorithm's heightened efficacy in scenarios characterized by a lower density of transactions and underscores its adaptability across different database types.

4.6.3.3 Comparison with respect to missing cost(N_S_L)

Figure 4.9 provides a comparative analysis of the average values of the missing cost objective function (N_S_L) across four databases (mushroom, chess, T10I4D100K, and foodmart) for the three algorithms—NSGAII4ID, NSGA2DT, and GMPSO. The analysis considers various sensitive itemset percentages for each database. Recall that N_S_L is the number of non-sensitive itemsets in the original database that the algorithm conceals in the sanitized database.

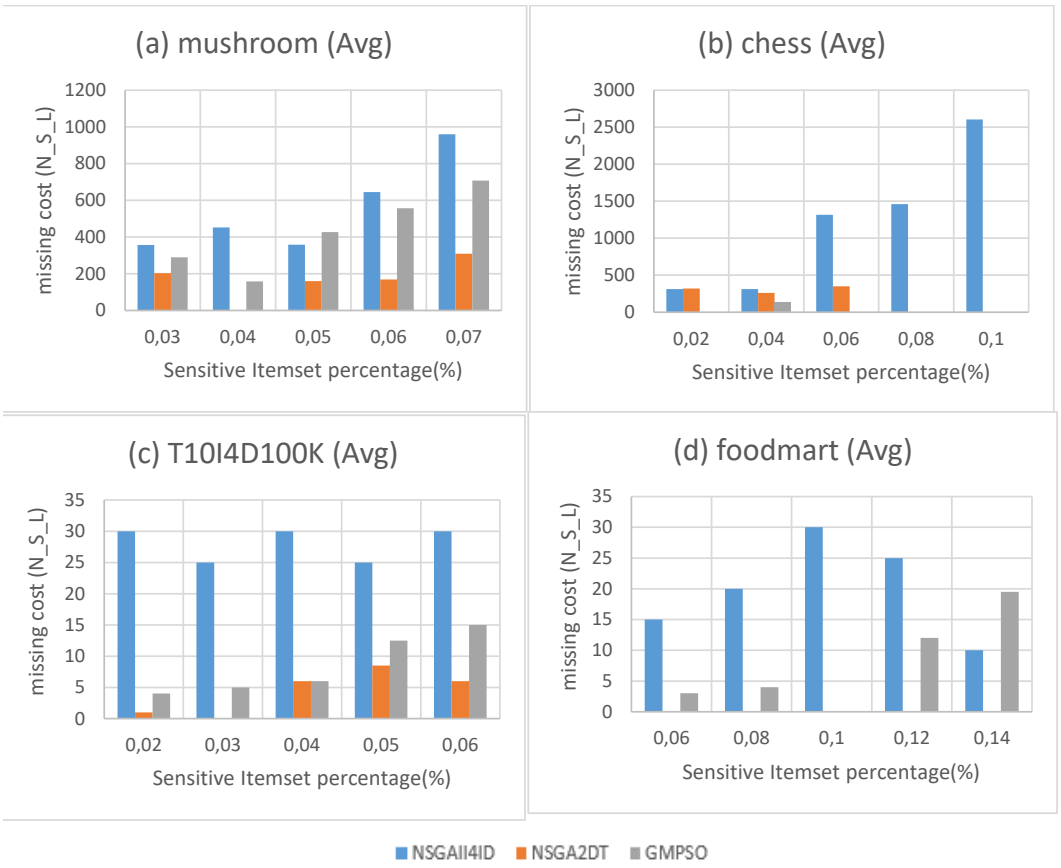

Figure 4.9: Missing cost of three algorithms for four databases

Figure 4.9 reveals that the designed algorithm does not outperform NSGA2DT and GMPSO in minimizing the missing cost. This trade-off arises from the algorithm's emphasis on concealing more sensitive frequent itemsets compared to the other algorithms. However, from a Privacy-Preserving Data Mining (PPDM) perspective, it is preferable to sacrifice some useful but non-sensitive information to avoid disclosing sensitive information, rather than the reverse scenario (retaining more useful and non-sensitive information but failing to conceal a maximum of sensitive information).

4.6.3.4 Comparison with respect to the number of removed items

In this experiment, we aim to evaluate the performance of the NSGAII4ID algorithm concerning the dissimilarity side effect, which focuses on minimizing alterations made to the original database during the sanitization process. The algorithm is compared against two single-objective algorithms, PSO2DT and sGA2DT, as well as two multi-objective algorithms, NSGA2DT and GMPSO.

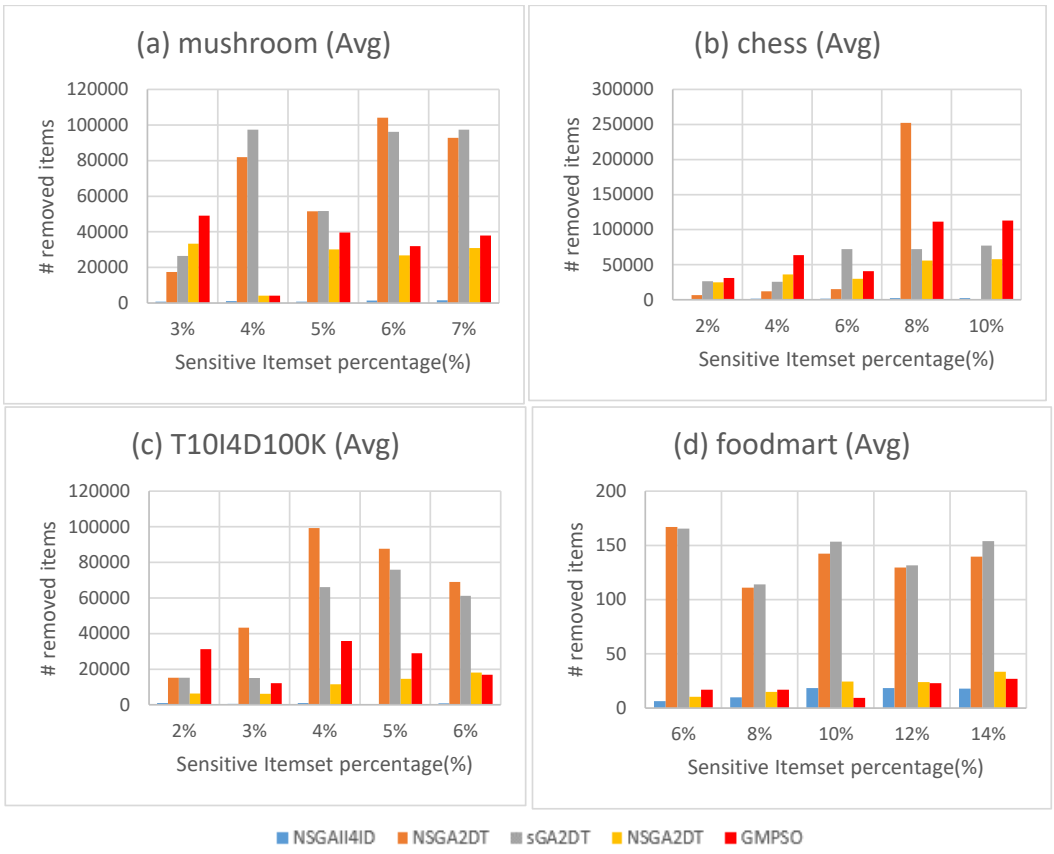

Figure 4.10: Dissimilarity of three algorithms for four databases

All four compared algorithms (NSGA2DT, GMPSO, PSO2DT, and sGA2DT) employ transaction deletion for database sanitization. The recorded value of the dissimilarity side effect for these algorithms corresponds to the total number of items contained in all removed transactions.

Figure 4.10 illustrates the average number of removed items after sanitization for the four databases and the five compared algorithms (NSGAII4ID, NSGA2DT, GMPSO, PSO2DT, and sGA2DT) across different percentages of sensitive itemsets.

The results consistently reveal that, in all cases, the NSGAII4ID algorithm removes significantly fewer items than the four other algorithms. This is attributed to the proposed algorithm selectively deleting specific items from chosen transactions, in contrast to removing entire transactions.

In summary, the overall performance of NSGAII4ID is highly satisfactory. The obtained results for runtime and memory cost are promising, showcasing accelerated convergence compared to other multi-objective algorithms at the transaction level. NSGAII4ID's runtime is closer to that of single-objective algorithms than the other multi-objective algorithms. Addi-

tionally, memory consumption for NSGAII4ID and the other multi-objective algorithms remains competitive, with expected higher values compared to single-objective algorithms but without constituting a significant limitation.

Regarding side effects, the experiments clearly demonstrate NSGAII4ID's superiority over all compared algorithms in hiding sensitive itemsets and minimizing changes to databases. Although NSGAII4ID does not outperform other multi-objective algorithms in terms of missing cost, future work can focus on minimizing this aspect further. Overall, the use of SCP at the item level in NSGAII4ID proves to be a strategic choice for achieving robust performance across multiple dimensions in privacy-preserving data mining.

4.7 Conclusion

In this chapter, a novel approach to privacy-preserving data mining (PPDM) was introduced, formulating the PPDM problem as a multi-objective optimization task with three objectives to be minimized. These objectives include the hiding of sensitive frequent itemsets, the preservation of non-sensitive frequent itemsets, and the minimization of dissimilarity between the original and modified databases. The NSGAII4ID algorithm, proposed in this context, addresses the challenge of hiding sensitive itemsets through a two-level optimization strategy. At the first level, a multi-objective genetic algorithm identifies optimal subsets of candidate transactions for modification. At the second level, a subset of items is selected from each transaction for deletion, involving the Set Cover Problem (SCP) as an optimization task. The optimization at the second level contributes to the evaluation of chromosomes at the first level.

The experimental study, conducted on four databases (two sparse and two dense), demonstrated that NSGAII4ID performs effectively in terms of execution time and minimizes the overall changes made to the original database during modification. The algorithm also achieved satisfactory results in minimizing hiding failure and missing cost side effects, with excellent results for hiding failure on all databases.

Chapter 5: Hiding sensitive expected frequent itemsets in the context of uncertain databases

5.1 Introduction

To protect sensitive itemsets, a widely adopted strategy involves altering a database to thwart their discovery through data mining. This procedure, referred to as database sanitization, commonly entails the removal or modification of sensitive transactions or records. However, such modifications may lead to unintended "side effects." With the escalating volume of data and the increasing prevalence of data mining techniques, Privacy-Preserving Data Mining (PPDM) has become a focal point of attention in recent decades. The central objective of PPDM is to find a delicate equilibrium between preserving data privacy and facilitating efficient data mining processes.

Moreover, uncertainty is pervasive in data originating from numerous real-life applications. Uncertainty in data can stem from various origins, encompassing our limited perception or comprehension of reality, constraints in observation equipment, and restrictions in available resources for data collection, storage, transformation, and analysis. Additionally, inherent characteristics of the considered data can contribute to uncertainty. For instance, data collected through sensors—whether chemical, electromagnetic, mechanical, optical, or thermal sensors in applications such as environmental surveillance, security, and manufacturing systems—may inherently contain noise [129]. Dynamic errors, including measurement inaccuracies, variations in sampling frequency, deviations caused by rapid changes in measured properties over

time (e.g., drift or noise), wireless transmission errors. In recent years, numerous models and algorithms have been developed to address various uncertain data mining tasks, including clustering, classification, and outlier detection in uncertain data.

The mining of frequent itemsets in uncertain databases has emerged as a significant topic in the data mining community, drawing the attention of numerous researchers seeking optimal strategies for this task. In this domain, the majority of existing work adopts a probabilistic framework to model data uncertainty.

In probabilistic databases, the identification of frequent itemsets relies on two primary semantic approaches [78]: Expected support-based frequent itemsets [11, 17] and probabilistic frequent itemsets [12]. Both definitions consider the support of an itemset as a discrete random variable. In the first definition [11, 15, 16, 17], the frequency of an itemset is assessed by its expected support, considering the itemset frequent if its expected support equals or exceeds a minimum expected support threshold denoted as *minesup*. The second definition [12, 18, 20] involves the computation or estimation of the "real" value of the "frequent probability" of an itemset, representing the probability that its support is not less than the *minsup* threshold. Formally, an itemset is considered frequent if its frequent probability is greater than or equal to a given probabilistic threshold, referred to as *minprob*. The expectation measure extends the definition of frequent itemsets in traditional databases, while the definition of probabilistic frequent itemsets encompasses the complete probability distribution of the support of an itemset. Despite these distinctions, both definitions share a close connection. Specifically, the support of an itemset is modeled as a random variable following the Poisson Binomial distribution, where the expected support of an itemset corresponds to the expectation of this random variable. Consequently, computing the frequent probability of an itemset is equivalent to determining the cumulative distribution function of this random variable.

There are two expected-support-based approaches for mining frequent itemsets in uncertain databases. The first approach, known as U-Apriori [16], extends the Apriori algorithm to the probabilistic setting. It extracts expected support-based frequent itemsets, significantly reducing the size of the candidate set. However, its performance is affected as it scans the uncertain database multiple times. The second approach builds upon tree-based algorithms, such as FP-Growth, and introduces UFP-Growth [17], which generalizes FP-Growth to the case of probabilistic databases. UFP-Growth scans the database twice and uses UFP-trees with mega-

nodes, where a mega-node includes nodes for items with similar existential probability. Despite its benefits, it contains a distinct tree path for each distinct item-existential probability pair.

In terms of PPDM, the increasing use of data mining techniques has raised concerns about privacy and information security. PPDM aims to protect sensitive information from unauthorized disclosure by developing algorithms that modify the original database to keep private, confidential, or sensitive data and knowledge secure. Various approaches in the existing literature differ in their assumptions about the data collection model and user privacy requirements.

However, modifications made during the sanitization process can lead to undesirable side effects, such as "Hiding failure," which represents the number of sensitive frequent itemsets that remain uncovered even after sanitization. "Missing cost" is the number of non-sensitive frequent itemsets lost after the modification of the database, and "dissimilarity" is the number of items/transactions removed from the database. Solving this problem is challenging because it is an NP-hard optimization problem to simultaneously minimize these side effects while hiding sensitive frequent itemsets.

In addition to exact methods, researchers have explored heuristic and metaheuristic approaches to address this challenge. Notably, all existing approaches for PPDM assume certain data. In this chapter, we focus on preserving sensitive frequent itemsets in uncertain databases, proposing a novel algorithm named U-NSGAII4ID (Uncertain-Non-dominated Sorting Genetic Algorithm for Item Deletion). This algorithm hides sensitive expected frequent itemsets by solving a multi-objective optimization problem, where the solution encodes a set of items to remove from a selected set of transactions. The main contributions of the proposal U-NSGAII4ID algorithm are the following:

1. The chapter presents an innovative approach to address the challenge of Privacy-Preserving Data Mining (PPDM) in uncertain transactional databases. The frequency of itemsets is determined by the concept of expected-support, where itemsets with an expected-support equal to or exceeding a specified threshold are identified as frequent. The proposed solution employs U-Apriori, an extended version of the Apriori algorithm tailored for probabilistic databases, utilizing the expected support measure. This adaptation allows for effective handling of uncertainty in transactional databases during the privacy-preserving data mining process.

2. We model the privacy peserving frequent itemset mining problem over uncetain transaction databases as a multi-objective optimization problem and we adapt NSGA-II algorithm [126] to solve it. The solution encodes the set of transactions from which some items should be removed.

3. To evaluate each solution encoded in the genetic algorithm and representing a set of transactions to update, a set of items to remove is determined from each selected transaction. The problem of finding the set of optimal items to remove from a given transaction is modeled as a Weighted SCP which is a well-known NP-Hard combinatorial problem. An approximate polynomial-time greedy algorithm is used to solve this problem.

4. The dissimilarity measure is computed as the sum of probability values of removed items after the sanitization process. this takes into account the probability information and captures the idea that the higher is the probability value associated with an item, the more expensive is its removing from the database.

The remaining of this chapter is organized as follow: Section 5.2 gives basic definitions related to our problem in the particular context of uncertain databases. Section 5.3 is devoted to the presentation of the proposed U-NSGAII4ID sanitization algorithm in uncertain databases. In Section 5.4, we report and discuss experiment results. Finally, Section 5.5 concludes the chapter.

5.2 Basic definitions for mining frequent itemsets from uncertain databases

In this section, we provide some fundamental definitions related to the uncertain database model and the problem of mining frequent itemsets over uncertain databases. The definitions of some concepts are the same as those introduced in the previous chapter, but we prefer to recall them here to make the chapter as self-contained as possible.

Let $I = x_1, x_2, \ldots, x_m$ be a set of m items. Any subset X of I is referred to as an itemset, and it is considered to be an l-itemset if $|X| = l$. A certain transaction consists of a set of items, and a certain transactional database is composed of N transactions.

In the uncertain model, each item in a transaction is associated with a non-zero probability of its existence in the transaction.

Definition 5.2.1 (Uncertain Transaction) *An uncertain transaction t contains a set of items (an itemset) where each item $x \in t$ is associated with its probability $Pr(x \in t)$ to appear in this transaction ($0 < Pr(x \in t) \leq 1$).*

Notice that in the case where $Pr(x \in t) = 1$, this means that the item x is a certain item in transaction t, i.e., it surely appears in t; and if $Pr(x \in t) = 0$ then surely x does not belong to t and it is not explicitly represented in t.

Definition 5.2.2 (Uncertain Database) *An uncertain transaction database (UDB) is a set $D = \{t_1, t_2, \ldots, t_N\}$ such that each t_i is an uncertain transaction.*

Definition 5.2.3 (Expected support-based frequent itemsets) *. Let us consider an uncertain database D with N transactions and let minsup be a minimum expected support threshold. An itemset X is considered an expected support-based frequent itemset if and only if $ExpSup(X) \geq$ minsup, where the expected support is determined by the following equation:*

$$ExpSup(X) = \frac{\sum_{t \in D} Pr(X \subseteq t)}{|D|} = \frac{\sum_{t \in D} (\prod_{x \in X} Pr(x \in t))}{|D|} \tag{5.1}$$

The set of expected support-based frequent itemsets is denoted by F.

It is worth noticing that the complexity of computing the expected support-based frequent itemset is $O(N)$.

Definition 5.2.4 (sensitive and non-sensitive expected frequent itemset) *Sensitive expected frequent itemsets are itemsets selected by the user and having an expected support which is greater than or equal to the threshold minsup. They represent sensitive information to be hidden. This set is denoted by SI. The set of reminding expected frequent itemsets is called the set of non-sensitive expected frequent itemsets and it corresponds to $L \setminus SI$.*

To hide sensible itemsets, the sanitization process consists in performing a minimum change (here removing some transactions and/or items) on the database D. Let D' denote the modified database and L' denote the new set of expected support-based frequent itemsets obtained from D'. There are three side effects that have to be minimized:

Definition 5.2.5 (Side effects) .

- *The "hiding failure" (S_F_H) is the number of sensitive itemsets that the sanitization process fails to hide. It is given by:* $S\_F\_H = |SI \cap L'|$.
- *The "missing cost" (N_S_L) is the number of non-sensitive frequent itemsets that are hidden (become infrequent) after the sanitization process. This number is given by:* $N\_S\_L = |(L - SI) - L'|$.
- *The dissimilarity measure (Dis) between the original uncertain database D and the modified database* D' *obtained by sanitization process. It can be defined by the total number of either items or transactions removed from D to obtain* D'.

5.3 The Proposed U-NSGAII4ID algorithm

This section introduces the proposed sanitization algorithm for uncertain databases, named Uncertain Non-dominated Sorting Genetic Algorithm II for Item Deletion (U-NSGAII4ID). Subsection 5.3.1 provides a description of the U-NSGAII4ID algorithm, followed by a detailed illustrative example of its application in Section 5.3.2.

The U-NSGAII4ID algorithm, an extension of the NSGA-II algorithm [126], is a multi-objective optimization approach tailored for privacy-preserving data mining and aiming at minimizing side effects (hiding failure, missing cost, and dissimilarity) in uncertain databases by identifying an optimal set of probabilistic items to be removed from sensitive transactions.

In this algorithm, a potential solution encodes a set of victim transactions, indicating transactions from which items should be removed in the context of multi-objective optimization. The evaluation process for a solution entails determining the optimal subset of probabilistic items to be removed from each transaction. This step is crucial for computing the objective function value associated with the solution. Importantly, the task of identifying the optimal subset of items to remove poses an inherent NP-Hard problem, specifically the well-known Weighted Set Cover Problem (WSCP). To efficiently address this challenge during the evaluation of potential solutions, a polynomial greedy algorithm is employed.

The detrmination of victim transactions is carried out iteratively, with each iteration involving the evaluation of a population of potential solutions. Each individual solution of the

population is termed a chromosome. In the specific context of this study, a chromosome is represented as a vector of transaction identifiers (TID) extracted from an uncertain database, indicating the selected victim transactions. All potential solutions within the population share a fixed dimension, denoted as *nVar* and determined by the user. Each position within a chromosome is referred to as a gene. Here is an example of a chromosome:

152	20	325	0	15	0

This chromosomes has a length of six and encodes four transactions having the TIDs: 152, 20, 325 and 15. The 0 value indicates that no transaction is encoded in the corresponding gene.

5.3.1 Algorithm description

The pseudo-code for the U-NSGAII4ID algorithm is outlined in Algorithm 4. As previously mentioned, U-NSGAII4ID adheres to the general framework of NSGA-II and incorporates a step to address the Weighted Set Cover Problem (WSCP) during the chromosome evaluation.

5.3.1.1 The main algorithm.

U-NSGAII4ID takes several elements as input, including the original uncertain database (D), the set of expected frequent itemsets (L) obtained using a suitable algorithm (here, U-Apriori is utilized, but alternatives like UFP-Growth and UH-Mine can be considered), the user-defined minimum support threshold ($minesup$), the set of sensitive expected frequent itemsets (SI) which is a subset of L, the population size (N), the length of a potential solution ($nVar$), and the maximum number of iterations (Max_iter). These values, namely N, $nVar$, and Max_iter, are parameters specified by the user for the algorithm.

The algorithm starts by projecting all sensitive transactions into a separate database, denoted as D^*, extracted from the original database D (lines 2-8). Specifically, it identifies every transaction in D that contains at least one sensitive itemset and includes it in the D^* database.

The population consists of N chromosomes, each representing a potential solution. In this context, a chromosome is a set of genes, where each gene corresponds to the Transaction ID (TID) of a sensitive uncertain transaction from the projected database D^*. Chromosomes are randomly initialized (lines 9-12) using the Roulette wheel selection technique.

Algorithm 4 Pseudo-code of U-NSGAII4ID algorithm

Require:

D: Original uncertain Database; L: Set of expected frequent itemsets; $minesup$: Minimum expected support threshold; SI: Set of sensitive expected frequent itemsets; N: Population size; $nVar$: Chromosome size; Max_iter: Maximum number of iterations;

Ensure:

Set of optimal solutions to be modified (Archive);

{Construction of the projected database containing uncertain transactions including at least one sensitive expected frequent itemset}

$D^* \leftarrow \emptyset$;
for (Each transaction T_q in D) **do**
  for (Each sensitive itemset s in SI) **do**
    if ($s \subseteq T_q$) **then**
      $D^* \leftarrow D^* \cup \{T_q\}$;
    end if
  end for
end for

{Initialization of the population}

for (Each solution i in population) **do**
  $Vector \leftarrow Roulette\_Wheel(D^*)$;
  $Pop_0[i] \leftarrow rand(Vector, N, nVar, null)$;
end for

{Evaluation of each chromosome : S_F_H, N_S_L, Dis are calculated within the evaluation function and each of them receives a discrete value}

for (each chromosome *chrom* of Pop_0) **do**
  $Prob\_Evaluate(chrom, D^*, SI, D, NS)$;
end for
$[Pop_0, F] \leftarrow Fast\_Non-Dominated\_Sort(Pop_0)$;
$Pop_0 \leftarrow Crowding\_distance(Pop_0, F)$;
$Pop_0 \leftarrow Sort\_Population(Pop_0)$;
$Archive \leftarrow Pop_0(F[1])$; {the first Front is saved in an extra population : Archive}
$Q_0 \leftarrow cross\_op(Pop_0)$;
$Q_0 \leftarrow mutate\_op(Q_0, Vector)$;

{Evaluate each chromosome}

for (each chromosome of Q_0) **do**
  $Prob\_Evaluate(chrom, D^*, SI, D, NS)$;
end for
$Iter \leftarrow 1$;
$Pop_t \leftarrow Pop_0$;
$Q_t \leftarrow Q_0$;
while ($Iter \leq Max\_iter$) **do**
  $R_t \leftarrow Pop_t \cup Q_t$; {Merge the current and offspring populations}
  $[Popt, F] \leftarrow Fast\_Non-Dominated\_Sort(Pop_t)$;
  $Pop_t \leftarrow Crowding\_distance(Pop_t, F)$;
  $Pop_t \leftarrow Sort\_Population(Pop_t)$;
  $Archive \leftarrow Pop_t(F[1])$; {Update the archive with the new first front obtained}
  $Pop_t \leftarrow select(R_t, N)$; {Select N first solutions from R_t}
  $Q_t \leftarrow cross\_op(Pop_t)$;
  $Q_t \leftarrow mutate\_op(Q_t, Vector)$;
  for (Each chromosome of Q_t) **do**
    $Prob\_Evaluate(chrom, D^*, SI, D, NS)$;
  end for
  $Iter \leftarrow Iter + 1$;
end while

Objective functions (fitness values) are then evaluated for each solution (lines 13-15). Three objective functions are assessed for each chromosome using the *Prob_Evaluate* function, detailed in Algorithm 5. Subsequently, the fast non-dominated sorting and crowding distance methods are applied to the population, ranking solutions from the best to the worst (lines 16-18). An external population, referred to as the "archive" (line 19), is employed to store the first rank obtained from previous steps.

The crossover operation is applied to two selected chromosomes by randomly choosing a gene in each chromosome and exchanging them. If a gene becomes redundant in the new chromosome, it is replaced with a new TID randomly selected from D^*.

Mutation involves randomly selecting a gene from a chromosome (a potential solution obtained after the crossover operation) (lines 20-21) and replacing it with another random solution from D^* that is not already present in the chromosome. Each chromosome in the offspring population is then evaluated (lines 22-24).

The evolution process unfolds iteratively. In each iteration, the current population and the offspring population are merged to form a pool of potential solutions with a size of $2N$ (line 29). Fast Non-Dominated Sorting and Crowding Distance strategies are applied to determine Pareto fronts (lines 30-32). Solutions are sorted into fronts, and the set of best solutions is saved in the archive (lines 33). This ensures that the archive is regularly updated with the best solutions.

The new population is formed by selecting the first N best potential solutions from the merged population (line 34). Crossover and mutation operations are applied (lines 35-36) to generate a new offspring population, and each chromosome in the offspring population is evaluated according to the three objective functions (lines 37-39).

The actions between lines 24 and 34 are repeated until the termination criterion is satisfied (line 28). The termination criterion can be a maximum number of iterations or a convergence criterion based on the Pareto front's convergence.

5.3.1.2 The evaluation function.

Prob_Evaluate is the proposed function used to evaluate each chromosome to determine the set of optimal items to delete from each uncertain candidate transaction and accordingly

to compute the three fitness values corresponding to the three side effects: Hiding failure, missing cost, and dissimilarity. The pseudo code of the *Prob_Evaluate* function is presented in Algorithm 5.

The proposed Algorithm 5 starts by extracting the set of sensitive uncertain transactions encoded in the chromosome c for evaluation (line 1). Then, for each extracted transaction t, the algorithm determines an optimal set of items to be removed to eliminate all the sensitive itemsets in this transaction. For the certain case, optimality consists in minimizing the number of removed items. In the probabilistic case considered in this chapter, removing items with low probability values is preferred. Hence, the optimal set of items to remove is the set covering all sensitive itemsets and having a minimal value for the total weight of its elements, which is the sum of probability values of all these elements. This is nothing but the Weighted Set Cover Problem (WSCP) which is the weighted version of the set cover problem used in the previous chapter. WSCP is also a well-known NP-Hard combinatorial optimization problem. We discuss this problem, its use for modeling our task and an efficient approximate way of solving it in the next section. In Algorithm 5, the function *WSCP* for solving the WSCP is called at line 7. This function receives two parameters: The set U of all sensitive expected frequent itemsets (line 5) and *Sitems* which contains all the items in U (line 6). The function *WSCP* returns as a result $Victimitems_t$ which contains the optimal subset of probabilistic items to remove from transaction t. Then, the modified database (after removing victim items) is computed (line 8) and the expected supports of current frequent itemsets are updated (lines 9-13). Finally the values of side effects are computed: The dissimilarity measure is computed by adding the sum of probabilities of the new removed probabilistic items (lines 14-18), the values of hiding failure (S_F_H) (resp. missing cost (N_S_L)) are computed by counting the number of expected frequent sensitive itemsets that remain frequent (resp. the number of non-sensitive expected frequent itemsets that become infrequent) after modifying the database (lines 21-28).

5.3.1.3 Weighted set cover problem.

The Weighted Set Cover problem (WSCP), as outlined by Yang [130], is a classic NP-Hard combinatorial optimization problem. In this problem, a universe of elements U and a set F of subsets of U are given, each associated with a weight. The goal of the WSCP is to identify a minimum weight collection (subset) C of subsets from F in such a way that the union of the elements in C covers all the elements in U. The decision version of the problem, which

Algorithm 5 Pseudo-code of *Prob_Evaluate* function

Require: c: chromosome; SI: sensitive expected frequent itemsets; D^*: projected uncertain transactions; D: original uncertain database; NS: set of non-sensitive expected frequent itemsets.

Ensure: S_F_H, N_S_L, Dis.

1: $Trans_c \leftarrow \{T_{id} \in D^* | id$ is an identifier of a transaction coded in chromosome $c\}$;
2: $D' \leftarrow D$;
3: $Dis \leftarrow 0$;
4: **for** (Each transaction t in $Trans_c$) **do**
5: $U \leftarrow \{s | s \subseteq t$ and $s \in SI\}$;
6: $Sitems \leftarrow \bigcup_{s \in U} \{(i,p) \mid (i,p) \in t \text{ and } i \in s\}$; {Put in $Sitems$ all probabilistic items appearing in sensitive itemsets present in transaction t}
7: $Victitems_t \leftarrow WSCP(U, Sitems)$; {Determine the set of items to be removed from the uncertain transaction by applying the WSCP function}
8: $D' \leftarrow Modif\_Database(Victitems_t, D')$; {modify the uncertain database D' by removing the probabilistic items of the set $victitems_t$ from transaction t.}
9: **for** (Each X in L) **do**
10: **if** $((X \subseteq t)$ and $(X \cap Victitems_t \neq \emptyset))$ **then**
11: $ExpSupport_{D'}(X) \leftarrow ExpSupport_{D'}(X) - (\prod_{x \in X} Pr(x \in t)/|D|)$;
12: **end if**
13: **end for**{Update the expected support value in D' of each expected frequent itemset}
14: $Dis \leftarrow 0$;
15: **for** (Each probabilistic item $(i,p) \in Victitems_t$) **do**
16: $Dis \leftarrow Dis + p$;
17: **end for**{Update the value of the dissimilarity between D and D'}
18: **end for**
19: $S\_F\_H \leftarrow 0$;
20: $N\_S\_L \leftarrow 0$;
21: **for** (Each X in L) **do**
22: **if** $(X \in SI$ and $ExpSupport_{D'}(X) \geq minesup)$ **then**
23: $S\_F\_H \leftarrow S\_F\_H + 1$; {One more sensitive expected frequent itemset in D which remains frequent in D'}
24: **end if**
25: **if** $(X \notin SI$ and $ExpSupport_{D'}(X) < minesup)$ **then**
26: $N\_S\_L \leftarrow N\_S\_L + 1$; {One more non-sensitive expected frequent itemset in D which becomes infrequent in D'}
27: **end if**
28: **end for**
29: Return S_F_H, N_S_L, Dis;

involves determining whether there exists a solution of weight at most a given threshold, is NP-complete. This indicates that finding exact solutions for large instances is computationally challenging. Fortunately, As for the simple version of SCP used in the previous chapter, there exists a straightforward greedy approximate algorithm that can yield a good approximate solution to the WSCP within polynomial time.

In the context of this chapter, given a transaction t containing some sensitive itemsets, we aim to detect a subset of probabilistic items that (1) covers all sensitive itemsets present in t, i.e., each sensitive itemset should contain at least one probabilistic item from the detected subset, and (2) has a minimum weight, where the weight here is the sum of the probability values of the detected probabilistic items. This problem is formulated as a WSCP as follows:

- The universe U is the set of sensitive itemsets appearing in the transaction
- The set F of weighted collections of U is determined as follows: Let (i, p) be a probabilistic item of t such that i appears in at least one sensitive itemset of U (this set of probabilistic items is denoted by $Sitems$). We construct a subset F_i of F which contains all sensitive itemsets of U containing i and we consider p, which is the probability that i belongs to t, as the weight of F_i.

The pseudo-code of the greedy approximate algorithm to solve the WSCP adapted to our case is presented in Algorithm 6. It receives the universe U (set of sensitive itemsets) as well as the set $Sitems$ of all items from elements of U and returns the set $Victimitems$ which contains the probabilistic items that are a discovered solution.

First, to construct the set F, the algorithm associates to each probabilistic item (i, p) in $Sitems$ a collection F_i as explained above (lines 1-3). Hence, each collection of F is indexed by a probabilistic item of $Sitems$ so that the presence of the collection F_i in the solution of the WSCP means that i belongs to the set of victim items to remove. The set of victim items is then calculated by applying the greedy principle by iteratively choosing the probabilistic item having a probability value that is as low as possible and which corresponds to the collection that tries to cover a maximum number of element of U not yet covered (lines 4-11). At the end of each iteration, the selected probabilistic item is removed from $Sitems$ (line 9) and the elements of the corresponding collection F_i are considered as already covered (line 10).

Algorithm 6 Pseudo-code of the greedy algorithm for the WSCP

Require: U : a finite set of sensitive itemsets; *Sitems*: the set of probabilistic items appearing in U.

Ensure: *Victimitems* : the set of victim probabilistic items to be removed.

```
for (Each probabilistic item (i, p) in Sitems) do
    F_i ← {s ∈ U | i ∈ s};
end for
Victimitems ← ∅;
W ← U; {copy universe U in W}
while (W is not empty) do
    (i, p) ← the probabilistic item from Sitems which minimizes p / |F_i ∩ W|; {The probabilistic item that maximizes the number of covered elements not yet covered in the remaining universe is selected, with priority given to an item if it has a lower probability}
    Victimitems ← Victimitems ∪ {(i, p)};
    Sitems ← Sitems − {(i, p)}; {remove probabilistic item (i, p) from the set Sitems}
    W ← W − F_i;
end while
Return Victimitems
```

5.3.2 An illustrative example

A detailed example is provided here to clarify the various steps involved in the proposed algorithm. This example encompasses the procedure for assessing solutions and showcases the operation of the *Prob_Evaluate* algorithm. Consider the uncertain database presented in Table 5.1, denoted as D, and the task of mining expected frequent itemsets in D with $minesup = 0.2$. Table 5.2 shows the set of expected frequent itemsets with their expected support values.

Table 5.1: An Example of an Uncertain database (D)

TID	Transactions
1	$\{a(0.4), b(0.6), d(0.9), f(0.1)\}$
2	$\{a(0.5), c(0.1), e(0.4), f(0.8)\}$
3	$\{b(0.2), c(0.3), d(0.6), e(0.1)\}$
4	$\{a(0.3), b(0.8), c(0.2), e(0.5), f(0.4)\}$
5	$\{a(0.1), b(0.5), d(0.5), e(0.3)\}$

Additionally, let us assume that the set of sensitive expected frequent itemsets is denoted as $SI = \{\{b\}, \{b,d\}\}$ (highlighted in Table 5.2), with the remaining expected frequent itemsets considered non-sensitive. The projected uncertain transactions D^*, which encompass at least one of the sensitive expected frequent itemsets, can be observed in Table 5.3.

Table 5.2: The Expected frequent itemsets found in Table 5.1 for $minesup = 0.2$

Itemsets	Expected support
$\{a\}$	0.26
$\{b\}$	**0.42**
$\{d\}$	0.4
$\{e\}$	0.26
$\{f\}$	0.26
$\{b,d\}$	**0.2**
$\{e,f\}$	0.4

Table 5.3: Projected uncertain database D^*

TID	Transactions
1	$\{a(0.4), b(0.6), d(0.9), f(0.1)\}$
3	$\{b(0.2), c(0.3), d(0.6), e(0.1)\}$
4	$\{a(0.3), b(0.8), c(0.2), e(0.5), f(0.4)\}$
5	$\{a(0.1), b(0.5), d(0.5), e(0.3)\}$

The *Prob_Evaluate* function is executed on each chromosome (potential solution) of the population D^* which is randomly generated and where each gene represents the identifier of a transaction from D^*. An example of a chromosome from the population may be the following:

0	4	0	5	1

This chromosome encodes the uncertain transactions 4, 5, and 1. Hence, from each of those transaction, the aim is to find the minimal set of probabilistic victim items to be deleted. After that, the algorithm computes the three objective functions based on the removed victim items with its probability value.

The *Prob_Evaluate* function is applied in two steps: The first step is to check each transaction t from D^* that is encoded in the chromosome. For each such transaction t, the set U is calculated of all items from sensitive itemsets that appear in t.

The second step for each uncertain transaction t identified in the first step is to apply the weighted set cover problem (WSCP) to find the victim items to remove with their probabil-

Table 5.4: Uncertain transactions in the solutions

TID	Transaction	SI	Exp. support	Victim	prob.
4	$\{a(0.3), b(0.8), c(0.2), e(0.5), f(0.4)\}$	$\{b\}$	0.42	b	0.8
5	$\{a(0.1), b(0.5), d(0.5), e(0.3)\}$	$\{\{b\}, \{d\}\}$	[0.420.2]	b	0.5
1	$\{a(0.4), b(0.6), d(0.9), f(0.1)\}$	$\{\{b\}, \{d\}\}$	[0.420.2]	b	0.6

ities. Let take the example of the transaction with TID 5. The universe is $U = \{s_1, s_2\}$ where $s_1 = \{b\}$ and $s_2 = \{b, d\}$. The union of the corresponding probabilistic items in U is $Sitem = \{(b, 0.5), (d, 0.5)\}$, where each item is associated with its probability value taken from the uncertain sensitive transaction 5. The returned set of victim items is $\{(b, 0.5)\}$ indicating the probabilistic items to remove from transaction 5. Table 5.4 presents the set of victim items with their corresponding probabilities for removal from each sensitive uncertain transaction.

The third step is to make an update of the modified uncertain database D' (see Table 5.5) by deleting items with their probability values from each sensitive uncertain transaction. This step also updates the expected support of each itemset that is frequent in D and the dissimilarity, which is computed as the sum of the probability value of each victim item deleted from the uncertain database. In our example, the dissimilarity value is equal to (0.8 + 0.5 + 0.6) = 1.9.

Table 5.5: Modified uncertain database D'

TID	Transactions
1	$\{a(0.4), d(0.9), f(0.1)\}$
2	$\{a(0.5), c(0.1), e(0.4), f(0.8)\}$
3	$\{b(0.2), c(0.3), d(0.6), e(0.1)\}$
4	$\{a(0.3), c(0.2), e(0.5), f(0.4)\}$
5	$\{a(0.1), d(0.5), e(0.3)\}$

Two additional objective function values need to be computed: (S_F_H) and (N_S_L).

S_F_H represents the count of sensitive expected frequent itemsets that can still be mined from the sanitized database D' even after the sanitization process has been applied. In the example, we see that both sensitive expected sensitive frequent itemsets are hidden, hence, $S\_F\_H = 0$.

N_S_L is the number of non-sensitive expected frequent itemsets that are lost after altering the uncertain database. We can see that there are four non-sensitive expected frequent itemsets that still appear in the modified database D'. $N\_S\_L = 0$. To sum up, the value of the three objective functions are $S\_F\_H = 0$, $N\_S\_L = 0$, and $Dissimilarity = 1.9$.

5.4 Experimental results

The proposed U-NSGAII4ID algorithm was implemented using Matlab R2017a and evaluated on a PC workstation featuring an Intel Core i5-4005U processor, 48 GB of RAM, and running the 64-bit Microsoft Windows 10 operating system.

In our experiments, we executed the U-NSGAII4ID algorithm for 10 generations, employing a population size of 10 with unlimited archive solutions. The crossover probability was set to 0.95, and the mutation probability was set to 0.05 for the variation part of the algorithm. Each experiment was run 10 times to ensure robust results.

For the experiments, we utilized seven uncertain databases sourced from SPMF [128], namely: mushroom, chess, pumsb, connect, accident, foodmart, and T10IKD10OK (generated using the IBM Quest database generator). These datasets were selected due to their popularity and widespread use in recent studies on Privacy Preserving Data Mining (PPDM) for validation purposes. The main features of the uncertain datasets used are summarized in Table 5.6

Table 5.6: Features of each uncertain dataset

Datasets	$\#\|D\|$	$\#\|I\|$	*AvgLen*	*MaxLen*	Type
mushroom	8416	119	23	23	Dense
chess	3196	74	37	37	Dense
pumsb	49046	2113	74	74	Dense
connect	67557	129	43	43	Dense
accidents	340183	468	33.8	33.8	Sparse
foodmart	21556	1569	4	11	Sparse
T10I4D100K	100000	870	10.1	29	Sparse

5.4.1 Runtime

In this experiment, we measured the runtime of the proposed U-NSGAII4ID algorithm for the seven uncertain datasets. Figure 5.1 illustrates the variation of runtime with respect to the percentage of sensitive itemsets among expected frequent itemsets. The existential probabilities of items in each transaction were randomly assigned.

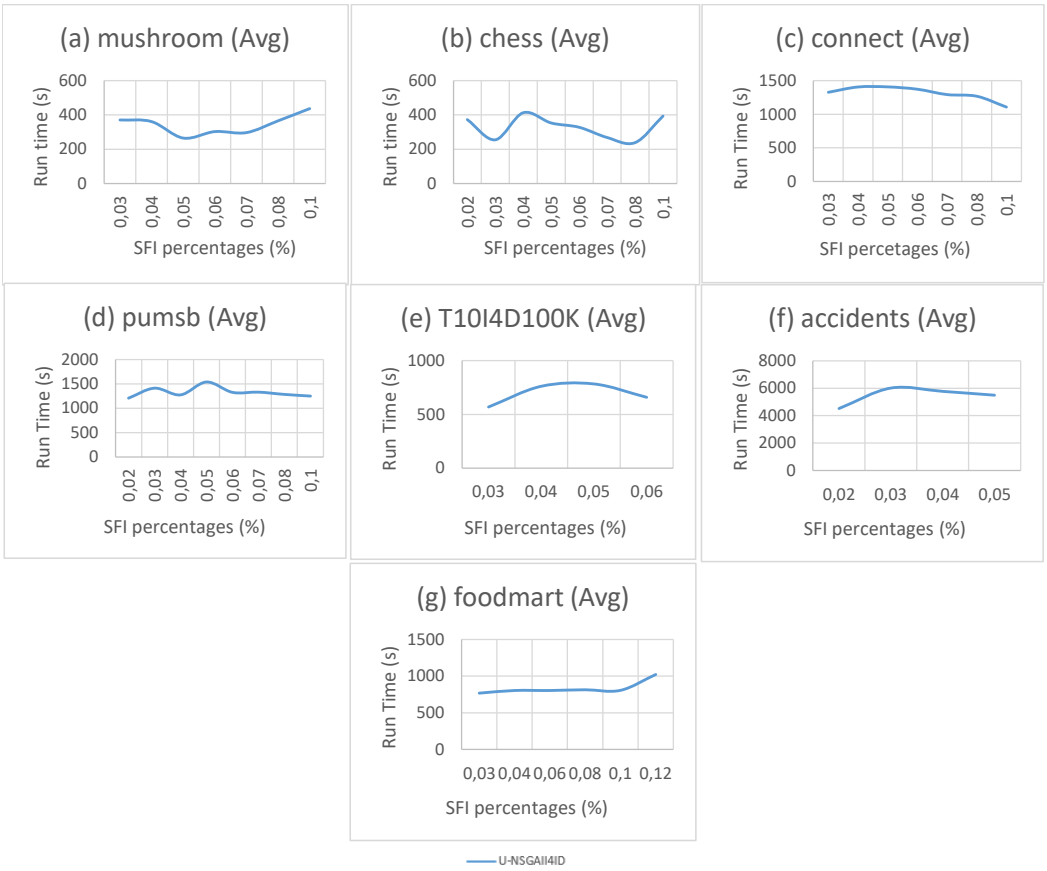

Figure 5.1: Runtime of U-NSGAII4ID for the seven uncertain databases

- Figure 5.1 (a) presents the average runtime for two *minesup* threshold values (0.1 and 0.2) for mushroom database. We can observe that the execution time of U-NSGAII4ID doesn't exceed 500 seconds as the percentage of sensitive expected frequent itemset is varied (from 3% to 10%).

- Figure 5.1 (b) shows the average execution time for the chess database with the two *minesup* threshold values (0.2 and 0.3). We notice that the average runtime is less than 400 s for different values of SI % (from 2% to 10%). The highest point is 400 s, when SI (%) is equal to 4% and 10%.

- For the connect database in Figure 5.1 (c), the average execution time for *minesup* threshold values (0.2 and0.3) is higher. It increases to 1400 s for SI equals to 3%, is stable until SI is 6% and starts to decrease until 1100 s for SI equals to 10%.

- Figure 5.1 (d) depicts the average runtime for *minesup* threshold values (0.2 and 0.3) for the pumsb database. The execution time reaches 1500s for SI =5% and then decreases and stabilizes in 1300 s when SI is from 6% to 10%.

- The average of the execution time for *minesup* values (0.01 and 0.02) for T10I4D100K (Figure 5.1 (e)) database increases until it reaches 800 s for 5% and decreases to 610 s for SI equals to 6%.

- In Figure 5.1 (f), we show the average runtime for *minesup* values (0.5 and 0.3) for the accidents database. We observe that the runtime increases from 4500 s to 6000 s for SI(%) varied from 2% to 3% and then decreases to 5500 s for SI (%) variation from 4% to 5%.

- Figure 5.1 (g) presents the average runtime for *minesup* values (0.002 and 0.0015) for the foodmart database. We can see that the execution time of U- NSGAII4ID Algorithm is stable in 800 s for SI (%) varying from 3% to 10 % and increases to 1000 s for SI% equals to 10%.

An overall remark that we can draw from the previous results is that there is no clear behavior in the variation of runtime with respect to the percentage of sensible itemsets. This means that runtime is not very sensible to this percentage.

5.4.2 Memory Cost

Figure 5.2 presents the experimental results obtained in terms of memory cost on the seven uncertain databases for the proposed U-NSGAII4ID algorithm.

Once again, we observe that there is no discernible pattern in the memory cost concerning the percentage of sensitive itemsets across various datasets. In general, the memory consumption remains within acceptable limits, with the chess dataset exhibiting the lowest consumption ranging between 1000 and 1500 MB, and the accidents dataset displaying the highest consumption between 2475 and 2532 MB.

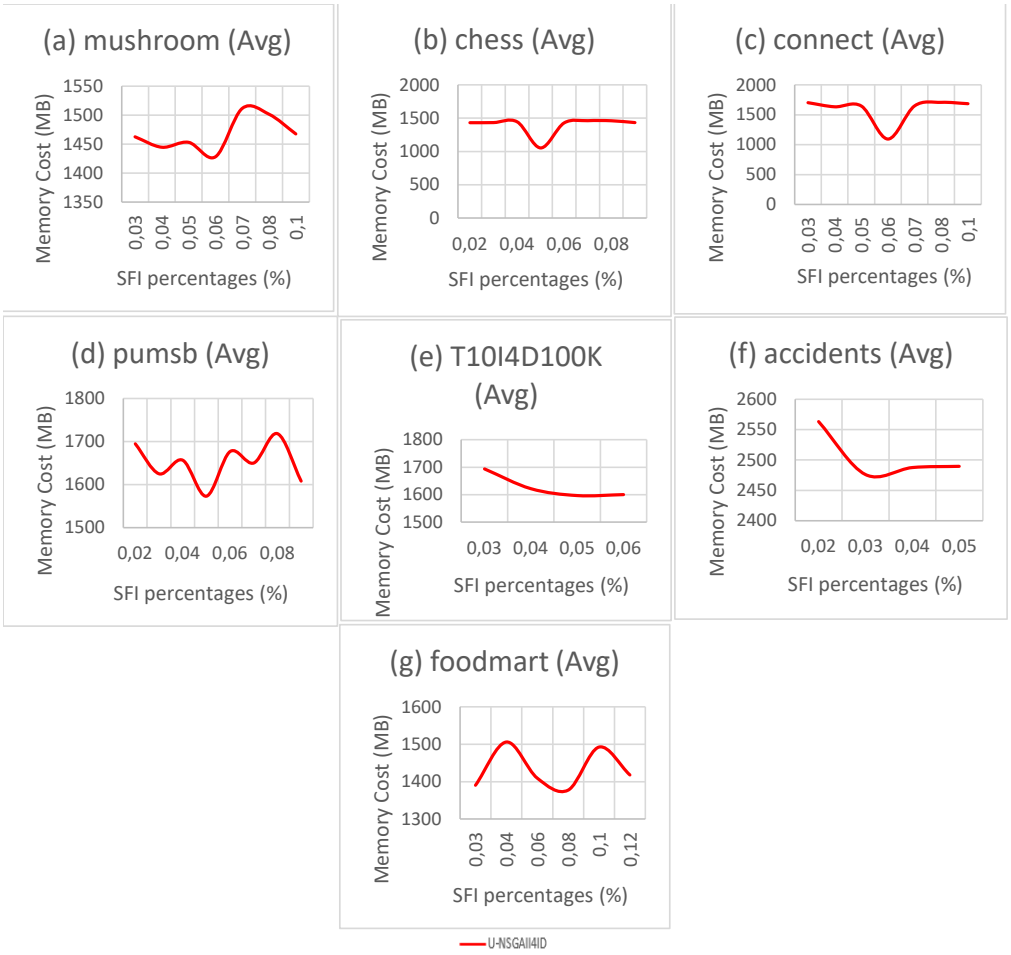

Figure 5.2: Memory Cost of U-NSGAII4ID for the seven uncertain databases

5.4.3 Side effects

Now let us evaluate in this subsection, the three side effects (objective functions) used in the proposed U-NSGAII4ID algorithm for the seven considered datasets.

5.4.3.1 Variation of side effects with respect to sensitive itemsets percentage.

In this experiment, we record the average outcomes of each side effect for different databases and various percentages of sensitive itemsets. The results are recorded based on two executions, each with distinct values of *minesup*. Specifically, we use 0.1 and 0.2 for the mushroom database, 0.2 and 0.3 for the chess, connect, and pumsb databases, 0.01 and 0.02 for the T10I4D100K database, and 0.002 and 0.0015 for the foodmart database.

Figure 5.3 presents the results obtained by the proposed U-NSGAII4ID algorithm in terms of the three side effects, namely: Hiding failure (S_F_H), missing cost (N_S_L) and Dissimi-

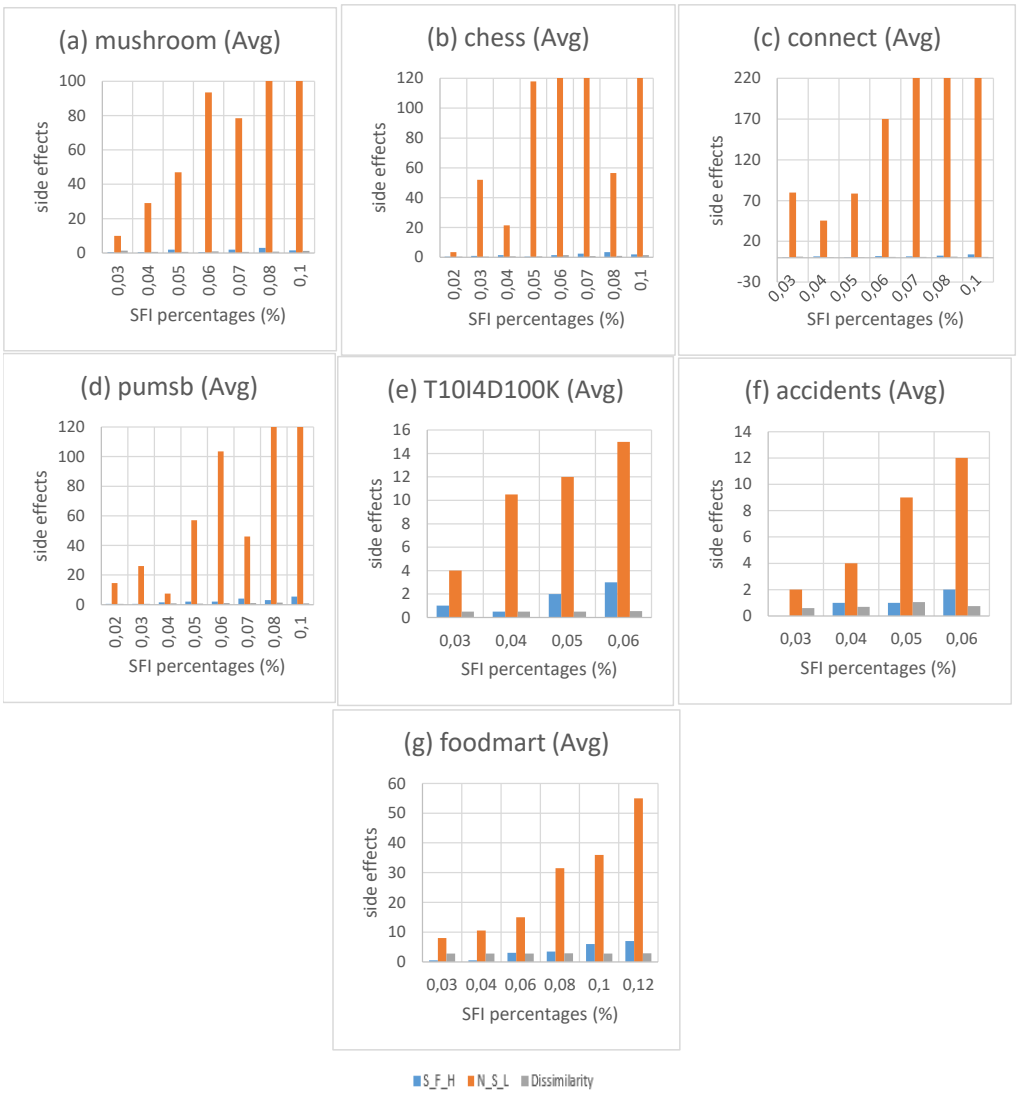

Figure 5.3: Side effects of U-NSGAII4ID w.r.t. different sensitive itemsets percentage for seven uncertain databases

larity. The following main points can be noticed from Figure 5.3:

- In general, the algorithm demonstrates excellent performance in terms of hiding failure (S_F_H) and dissimilarity. This indicates that the utilization of the Weighted Set Cover Problem (WSCP) is highly effective in concealing sensitive itemsets while introducing minimal modifications to the dataset. This outcome is highly promising and encouraging.

- Nevertheless, it is evident that the algorithm doesn't achieve the same level of success for the missing cost (N_S_L). This is due to the fact that the current evaluation function primarily prioritizes the minimization of S_F_H in the selection of a minimal set of victim items. There is a potential for enhancement by modifying the evaluation function to consider the minimization of N_S_L when choosing a minimal set of victim items.

- It is also noteworthy that the trade-off between hiding failure and dissimilarity on one hand, and missing cost on the other hand, is more pronounced in dense datasets compared to sparse ones.

5.4.3.2 Variation of side effects with respect to uncertainty degree

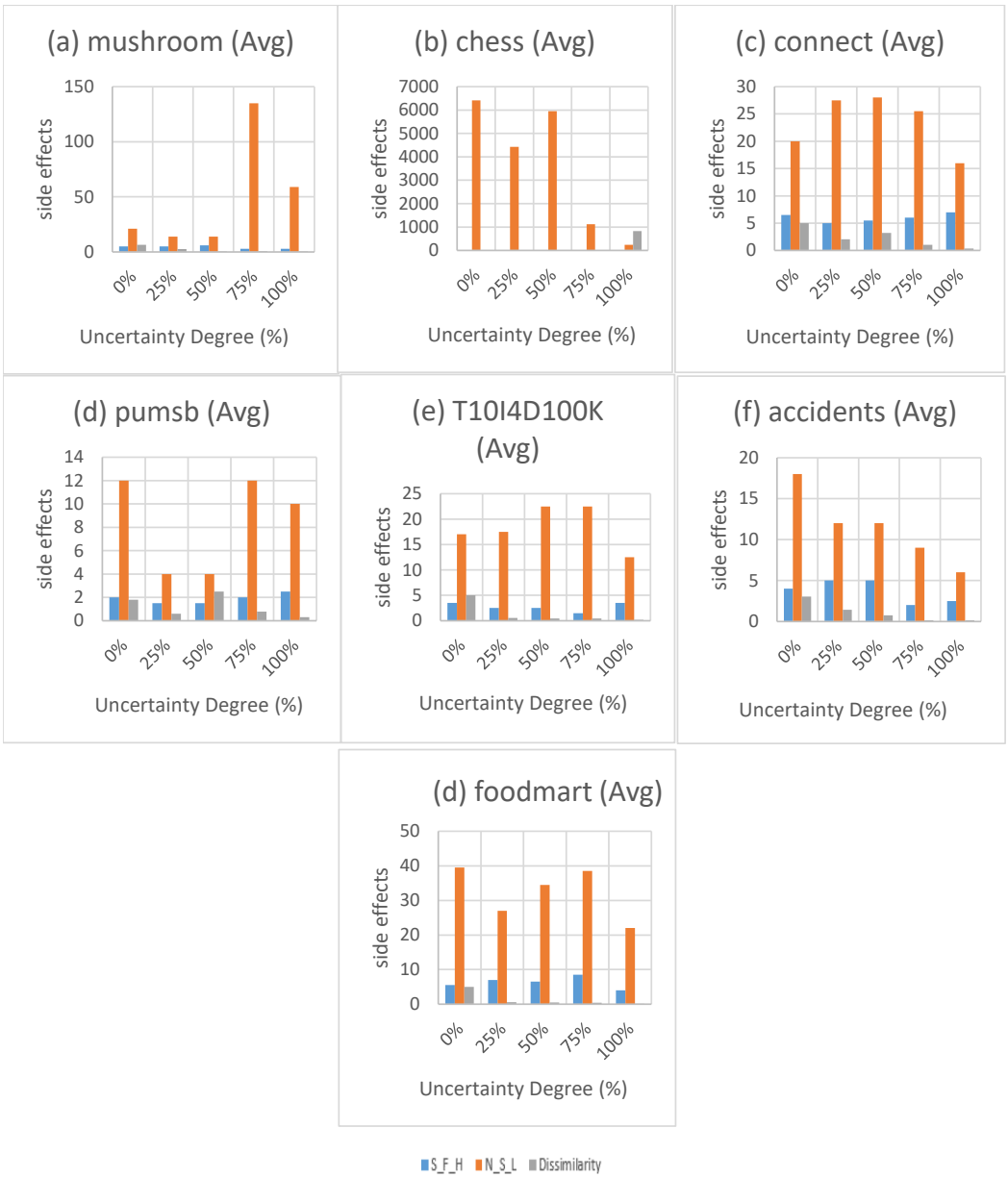

Figure 5.4: Side effects of U-NSGAII4ID w.r.t. uncertainty degree for seven uncertain databases

In this experiment, we analyze the average results of each side effect for various percentages of uncertainty degree in the dataset. The uncertainty degree represents the percentage of uncertain items, i.e., items with existential probabilities strictly less than 1. The experiment is

conducted with two different values of the minimum expected support threshold (*minesup*) for each database: 0.1 and 0.2 for the mushroom database, 0.2 and 0.3 for the chess, connect, and pumsb databases, 0.01 and 0.02 for the T10I4D100K database, and 0.002 and 0.0015 for the foodmart database.

Figure 5.4 presents the results obtained by the proposed U-NSGAII4ID algorithm in terms of the three side effects, namely: Hiding failure (S_F_H), missing cost (N_S_L) and Dissimilarity. Figure 5.4 shows that by varying the uncertainty degree of the dataset, we notice almost the same remarks as in the previous experiment, namely: very good results in terms of hiding failure and dissimilarity that are better than those obtained for the missing cost.

5.5 Conclusion

Preserving sensitive information within expansive databases has emerged as a critical concern, leading to the development of the Privacy Preserving Data Mining (PPDM) research area. While existing methods focus on safeguarding sensitive itemsets in traditional, certain databases, our work addresses the pervasive issue of uncertainty in real-world domains, such as sensor monitoring systems and medical analyses. In this chapter, we propose an innovative PPDM approach tailored for uncertain databases. Our method aims at hiding sets of sensitive itemsets by strategically removing uncertain items from selected transactions.

Incorporating the expected support-based approach to handle probabilistic information, our algorithm employs the U-Apriori algorithm, an extension of the Apriori algorithm designed for mining expected support-based frequent itemsets in probabilistic transactional databases. To our knowledge, this algorithm represents a pioneering effort in the realm of PPDM for uncertain databases.

The proposed algorithm formulates the problem as a multi-objective optimization task with the aim of minimizing three crucial side effects: Firstly, the hiding failure quantified as the number of sensitive expected frequent itemsets that persist after the sanitization process; secondly, the missing cost, signifying the number of non-sensitive expected frequent itemsets lost during database modification and finally, the third objective to minimize dissimilarity, computed as the sum of probability values associated with removed probabilistic items during sanitization.

Innovatively, our U-NSGAII4ID algorithm incorporates two optimization levels. At the primary optimization level, a multi-objective genetic algorithm inspired by NSGAII identifies an optimal subset of candidate victim transactions for uncertain item removal. The secondary optimization level addresses the NP-Hard weighted set cover problem, determining the optimal subset of probabilistic items to remove from each transaction. We employ a polynomial-time greedy approximation algorithm to efficiently solve this second-level optimization problem.

Experimentation across seven datasets demonstrates the algorithm's efficiency in terms of resource consumption (time and space). It notably excels in minimizing side effects during database sanitization, particularly hiding failure and dissimilarity. While results related to missing cost show promise, further improvements are conceivable.

Future work could enhance missing cost considerations by modifying the second optimization level based on the Weighted Set Cover Problem. Additionally, adapting the algorithm to alternative approaches that consider the real value of frequency probability for itemsets is an avenue for exploration. Finally, extending the approach to encompass various data types and patterns, such as sequential patterns, episodes, high-utility patterns, or periodic patterns, presents an exciting prospect for further research.

Chapter 6: Heuristic approaches for hiding sensitive frequent itemsets in uncertain databases

6.1 Introduction

As stated in the previous chapters, privacy-preserving data mining (PPDM) aims to protect sensitive information, often presented as itemsets or association rules. Various strategies have been developed to modify the original database, creating a sanitized version that conceals sensitive information while minimizing several side effects. Three main approaches have emerged for sanitizing a database, namely, exact, meta-heuristic, and heuristic approaches.

While the previous chapter proposes a new metaheuristic approach to solve PPDM problem, the present chapter focuses on an heuristic approach. More specifically, we aim at generalizing the work presented in Amiri's study (2007)'[131]. Indeed, [131] introduces three heuristic techniques for sanitizing certain databases: the Aggregate, Disaggregate, and Hybrid approaches. The aim of this chapter is to adapt these heuristics to the case of uncetain data.

The subsequent sections of this chapter are organized as follows. Section 6.2 is devoted to roughly introducing heuristic approaches designed for PPDM. The focal point of our exploration is Section 6.3, where we provide detailed insights into the three heuristic approaches proposed for solving PPDM problem in the case of uncertain databases. Section 6.4 presents the outcomes of our experiments, featuring a thorough analysis and discussion of the results. Section 6.5 summarizes the key findings of the chapter.

6.2 Heuristic approaches for PPDM

Heuristics are strategies developed to efficiently tackle problems that are either too computationally demanding for traditional methods or lacking exact solutions. The primary goal of heuristic-based approaches is to generate an approximate solution within a reasonable time frame, considered satisfactory for the specific problem at hand. In the case of Privacy-Preserving Data Mining (PPDM), [131] introduced three heuristic approaches to address the sanitization problem, with a specific focus on modifying transactions containing sensitive itemsets.

The first approach, referred to as the Aggregate approach, revolves around the elimination of transactions containing at least one sensitive frequent itemset from the database. This method ensures the hiding of all sensitive itemsets by reducing their support to levels below the predefined minimum support (minsup) threshold. The second approach, termed the Disaggregate approach, involves the modification of transactions by selectively removing some of their items instead of entirely deleting transactions. In this approach, each transaction undergoes individual modification, wherein specific items are chosen for deletion to lower the support of sensitive frequent itemsets below the minsup threshold. The third approach, known as the Hybrid approach, represents a fusion of the Aggregate and Disaggregate approaches, incorporating elements from both methodologies. In all cases, the choice of transactions or items to remove are guided by simple and intuitive principles.

6.3 Description of U-heuristic algorithms

[131] presents three heuristic approaches for hiding sensitive frequent itemsets in traditional (certain) databases. In the subsequent sections, we will adapt and customize these approaches to align with uncertain databases. It is paramount to highlight that our primary objective is to alter the input database in such a manner that maximizes the hiding of sensitive Expected Frequent (E-F) itemsets while minimizing the loss of non-sensitive E-F itemsets.

We use in this chapter the same definitions of the main concepts (uncertain database, uncertain item, expected support, sensitive itemsets as well as the three side effects) introduced in the previous chapters. Let us start by giving an example of an uncertain database that will be used as a running example throughout the chapter.

Example 1 . *Table 6.1 gives an example of an uncertain database. We suppose that the chosen value for minsup threshold is 0.2. Notice that the set of E-F itemsets extracted from this database is :* $F = \{\{a\},\{b\},\{d\},\{e\},\{f\},\{b,d\},\{e,f\}\}$.

Table 6.1: Example of uncertain database (D)

TID	Transaction
1	$\{a:0.4, b:0.6, d:0.9, f:0.1\}$
2	$\{a:0.5, c:0.1, e:0.4, f:0.8\}$
3	$\{b:0.2, c:0.3, d:0.6, e:0.1\}$
4	$\{a:0.3, b:0.8, c:0.2, e:0.5, f:0.4\}$
5	$\{a:0.1, b:0.5, d:0.5, e:0.3\}$

6.3.1 U-Aggregate approach

In this adaptation, our objective is to identify a set of transactions within the uncertain database for deletion. Each iteration involves selecting one transaction for removal, with the process continuing until the expected support for all sensitive expected frequent itemsets falls below the specified *minsup* threshold.

Our primary criterion for optimization is the number of sensitive itemsets supported by the chosen transaction, denoted as b_k. Additionally, we aim to minimize the concealment of non-sensitive E-F itemsets throughout the sanitization process. Therefore, our preference is to select transactions that support a minimal number of non-sensitive E-F itemsets, denoted as a_k.

Moreover, adhering to the principle of minimal change, we prioritize the deletion of items with low probabilities over those with high probabilities. Consequently, we calculate the cost associated with deleting a transaction as the sum of the probabilities of all its items ($\sum_{x \in k} p(x \in k)$), with our objective being to minimize this cost. The steps of this approach are as follows:

Step1: Initialization

1. The sanitized database $D_1 \leftarrow$ the original database D.

2. $D^* \leftarrow$ Set of all transactions containing at least one sensitive itemset.

3. For every frequent (sensitive or not) itemset $X \in S \cup N$, compute $ExpSup(X)$, the ex-

pected support of X (initially: $ExpSup(X) \geq minesup\ \forall X \in F$).

4. $S_1 \leftarrow S, N_1 \leftarrow N$.

Step2: While $(S_1 \neq \emptyset)$ **do**

1. **For** (every transaction k of D^*)

   - $b_k \leftarrow$ the number of sensitive E-F itemsets (from S_1) that belong to k.
   - $(a_k) \leftarrow$ the number of non-sensitive E-F itemsets (from N_1) that belong to k.
   - Compute f_k as follows:

     If $(a_k \neq 0)$ **Then** $f_k \leftarrow \frac{b_k}{a_k \times \sum_{x \in k} p(x \in k)}$ **Else** $f_k \leftarrow \frac{b_k}{\sum_{x \in k} p(x \in k)}$

2. Select the transaction k^* from D^* which verifies : $f_{k^*} = max\{f_k\ :\ k \in D^*\}$.

3. Reduce the expected support of every itemset $X \in S_1 \cup N_1$ as follows:

$$ExpSup(X) \leftarrow \frac{|D_1|}{|D_1|-1} \times ExpSup(X) - \frac{\prod_{x \in X} p(x \in k^*)}{|D_1|-1}$$

4. $D^* \leftarrow D^* \setminus \{k^*\}$ and $D_1 \leftarrow D_1 \setminus \{k^*\}$.

5. For every itemset $X \in S_1$ (resp. $X \in N_1$), if $ExpSup(X) < minesup$ then $S_1 \leftarrow S_1 \setminus \{X\}$ (resp. $N_1 \leftarrow N_1 \setminus \{X\}$).

Example 1 (Cont). *Let us show the application of the proposed U-Aggregate approach to the uncertain database of Table 6.1. Recall that the set F of E-F frequent itemsets obtained from this database is : $F = \{\{a\},\{b\},\{d\},\{e\},\{f\},\{b,d\},\{e,f\}\}$.*

Consider the sensitive E-F itemsets to be hidden: $\{b\}$, $\{e\}$ and $\{b,d\}$. The transactions 1, 2, 3, 4, 5, which contain at least one sensitive E-F itemset, form the projected database D^*. The application of the U-Aggregate approach removes transactions 4 and 5 from the uncertain database. In the resulting modified database, all sensitive E-F itemsets ($\{b\}$, $\{e\}$ and $\{b,d\}$) are effectively hidden, i.e., their expected support is now less than minesup. However, a drawback of this approach is the loss of non-sensitive E-F itemsets such as $\{d\}$, $\{f\}$ and $\{e,f\}$, as they become non-frequent. Nevertheless, the remaining non-sensitive E-F itemsets in the original database, namely ($\{a\}$ and $\{d\}$) , remain as E-F itemsets in the modified database.

6.3.2 U-Disaggregate approach

Unlike the U-Aggregate approach, the U-Disaggregate approach modifies an uncertain transactional database by selectively eliminating specific items from transactions to hide sensitive E-F itemsets. For each transaction in the projected database D^*, we pinpoint the specific item x^* whose removal is determined by the following criteria: it maximizes the reduction in the expected support of sensitive E-F frequent itemsets while minimizing the reduction in the expected support of non-sensitive E-F frequent itemsets. Furthermore, we give priority to removing items with low probabilities. Similar to the U-Aggregate approach, this iterative process stops when the set S' becomes empty, indicating no more sensitive E-F itemsets to hide.

Step1: Initialization

1. The sanitized database $D_1 \leftarrow$ the original database D.
2. $D^* \leftarrow$ Set of all transactions containing at least one sensitive itemset.
3. For every frequent (sensitive or not) itemset $X \in S \cup N$, compute $ExpSup(X)$, the expected support of X (initially: $ExpSup(X) \geq minesup \ \forall X \in F$).
4. $S_1 \leftarrow S, N_1 \leftarrow N$.

Step2: While $(S_1 \neq \emptyset)$ **do**

1. **For** (every transaction k of D^*)
   - **For** (every item x of k)
     - $(b_{k,x}) \leftarrow$ Number of sensitive E-F itemsets in S_1 that are included in k and containing x.
     - $(a_{k,x}) \leftarrow$ Number of non-sensitive E-F itemsets in N_1 that are included in k and containing x.
     - Compute $f_{k,x}$ as follows:

       If $(a_{k,x} \neq 0)$ **Then** $f_{k,x} \leftarrow \frac{b_{k,x}}{a_{k,x} \times p(x \in k)}$ **Else** $f_{k,x} \leftarrow \frac{b_{k,x}}{p(x \in k)}$

2. Select from k^*, the item x^* which verifies: $f_{k^*,x^*} = max\{f_{k,x} \ : \ k \in D^*, x \in k\}$.

3. For every itemset $X \in S_1 \cup N_1$ with $X \subseteq K^*$ and $x^* \in X$, update its expected support:

$$ExpSup(X) \leftarrow ExpSup(X) - \frac{\prod_{x \in X} p(x \in k^*)}{|D_1|}$$

4. Remove x^* from transaction k^* (in D^* and D_1).

5. Remove from S_1 (resp. N_1) every sensitive itemset whose expected support becomes less than *minesup*.

Example 1 (Cont). *The application of the U-Disaggregate algorithm to the database D presented in Table 6.1 results in the removal of certain items from transactions that originally contained at least one sensitive E-F itemset. Specifically, it eliminates item b from transaction 1, item e from transaction 2, items b and e from transaction 4, and item e from transaction 5. As a consequence of these modifications, U-Disaggregate effectively conceals all sensitive E-F itemsets, while preserving the status of the remaining non-sensitive E-F itemsets as E-F frequent in the modified database.*

6.3.3 U-Hybrid approach

The U-Hybrid approach combines the strategies of both U-Aggregate and U-Disaggregate. Initially, it identifies transactions in D^* earmarked for deletion if the U-Aggregate approach were to be exclusively implemented. However, instead of directly eliminating these transactions, the U-Hybrid approach opts for the U-Disaggregate method: It selects a set of items from the identified transactions to delete, aligning with the disaggregate principle. This principle is designed to curtail the expected support of sensitive itemsets, ensuring it falls below the specified *minesup* threshold, while concurrently minimizing the extent to which non-sensitive frequent itemsets transition to a non-frequent state.

Example 1 (Cont). *The U-Hybrid algorithm operates on uncertain databases by identifying transactions that contain at least one sensitive E-F itemset and selectively removing certain items from these transactions. This iterative process continues until all sensitive E-F itemsets are successfully concealed. For the uncertain database illustrated in Table 6.1, the algorithm's application results in the deletion of items b and e from transaction* 5*, items b and d from transaction* 1*, and items b and e from transaction* 4*. Following these modifications, all sensitive*

E-F itemsets are effectively hidden, while the non-sensitive E-F itemsets remain unaltered.

6.4 Experimental results

To evaluate the performance of the three proposed heuristic approaches in the context of uncertain databases, we conducted extensive experiments. First, different values of the *minesup* threshold are testes. After careful consideration, we selected the value 0.2 for all experiments, ensuring a reasonable number of Expected Frequent (E-F) itemsets. Additionally, we varied the percentage of sensitive E-F itemsets in the experiments.

The uncertain heuristic approaches have been implemented using Matlab R2017a, and the experiments were executed on a PC equipped with an Intel Core i3-4005U processor, 4 GB of RAM, and a 64-bit Microsoft Windows 10 operating system.

The efficiency of the three uncertain heuristic approaches is evaluated using three authentic datasets sourced from the SPMF library [132]: **Chess** (created using the UCI chess dataset), **Pumsb Census Data** (focused on population and housing statistics), and **Connect** (constructed based on the UCI Connect-4 dataset). The main features of the used datasets are summarized in Table 6.2. It is crucial to highlight that the probability values associated with items in each uncertain database were randomly generated during the experiments.[1]

Table 6.2: Features of the datasets

Dataset	$\#\|D\|$	$\#\|I\|$	*AvgLen*	*MaxLen*
chess	3196	75	37	37
pumsb	49046	2113	74	74
connect	67557	129	43	43

6.4.1 Run Time and Memory Cost

In this experimental analysis, we conducted a comparative assessment of the execution time and memory cost for our three heuristics across the three databases. Figure 6.1 presents the results, detailing the run time (in seconds) and memory cost (in Mega Bytes).

[1]The uncertain databases that we use in the comparison of our three uncertain heuristic approaches are available on Github at the address: https://github.com/minalef/uncertain-databases

In Figure 6.1-(A1-A3), it is apparent that the U-Aggregate approach consistently outperforms the U-Disaggregate approach, which, in turn, exhibits faster performance than the U-Hybrid approach.

Turning our attention to memory consumption, as depicted in Figure 6.1-(B1), the U-Hybrid approach demonstrates higher memory usage in the chess dataset compared to the other two approaches, which show comparable memory costs. Figure 6.1-(B2) indicates that U-Disaggregate and U-Hybrid have similar memory costs, both surpassing that of U-Aggregate. Finally, in Figure 6.1-(B3), all three approaches showcase competitive memory consumption.

In summary, the overall results suggest that the U-Aggregate approach stands out as the most efficient in terms of both run time and memory cost, followed by the U-Disaggregate approach, and lastly, the U-Hybrid approach. This observation aligns with the expected order of simplicity among the three considered approaches.

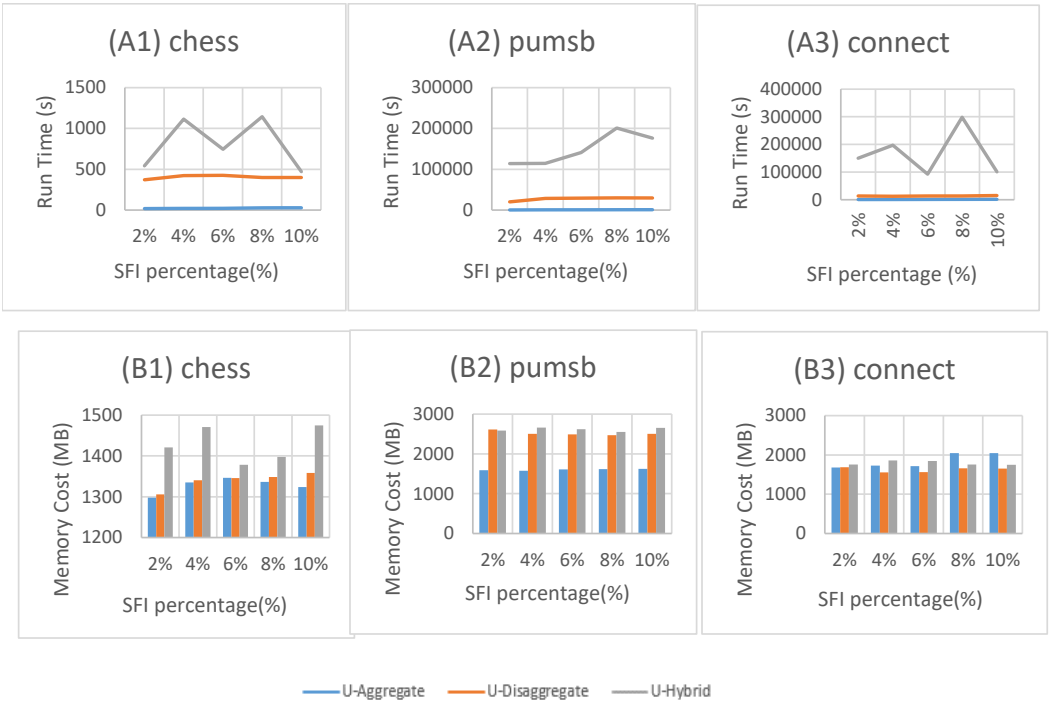

Figure 6.1: Run time and memory cost of three heuristics for three uncertain databases

6.4.2 Side effects

Now, let's explore the results concerning the two side effects: Missing cost and dissimilarity. It's crucial to emphasize that in all the proposed heuristics, the hiding failure side effect is entirely mitigated, given that all heuristics conclude their execution when all sensitive (E-F)

itemsets are successfully hidden. Figure 6.2 showcases the results related to the two scrutinized side effects for our three databases with a *minsup* threshold set at 0.2.

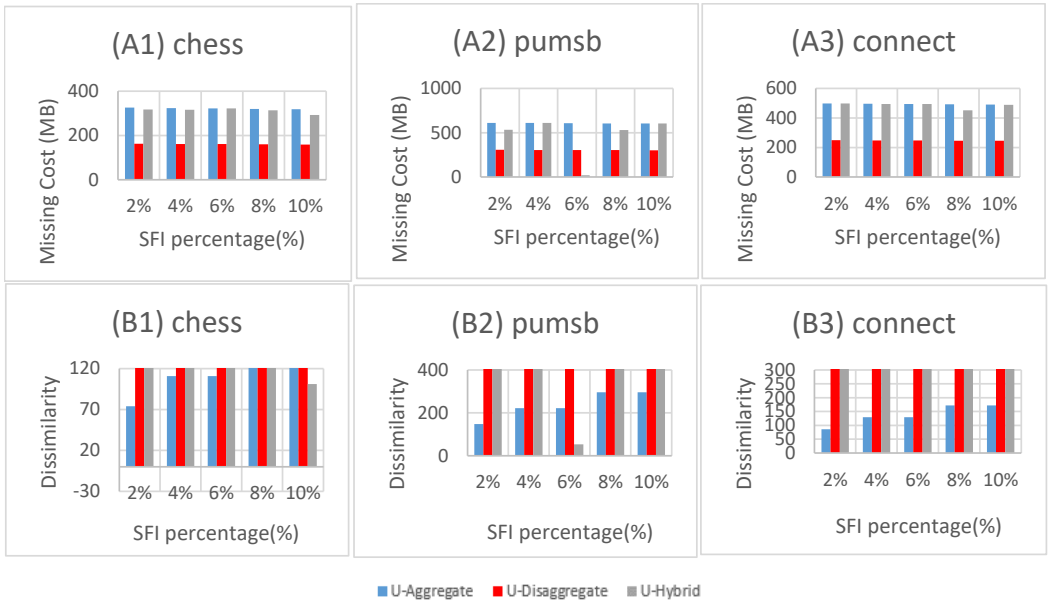

Figure 6.2: Missing Cost and dissimilarity of three heuristics for three uncertain databases

The analysis reveals that, in general, the H-Disaggregate approach outperforms the other two heuristics in minimizing missing cost. Conversely, the H-Aggregate approach surpasses the other heuristics concerning dissimilarity. This observation suggests that the hybridization method, combining both U-Aggregate and U-Disaggregate approaches, does not yield superior results in our specific case.

6.5 Conclusion

In this chapter, we present an adaptation of three heuristic algorithms for privacy-preserving pattern mining in the context of probabilistic transactional databases. The proposed heuristics, U-Aggregate, U-Disaggregate, and U-Hybrid, operate on the basis of the expected support-based interpretation of probabilities. Their primary objectives involve the strategic selection of transactions or items for deletion, aiming to achieve maximum hiding of sensitive expected frequent itemsets (E-S itemsets) while minimizing the impact on non-sensitive itemsets. This mirrors the objectives of the original heuristics designed for certain databases. Additionally, the integration of probabilistic information into the sanitization process is realized by prioritizing the deletion of items with low probabilities, consistent with the principle of minimal change.

Importantly, this work stands as a pioneering effort, as no prior research specifically addresses the privacy-preserving pattern mining problem for uncertain databases. Consequently, it establishes a solid foundation for future research endeavors in this innovative and unexplored research domain. Some of them are:

- In the process of choosing transactions or items for deletion, a more informed approach entails considering the cumulative probabilities associated with each sensitive (or non-sensitive) itemset's membership in the transaction. This approach goes beyond merely counting occurrences of sensitive or non-sensitive itemsets and incorporates information about the probability of each itemset's presence. By assessing the cumulative probabilities, the approach factors in the likelihood of each itemset contributing to the overall sensitivity or non-sensitivity of the transaction.

- Enhancing the evaluation of the dissimilarity measure could involve considering the existential probabilities of deleted items. In the current context, the assessment takes into account the probabilities associated with deleted items. For instance, deleting two items, a and b, with probabilities 0.1 and 0.2, respectively, is considered more costly than deleting one item, c, with a probability of 0.9. This consideration provides a more accurate reflection of the available probability values in the evaluation process.

- There is substantial work yet to be undertaken to extend other Privacy-Preserving Data Mining (PPDM) approaches, including exact methods and metaheuristic methods, to accommodate the uncertain data scenario.

Chapter 7: General Conclusion

7.1 Summary of contributions

Based on the extensive experiments conducted in Chapter 4 across four databases, encompassing both sparse and dense datasets, the NSGAII4ID algorithm demonstrated remarkable performance. It excelled in terms of execution time and the minimization of total modifications required to transform the original database into its sanitized counterpart. Furthermore, the results indicated that NSGAII4ID effectively mitigated the side effects of hiding failure and missing cost, achieving outstanding outcomes, particularly in the context of hiding failure across all databases.

Uncertain data, which are encountered in numerous real-world applications, result from limitations in our ability to perceive or comprehend reality, restrictions in observation equipment, or constraints in available resources for data collection, storage, transformation, or analysis.

In Chapter 5, we introduced a novel method known as U-NSGAII4ID, designed as an uncertain counterpart to the NSGAII4ID algorithm. Its primary objective is to tackle the problem of hiding sensitive expected frequent itemsets, transforming it into a multi-objective optimization challenge. Specifically, the U-NSGAII4ID algorithm operates by selectively removing items from designated transactions while simultaneously minimizing the aforementioned side effects. Our experimental assessments were conducted across seven uncertain databases (mushroom, chess, foodmart, T10I4D10IK, pumsb, connect, accident), and the results are presented with respect to runtime, memory cost, and the evaluation of three side effects. It's noteworthy that, given the absence of prior work in the domain of Privacy Preserving Data Mining (PPDM) addressing uncertainty, our project stands as a pioneering effort to confront this challenge.

Both NSGAII4ID and U-NSGAII4ID algorithm encompass two distinct optimization tasks that function at different levels. In the first level, a multi-objective genetic algorithm is employed to identify optimal subsets of candidate (un)certain sensitive transactions eligible for modification. The second level involves the selection of victim items within each (un)certain sensitive transaction for removal. To determine the optimal items for removal from selected transactions, we use the (Weighted) Set Cover Problem which is an NP-Hard problem. To solve it, a polynomial-time greedy algorithm is used to provide approximate solutions.

As a last contribution, we have addressed in Chapter 6 the heuristic-based approaches for solving the PPDM problem over uncertain data. For that purpose, we have adapted three heuristics proposed for certain databases to the situation where data is uncertain.

7.2 Future work and perspectives

While the methods discussed in this thesis have demonstrated satisfactory results, there are several intriguing and unresolved issues that warrant further investigation.

- We have noticed that both NSGAII4ID and U-NSGAII4ID obtain very good results on minimizing hiding failure and dissimilarity side effects but the results for missing cost are of the same quality. The reason behind this is that the selection method focuses more on hiding sensitive frequent itemsets than on preserving non-sensitive frequent itemsets. We plan to revisit the selection method to take into account the two aspects and hence improve the results in terms of missing cost.

- in the context of uncertain databases, the proposed sanitization process deletes selected probabilistic items to hide sensitive itemsets. We plan to consider finer operations which consist to not necessarily delete probabilistic items but only decrease their probability values. This allows one to further minimize the necessary update of the input database during the sanitization process.

- We have considered in this thesis the PPDM problem for frequent itemsets. As a future research, we would like to generalize our approaches to various other data and pattern types that have been extensively studied in the literature. This includes the consideration of other relevance criteria than frequency such as high utility or periodicity as well as

different kinds of data such as sequential databases, long sequences where the extracted patterns are called episodes, graphs, etc.

- As for the heuristic approach, we would like to consider the (W)SCP as a basis of new heuristics to solve the PPDM problem. We beleive that the resulting heuristics will perform meticulous selection of victim items.

References

[1] T. Cormen, C. Leisercon, R. Rivest, and C. Stein, "Introduction to algorithms, third 499 edition," 2009.

[2] C. C. Aggarwal1 and S. Y. Philip, *Privacy-preserving data mining: models and algorithms*. Springer Science & Business Media, 2008.

[3] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-art in privacy preserving data mining," *ACM Sigmod Record*, vol. 33, no. 1, pp. 50–57, 2004.

[4] S. Upadhyay, C. Sharma, P. Sharma, P. Bharadwaj, and K. Seeja, "Privacy preserving data mining with 3-d rotation transformation," *Journal of King Saud University - Computer and Information Sciences*, vol. 30, no. 4, pp. 524–530, 2018.

[5] L. Xu, C. Jiang, J. Wang, J. Yuan, and Y. Ren, "Information security in big data: privacy and data mining," *Ieee Access*, vol. 2, pp. 1149–1176, 2014.

[6] K. Liu, H. Kargupta, and J. Ryan, "Random projection-based multiplicative data perturbation for privacy preserving distributed data mining," *IEEE Transactions on knowledge and Data Engineering*, vol. 18, no. 1, pp. 92–106, 2005.

[7] K. Chen1 and L. Liu, "Privacy-preserving multiparty collaborative mining with geometric data perturbation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 12, pp. 1764–1776, 2009.

[8] J. Wang, Y. Luo, Y. Zhao, and J. Le, "A survey on privacy preserving data mining," in *2009 First International Workshop on Database Technology and Applications*. IEEE, 2009, pp. 111–114.

[9] W. Du, Y. S. Han, and S. Chen, "Privacy-preserving multivariate statistical analysis: Linear regression and classification," in *Proceedings of the 2004 SIAM international conference on data mining*. SIAM, 2004, pp. 222–233.

[10] K. Chen and L. Liu, "Privacy preserving data classification with rotation perturbation," in *Fifth IEEE International Conference on Data Mining (ICDM'05)*. IEEE, 2005, pp. 4–pp.

[11] C. C. Aggarwal, Y. Li, J. Wang, and J. Wang, "Frequent pattern mining with uncertain data," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 29–38.

[12] T. Bernecker, H.-P. Kriegel, M. Renz, F. Verhein, and A. Zuefle, "Probabilistic frequent itemset mining in uncertain databases," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 119–128.

[13] T. Calders, C. Garboni, and B. Goethals, "Approximation of frequentness probability of itemsets in uncertain data," in *2010 IEEE International Conference on Data Mining*. IEEE, 2010, pp. 749–754.

[14] ——, "Efficient pattern mining of uncertain data with sampling," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2010, pp. 480–487.

[15] C.-K. Chui and B. Kao, "A decremental approach for mining frequent itemsets from uncertain data," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2008, pp. 64–75.

[16] C.-K. Chui, B. Kao, and E. Hung, "Mining frequent itemsets from uncertain data," in *Pacific-Asia Conference on knowledge discovery and data mining*. Springer, 2007, pp. 47–58.

[17] C. K.-S. Leung, M. A. F. Mateo, and D. A. Brajczuk, "A tree-based approach for frequent pattern mining from uncertain data," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2008, pp. 653–661.

[18] L. Sun, R. Cheng, D. W. Cheung, and J. Cheng, "Mining uncertain data with probabilistic guarantees," in *Proceedings of the 16th ACM SIGKDD international conference on*

Knowledge discovery and data mining, 2010, pp. 273–282.

[19] Y. Tong, L. Chen, and B. Ding, "Discovering threshold-based frequent closed itemsets over probabilistic data," in *2012 IEEE 28th International Conference on Data Engineering*. IEEE, 2012, pp. 270–281.

[20] L. Wang, R. Cheng, S. D. Lee, and D. Cheung, "Accelerating probabilistic frequent itemset mining: a model-based approach," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, 2010, pp. 429–438.

[21] Q. Zhang, F. Li, and K. Yi, "Finding frequent items in probabilistic data," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 819–832.

[22] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, 1993, pp. 207–216.

[23] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, vol. 1215. Citeseer, 1994, pp. 487–499.

[24] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *ACM sigmod record*, vol. 29, no. 2, pp. 1–12, 2000.

[25] L. Sweeney, "k-anonymity: A model for protecting privacy," *International journal of uncertainty, fuzziness and knowledge-based systems*, vol. 10, no. 05, pp. 557–570, 2002.

[26] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 3–es, 2007.

[27] M. E. Nergiz, M. Atzori, and C. Clifton, "Hiding the presence of individuals from shared databases," in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, 2007, pp. 665–676.

[28] C. C. Aggarwal, M. A. Bhuiyan, and M. A. Hasan, "Frequent pattern mining algorithms: A survey," in *Frequent pattern mining*. Springer, 2014, pp. 19–64.

[29] D. J. Hand, "Principles of data mining," *Drug safety*, vol. 30, no. 7, pp. 621–622, 2007.

[30] P. Cabena, P. Hadjinian, R. Stadler, J. Verhees, and A. Zanasi, *Discovering data mining: from concept to implementation*. Prentice-Hall, Inc., 1998.

[31] B. Hssina, A. Merbouha, H. Ezzikouri, and M. Erritali, "A comparative study of decision tree id3 and c4. 5," *International Journal of Advanced Computer Science and Applications*, vol. 4, no. 2, pp. 13–19, 2014.

[32] W. Gan, J. C.-W. Lin, P. Fournier-Viger, H.-C. Chao, and P. S. Yu, "A survey of parallel sequential pattern mining," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 3, pp. 1–34, 2019.

[33] M. Patil and T. Patil, "Apriori algorithm against fp growth algorithm: A comparative study of data mining algorithms," *Available at SSRN 4113695*, 2022.

[34] S. Gupta and L. Mohanty, "A comparative study of various apriori and fp-growth tree-based incremental mining methods," in *Soft Computing for Problem Solving*. Springer, 2021, pp. 115–125.

[35] W. Gan, L. Chen, S. Wan, J. Chen, and C.-M. Chen, "Anomaly rule detection in sequence data," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[36] D. Maylawati, "The concept of frequent itemset mining for text," in *IOP Conference Series: Materials Science and Engineering*, vol. 434, no. 1. IOP Publishing, 2018, p. 012043.

[37] T. Uno, M. Kiyomi, H. Arimura *et al.*, "Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets," in *Fimi*, vol. 126, 2004.

[38] C. Romero and S. Ventura, "Educational data mining: a review of the state of the art," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 6, pp. 601–618, 2010.

[39] D. Sánchez, M. Vila, L. Cerda, and J.-M. Serrano, "Association rules applied to credit card fraud detection," *Expert systems with applications*, vol. 36, no. 2, pp. 3630–3640, 2009.

[40] J. M. Luna, P. Fournier-Viger, and S. Ventura, "Frequent itemset mining: A 25 years review," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 9, no. 6, p. e1329, 2019.

[41] Y. S. Koh and S. D. Ravana, "Unsupervised rare pattern mining: a survey," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 10, no. 4, pp. 1–29, 2016.

[42] A. Savasere, E. Omiecinski, and S. Navathe, "Mining for strong negative associations in a large database of customer transactions," in *Proceedings 14th International Conference on Data Engineering*. IEEE, 1998, pp. 494–502.

[43] H. Wang, X. Zhang, and G. Chen, "Mining a complete set of both positive and negative association rules from large databases," in *Advances in Knowledge Discovery and Data Mining: 12th Pacific-Asia Conference, PAKDD 2008 Osaka, Japan, May 20-23, 2008 Proceedings 12*. Springer, 2008, pp. 777–784.

[44] J. Han, M. Kamber, and D. Mining, "Concepts and techniques," *Morgan Kaufmann*, vol. 340, pp. 94 104–3205, 2006.

[45] M. J. Zaki, "Scalable algorithms for association mining," *IEEE transactions on knowledge and data engineering*, vol. 12, no. 3, pp. 372–390, 2000.

[46] C. Borgelt, "Frequent item set mining," *Wiley interdisciplinary reviews: data mining and knowledge discovery*, vol. 2, no. 6, pp. 437–456, 2012.

[47] P. Fournier-Viger, J. C.-W. Lin, R. U. Kiran, Y. S. Koh, and R. Thomas, "A survey of sequential pattern mining," *Data Science and Pattern Recognition*, vol. 1, no. 1, pp. 54–77, 2017.

[48] O. Abul, F. Bonchi, and F. Giannotti, "Hiding sequential and spatiotemporal patterns," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 12, pp. 1709–1723, 2010.

[49] N. R. Adam and J. C. Worthmann, "Security-control methods for statistical databases: a comparative study," *ACM Computing Surveys (CSUR)*, vol. 21, no. 4, pp. 515–556, 1989.

[50] D. Agrawal and C. C. Aggarwal, "On the design and quantification of privacy preserv-

ing data mining algorithms," in *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 2001, pp. 247–255.

[51] S. Agrawal and J. R. Haritsa, "A framework for high-accuracy privacy-preserving mining," in *21st International Conference on Data Engineering (ICDE'05)*. IEEE, 2005, pp. 193–204.

[52] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 439–450.

[53] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[54] N. R. Mabroukeh and C. I. Ezeife, "A taxonomy of sequential pattern mining algorithms," *ACM Computing Surveys (CSUR)*, vol. 43, no. 1, pp. 1–41, 2010.

[55] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proceedings of the eleventh international conference on data engineering*. IEEE, 1995, pp. 3–14.

[56] M. J. Zaki and K. Gouda, "Fast vertical mining using diffsets," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003, pp. 326–335.

[57] A. Achar, S. Laxman, and P. Sastry, "A unified view of the apriori-based algorithms for frequent episode discovery," *Knowledge and information systems*, vol. 31, no. 2, pp. 223–250, 2012.

[58] N. Tatti and B. Cule, "Mining closed episodes with simultaneous events," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 1172–1180.

[59] H. Mannila and H. Toivonen, "Discovering generalized episodes using minimal occurrences." in *KDD*, vol. 96, 1996, pp. 146–151.

[60] H. Mannila, H. Toivonen, and A. Inkeri Verkamo, "Discovery of frequent episodes in event sequences," *Data mining and knowledge discovery*, vol. 1, no. 3, pp. 259–289, 1997.

[61] S. Laxman, P. Sastry, and K. Unnikrishnan, "Discovering frequent episodes and learning

hidden markov models: A formal connection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 11, pp. 1505–1517, 2005.

[62] ——, "Discovering frequent generalized episodes when events persist for different durations," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 9, pp. 1188–1201, 2007.

[63] A. Ng and A. W.-c. Fu, "Mining frequent episodes for relating financial events and stock trends," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2003, pp. 27–39.

[64] K.-Y. Huang and C.-H. Chang, "Efficient mining of frequent episodes from complex sequences," *Information Systems*, vol. 33, no. 1, pp. 96–114, 2008.

[65] B. Bouqata, C. D. Carothers, B. K. Szymanski, and M. J. Zaki, "Vogue: a novel variable order-gap state machine for modeling sequences," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2006, pp. 42–54.

[66] N. Meger, C. Leschi, N. Lucas, and C. Rigotti, "Mining episode rules in stulong dataset," in *In Proc. of ECML/PKDD'04 Discovery Challenge-A Collaborative Effort in Knowledge Discovery. Prague: Univ. of Economics*. Citeseer, 2004.

[67] H. Zhu, P. Wang, X. He, Y. Li, W. Wang, and B. Shi, "Efficient episode mining with minimal and non-overlapping occurrences," in *2010 IEEE International Conference on Data Mining*. IEEE, 2010, pp. 1211–1216.

[68] X. Ao, P. Luo, C. Li, F. Zhuang, Q. He, and Z. Shi, "Discovering and learning sensational episodes of news events," in *Proceedings of the 23rd International Conference on World Wide Web*, 2014, pp. 217–218.

[69] X. Ao, P. Luo, C. Li, F. Zhuang, and Q. He, "Online frequent episode mining," in *2015 IEEE 31st International Conference on Data Engineering*. IEEE, 2015, pp. 891–902.

[70] X. Dai, M. L. Yiu, N. Mamoulis, Y. Tao, and M. Vaitis, "Probabilistic spatial queries on existentially uncertain data," in *International Symposium on Spatial and Temporal Databases*. Springer, 2005, pp. 400–417.

[71] H.-P. Kriegel and M. Pfeifle, "Density-based clustering of uncertain data," in *Proceed-*

ings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, 2005, pp. 672–677.

[72] J. Ren, S. D. Lee, X. Chen, B. Kao, R. Cheng, and D. Cheung, "Naive bayes classification of uncertain data," in *2009 Ninth IEEE international conference on data mining*. IEEE, 2009, pp. 944–949.

[73] S. Tsang, B. Kao, K. Y. Yip, W.-S. Ho, and S. D. Lee, "Decision trees for uncertain data," *IEEE transactions on knowledge and data engineering*, vol. 23, no. 1, pp. 64–78, 2009.

[74] C. C. Aggarwal and S. Y. Philip, "A survey of uncertain data algorithms and applications," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 5, pp. 609–623, 2008.

[75] S. Abiteboul, P. Kanellakis, and G. Grahne, "On the representation and querying of sets of possible worlds," *Theoretical computer science*, vol. 78, no. 1, pp. 159–187, 1991.

[76] E. Upfal and M. Mitzenmacher, "Probability and computing," 2005.

[77] M. Mitzenmacher and E. Upfal, *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press, 2017.

[78] Y. Tong, L. Chen, Y. Cheng, and P. S. Yu, "Mining frequent itemsets over uncertain databases," *arXiv preprint arXiv:1208.0292*, 2012.

[79] J. C.-W. Lin, J. M.-T. Wu, P. Fournier-Viger, Y. Djenouri, C.-H. Chen, and Y. Zhang, "A sanitization approach to secure shared data in an iot environment," *IEEE Access*, vol. 7, pp. 25 359–25 368, 2019.

[80] J. M.-T. Wu, G. Srivastava, A. Jolfaei, P. Fournier-Viger, and J. C.-W. Lin, "Hiding sensitive information in ehealth datasets," *Future Generation Computer Systems*, vol. 117, pp. 169–180, 2021.

[81] J. C.-W. Lin, L. Yang, P. Fournier-Viger, J. M.-T. Wu, T.-P. Hong, L. S.-L. Wang, and J. Zhan, "Mining high-utility itemsets based on particle swarm optimization," *Engineering Applications of Artificial Intelligence*, vol. 55, pp. 320–330, 2016.

[82] J. M.-T. Wu, C. W. Lin, P. Fournier-Viger, Y. Djenouri, C.-H. Chen, and Z. Li, "The density-based clustering method for privacy-preserving data mining," *American Institute*, 2019.

[83] S. Virupaksha and V. Dondeti, "Anonymized noise addition in subspaces for privacy preserved data mining in high dimensional continuous data," *Peer-to-Peer Networking and Applications*, vol. 14, no. 3, pp. 1608–1628, 2021.

[84] U. Ahmed, J. C.-W. Lin, G. Srivastava, and Y. Djenouri, "A deep q-learning sanitization approach for privacy preserving data mining," in *Adjunct Proceedings of the 2021 International Conference on Distributed Computing and Networking*, 2021, pp. 43–48.

[85] P. Kalia, D. Bansal, and S. Sofat, "A hybrid approach for preserving privacy for real estate data," *International Journal of Information and Computer Security*, vol. 15, no. 4, pp. 400–410, 2021.

[86] J. Wang, J. Zhang, W. Bao, X. Zhu, B. Cao, and P. S. Yu, "Not just privacy: Improving performance of private deep learning in mobile cloud," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 2407–2416.

[87] J. Wang, L. Wu, S. Zeadally, M. K. Khan, and D. He, "Privacy-preserving data aggregation against malicious data mining attack for iot-enabled smart grid," *ACM Transactions on Sensor Networks (TOSN)*, vol. 17, no. 3, pp. 1–25, 2021.

[88] X. Zhang, S. Jiang, Y. Liu, T. Jiang, and Y. Zhou, "Privacy-preserving scheme with account-mapping and noise-adding for energy trading based on consortium blockchain," *IEEE Transactions on Network and Service Management*, 2021.

[89] N. Nasiri and M. Keyvanpour, "Classification and evaluation of privacy preserving data mining methods," in *2020 11th International Conference on Information and Knowledge Technology (IKT)*. IEEE, 2020, pp. 17–22.

[90] K. Liu, C. Giannella, and H. Kargupta, "A survey of attack techniques on privacy-preserving data perturbation methods," in *Privacy-Preserving Data Mining*. Springer, 2008, pp. 359–381.

[91] E. Zorarpacı and S. A. Özel, "Privacy preserving classification over differentially private data," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 11, no. 3, p. e1399, 2021.

[92] E. Zorarpac1ı and S. A. Özel, "Privacy preserving rule-based classifier using modified artificial bee colony algorithm," *Expert Systems with Applications*, vol. 183, p. 115437, 2021.

[93] A. Aminifar, F. Rabbi, K. I. Pun, and Y. Lamo, "Privacy preserving distributed extremely randomized trees," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, 2021, pp. 1102–1105.

[94] N. Kousika and K. Premalatha, "An improved privacy-preserving data mining technique using singular value decomposition with three-dimensional rotation data perturbation," *The Journal of Supercomputing*, vol. 77, no. 9, pp. 10 003–10 011, 2021.

[95] M. D. Shastri and A. A. Pandit, "Remodeling: improved privacy preserving data mining (ppdm)," *International Journal of Information Technology*, vol. 13, no. 1, pp. 131–137, 2021.

[96] M. L. Merani, D. Croce, and I. Tinnirello, "Rings for privacy: an architecture for large scale privacy-preserving data mining," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 6, pp. 1340–1352, 2021.

[97] A. Pika, M. T. Wynn, S. Budiono, A. H. Ter Hofstede, W. M. van der Aalst, and H. A. Reijers, "Privacy-preserving process mining in healthcare," *International journal of environmental research and public health*, vol. 17, no. 5, p. 1612, 2020.

[98] J. C.-W. Lin, G. Srivastava, Y. Zhang, Y. Djenouri, and M. Aloqaily, "Privacy-preserving multiobjective sanitization model in 6g iot environments," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5340–5349, 2020.

[99] U. Ahmed, G. Srivastava, and J. C.-W. Lin, "A machine learning model for data sanitization," *Computer Networks*, vol. 189, p. 107914, 2021.

[100] J.-S. Lee and S.-P. Jun, "Privacy-preserving data mining for open government data from heterogeneous sources," *Government Information Quarterly*, vol. 38, no. 1, p. 101544,

2021.

[101] C.-W. Lin, T.-P. Hong, C.-C. Chang, and S.-L. Wang, "A greedy-based approach for hiding sensitive itemsets by transaction insertion." *J. Inf. Hiding Multim. Signal Process.*, vol. 4, no. 4, pp. 201–214, 2013.

[102] T.-P. Hong, C.-W. Lin, K.-T. Yang, and S.-L. Wang, "Using tf-idf to hide sensitive itemsets," *Applied Intelligence*, vol. 38, no. 4, pp. 502–510, 2013.

[103] J. M.-T. Wu, G. Srivastava, U. Yun, S. Tayeb, and J. C.-W. Lin, "An evolutionary computation-based privacy-preserving data mining model under a multithreshold constraint," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 3, p. e4209, 2021.

[104] J. M.-T. Wu, G. Srivastava, S. Tayeb, and J. C.-W. Lin, "A pso-based sanitization process with multi-thresholds model," in *International Conference on Pattern Recognition*. Springer, 2021, pp. 439–446.

[105] B. Abdollahzadeh and F. S. Gharehchopogh, "A multi-objective optimization algorithm for feature selection problems," *Engineering with Computers*, pp. 1–19, 2021.

[106] A. M. AbdelAziz, L. Alarabi, S. Basalamah, and A. Hendawi, "A multi-objective optimization method for hospital admission problem—a case study on covid-19 patients," *Algorithms*, vol. 14, no. 2, p. 38, 2021.

[107] A. A. Ewees, M. Abd Elaziz, and D. Oliva, "A new multi-objective optimization algorithm combined with opposition-based learning," *Expert Systems with Applications*, vol. 165, p. 113844, 2021.

[108] V. Coleto-Alcudia and M. A. Vega-Rodríguez, "A multi-objective optimization approach for the identification of cancer biomarkers from rna-seq data," *Expert Systems with Applications*, p. 116480, 2022.

[109] S. Goyal, P. Bedi, A. S. Rajawat, R. N. Shaw, and A. Ghosh, "Multi-objective fuzzy-swarm optimizer for data partitioning," in *Advanced Computing and Intelligent Technologies*. Springer, 2022, pp. 307–318.

[110] D. Lee and J.-H. Lee, "Hiding sensitive itemsets in transaction datasets using an evolu-

tionary algorithm," *Expert Systems with Applications*, vol. 39, no. 18, pp. 13 520–13 527, 2012.

[111] J. Li, H. Wang, J. Han, J. P. Zhang, and J. Lu, "Privacy preserving frequent itemset mining via an evolutionary algorithm," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 4, pp. 886–899, 2014.

[112] R. Li, Q. Zou, Y.-P. P. Chen, and B. Yang, "A novel genetic algorithm for privacy-preserving frequent itemset mining," *Journal of Ambient Intelligence and Humanized Computing*, vol. 6, no. 5, pp. 673–683, 2015.

[113] X. Lu, X. Jiang, and J. Shen, "Sensitive information hiding based on differential evolution algorithm," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 3, pp. 701–711, 2018.

[114] C.-W. Lin, B. Zhang, K.-T. Yang, and T.-P. Hong, "Efficiently hiding sensitive itemsets with transaction deletion based on genetic algorithms," *The Scientific World Journal*, vol. 2014, 2014.

[115] C.-W. Lin, T.-P. Hong, K.-T. Yang, and S.-L. Wang, "The ga-based algorithms for optimizing hiding sensitive itemsets through transaction deletion," *Applied Intelligence*, vol. 42, no. 2, pp. 210–230, 2015.

[116] J. C.-W. Lin, Q. Liu, P. Fournier-Viger, T.-P. Hong, M. Voznak, and J. Zhan, "A sanitization approach for hiding sensitive itemsets based on particle swarm optimization," *Engineering Applications of Artificial Intelligence*, vol. 53, pp. 1–18, 2016.

[117] J. M.-T. Wu, J. Zhan, and J. C.-W. Lin, "Ant colony system sanitization approach to hiding sensitive itemsets," *IEEE Access*, vol. 5, pp. 10 024–10 039, 2017.

[118] C. Liu, J. Zhang, and H. Xiong, "Multi-objective genetic algorithm for privacy preserving association rule mining," *Information Sciences*, vol. 232, pp. 146–164, 2013.

[119] X. Jiang, K. Huang, Y. Peng, and J. Wang, "Multi-objective privacy-preserving frequent itemset mining using non-dominated sorting genetic algorithm ii," *Applied Soft Computing*, vol. 40, pp. 131–145, 2016.

[120] J. Zhang, X. Liu, and B. Huang, "Multi-objective bat algorithm for privacy-preserving

frequent itemset mining," *Engineering Applications of Artificial Intelligence*, vol. 77, pp. 18–29, 2019.

[121] R. Wang, S. Zhang, and L. Cui, "Multi-objective genetic algorithm for privacy-preserving frequent itemset mining," *Journal of Intelligent & Fuzzy Systems*, vol. 36, no. 4, pp. 3789–3802, 2019.

[122] J. C.-W. Lin, Y. Zhang, B. Zhang, P. Fournier-Viger, and Y. Djenouri, "Hiding sensitive itemsets with multiple objective optimization," *Soft Computing*, vol. 23, no. 23, pp. 12 779–12 797, 2019.

[123] T.-Y. Wu, J. C.-W. Lin, Y. Zhang, and C.-H. Chen, "A grid-based swarm intelligence algorithm for privacy-preserving data mining," *Applied Sciences*, vol. 9, no. 4, p. 774, 2019.

[124] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data mining and knowledge discovery*, vol. 8, no. 1, pp. 53–87, 2004.

[125] S. R. Oliveira and O. R. Zaine, "Protecting sensitive knowledge by data sanitization," in *Third IEEE International conference on data mining*. IEEE, 2003, pp. 613–616.

[126] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[127] N. Bilal, P. Galinier, and F. Guibault, "A new formulation of the set covering problem for metaheuristic approaches," *International Scholarly Research Notices*, 2013.

[128] P. Fournier-Viger, J. Lin, A. Gomariz, T. Gueniche, A. Soltani, and Z. Deng, "The spmf open-source data mining library version 2," in *Proc. Joint European conference on machine learning and knowledge discovery in databases*, 2016, pp. 36–40.

[129] C. K.-S. Leung, "Uncertain frequent pattern mining," in *Frequent pattern mining*. Springer, 2014, pp. 339–367.

[130] J. Yang and J. Y.-T. Leung, "A generalization of the weighted set covering problem," *Naval Research Logistics (NRL)*, vol. 52, no. 2, pp. 142–149, 2005.

[131] A. Amiri, “Dare to share: Protecting sensitive knowledge with data sanitization,” *Decision Support Systems*, vol. 43, no. 1, pp. 181–191, 2007.

[132] P. Fournier-Viger, J. C.-W. Lin, A. Gomariz, T. Gueniche, A. Soltani, Z. Deng, and H. T. Lam, “The spmf open-source data mining library version 2,” in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part III 16*. Springer, 2016, pp. 36–40.