Towards Privacy-Preserving Publication of Continuous and Dynamic Data Spatial Indexing and Bucketization Approaches

Adeel Anjum

l'université nantes angers le mans
PÔLE DE RECHERCHE ET D'ENSEIGNEMENT SUPÉRIEUR

UNIVERSITÉ DE NANTES

Thèse de Doctorat

Adeel Anjum

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université de Nantes
sous le label de l'Université de Nantes Angers Le Mans

Discipline : Informatique et applications
Spécialité : Informatique
Laboratoire : Laboratoire d'informatique de Nantes-Atlantique (LINA)

Soutenue le 16 mai 2013

École doctorale : 503 (STIM)
Thèse n° : 000000000

Towards Privacy-Preserving Publication of Continuous and Dynamic Data

Spatial Indexing and Bucketization Approaches

JURY

Rapporteurs :	**M. PHILIPPE PUCHERAL**, Professeur, Université de Versailles St-Quentin en Yvelines **M. DAVID GROSS-AMBLARD**, Professeur, Université de Rennes 1
Examinateurs :	**M. BERND AMANN**, Professeur, Université Pierre et Marie Curie **M^{me} PATRICIA SERRANO-ALVARADO**, Maître de conférences, Université de Nantes
Directeur de thèse :	**M. MARC GELGON**, Professeur, Université de Nantes
Co-encadrant de thèse :	**M. GUILLAUME RASCHIA**, Maître de conférences, Université de Nantes

Contents

1

Introduction

***Summary**: The protection of data privacy is no more discretionary – its the law! The information surge has made the retrieval of public and private information of individuals a part of day-to-day life. Many critical services e.g., health care, typically gather this information for genuine needs; however, given the co-dependency of the Internet and information systems, sensitive data is under the radar of theft and corruption. Data privacy has received global attention for the past few decades. The rapid technology advancement has changed the way how privacy is protected and violated. Though it is hard to find "exact" definition of privacy, governments and institutions are facing a dilemma between information sharing and privacy protection. Thanks to dedicated efforts of research community, privacy preserving data publication appeared as a promising aspect to provide a first hand solution to this dilemma. In this Chapter, we motivate the need for privacy-aware systems that can be used effectively and efficiently for data publication tasks.*

Contents

A popular Government without popular information, or the means of acquiring it, is but a prologue to a farce or a tragedy; or, perhaps both. Knowledge will forever govern ignorance....
- James Madison

As stated by Jim Gray [53], we are entering the fourth age of science defined by a new paradigm where data play a central role in the production of science and innovation. To achieve that bright vision, scientific data must be unleashed from private repositories, and publicly released for all the research community. The Open Access movement, first focused on free access to scientific publications, turns now to Open Data initiative. In the same time, new business models have emerged to offer valuable services and take benefits from open data.

One of the major accomplishments of computer science is the flurry of information which seemed scarce few decades ago. The rapid advancement in hardware, especially enhanced processing speeds, the advent of giant storage abilities along with the reliable communication facilities and efficient information retrieval methods made such breakthrough possible. These advance capabilities have affected the basic means of human interaction including the way they work, communicate, and even their shopping preferences. On the other hand, these advancements have given rise to the explosion of data collection as almost every action of the individual is electronically recorded - every website she visits, every item she purchases etc. Such data collection not only benefits the individuals in terms of their everyday routine but also many important services like health-care have substantially improved due to the digitization of medical data.

Then, organizations are strongly encouraged to release their micro-data to support data analysis, to provide new business opportunities and to allow every kind of scientific study, to support data journalism and fact checking as well. For example, patients' medical records may be released by a clinic to support medical research and epidemiological studies. These organizations such as public and private institutions e.g., hospitals, collect the micro-data (e.g., medical reports, financial transactions, and residence records), and publish them regularly to serve the purposes of research and public benefits. For example, a *decision tree* based on the medical data of patients may help the practitioners to employ appropriate protocols for newly diagnosed diseases.

However, as a consequence, these data collections are responsible for tracking the public and private lives of concerned individuals, thus putting a big question mark on their privacy [44]. For instance, in October 2004, Choicepoint [1] released the financial information concerning 145,000 individuals to a group of criminals operating a scam. In August 2006, America OnLine (AOL) released 20 Million anonymous logs of search queries collected from 658 000 users to facilitate information retrieval research for academic purposes, after mapping each user to a randomly generated identifier. However,

1. ChoicePoint was a US based data aggregation company which used to provide intelligence services to the government and private institutions.

the privacy of the concerned individuals was easily breached [99] thereby revealing their private lives to millions.
Privacy is thus a global issue and being computer scientists, we own the power and obligation to design and develop tools to assure that the most basic right of human civilization i.e., privacy, is guarded from unwanted access while empowering the promulgation of precious data about humans for facilitating their day-to-day life. This thesis aims at identifying the techniques for publishing *useful* personal information with provable privacy guarantee and thus serves as a step towards accomplishing this obligation.

1.1 Problem Setting

Privacy preserving data publishing techniques focus on providing a sanitized view of a private dataset to the recipients, e.g. government institutions, research organizations, statisticians, etc. The private dataset contains the sensitive data about the individuals, e.g. hospital releases data about the patients for research or funding purposes. The algorithms for sanitizing such private datasets can be classified into *interactive* (those answer the queries posed on the dataset continuously) and *non-interactive* (those produce a sanitized dataset for the recipients). This thesis deals in the non-interactive data publishing scenario. *Non-interactive data publishing* can further be categorized into *local perturbation* and *centralized publishing* [89]. In Local perturbation approach, individuals themselves are responsible for perturbation and distribution of data to the recipients [106]. Centralized publishing usually assumes a *trusted server*, called a *publisher*, which is responsible for data collection of individuals, executing one or more privacy algorithm on the collected data for preserving the privacy of individuals and publishing for the end users. The algorithms for centralized publishing are known to provide balanced privacy/utility trade-off as compared to local perturbation algorithms due to the advantages offered by centralization of complete datasets [89]. The context of this thesis is the *centralized data publishing scenario*.
Figure 1.1 depicts the sanitization model in typical data publishing systems. In the *data collection phase*, the data publisher collects data from individuals (e.g., patients' data collected by hospitals). In sanitization phase, the data publisher employs a sanitization mechanism to protect the privacy of concerned individuals. In data publishing phase, the data publisher disseminate the data to external organizations or general public. Throughout the thesis, we assume a *trusted server* e.g., a hospital, which is responsible for data collection, sanitization and publication for the end user. This model is commonly referred to as *trusted model* [45]. In *un-trusted model*, the data publisher is not reliable and even can be one of the attackers. Thus, individuals themselves sanitize their data and provide them to the publisher. We invite the interested readers to [106] for statistical methods, [17,57] for anonymous communications and [116] for cryptographic solutions dealing in un-trusted model for data publishing.

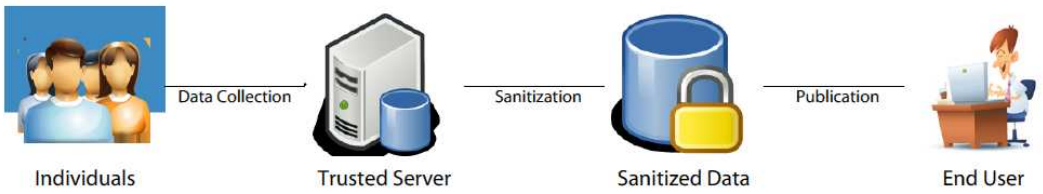


Figure 1.1: Data Sanitization Model

For more than half a century, many efficient database systems have been built which provide extremely proficient store and search facilities. Throughout the thesis, we assume relational data being stored in a relational database system [87]. A relational database consists of a set of *tables* or *relations*. An example of a relation is depicted in Table 1.1. Each relation contains a set of uniquely identifiable rows (also known as *tuples* or *records*) and each tuple corresponds to the medical record of an individual in the hospital. Columns in the relation correspond to the attributes of each patient. We denote by $t[A]$ the value of the A_{th} attribute value for a tuple t.
We introduce the problem setting with the help of following scenario. A trusted publisher, say *PIMS* hospital [2], collects the data about the patients. Typically, PIMS collect the micro-data as shown in Table 1.1 which has three kinds of attributes:

1. **Identifier(s)** (denoted by *ID*) are uniquely identifying attributes e.g., *Social Security Number, Name* etc.
2. **Quasi-Identifiers** (denoted by *QI*) is the set of attributes that can be used for linking with some externally available dataset e.g., Age, Zip Code and Gender.
3. **Sensitive Attribute** (denoted by *S*) contains sensitive information about individuals in the dataset that must be protected from adversary. In Table 1.1, *Disease* can be termed as a sensitive attribute.

This micro-data contains information related to several individuals. After every two months, PIMS releases this information to a pharmaceutical company ICI which conducts research and development of medicines for specific diseases and are interested in a study on how these diseases correlate with age and gender.

While relational database systems provide efficient solution for data management, the privacy of the individuals is of utmost importance. With the increasing anxiousness about privacy, organizations find themselves between a rock and a hard place. They affront a contention between privacy of their patients and the need to allow information

2. Pakistan Institute of Medical Sciences (PIMS) is a government hospital in Pakistan which provides health services to the needy people free of cost.

ID	Age	Zip Code	Gender	Disease
1	62	44120	F	Flu
2	51	44190	M	Flu
3	48	44100	M	HIV
4	59	44470	F	Flu
5	77	44420	M	Gastritis
6	66	44420	M	HIV

Table 1.1: Micro-data Table

processing for the benefit of everyone. While patients trust the way organizations handle their data, they might not be confident on how their data may be utilized once they are made public.

The intriguing question therefore is: *how the data publisher like PIMS hospital, can sanitize the micro-data for external organizations or even the general public while preventing an adversary from "linking" an individual to his/her sensitive information in the published data?* The aim of the thesis is to answer this question in various scenarios.

1.2 Motivation

Many public and private organizations like hospitals, the Census Bureau and, even search engine companies collect personal information from individuals and share it with the public with the intent of data analysis. The most commonly anticipated sanitization mechanism employed by the organizations is to simply discard the *identifier attributes* before release. However, this sanitization of data is insufficient to protect the privacy of the concerned individuals. Sweeney [96] in an initial study, estimated that - in United States, 87% of the population can be uniquely identified using a set of naive attributes like gender, birth date, and zip code. In fact, she used these three attributes to link Massachusetts voter registration records (comprising of name, gender, zip code, and birth date) to supposedly sanitized medical data from the GIC [3] insurance company (comprising of gender, zip code, birth date and diagnosis). Using this "linking attack", also coined *re-identification attack*, she was able to uniquely identify the medical records of William Weld, the governor of Massachusetts.
This real life example illustrates that the adversary may be able to "link" an individual uniquely or to a small number of data records in the sanitized release through quasi-identifiers. This disclosure became possible due to the use of extremely simple pseudonymization process of data sanitization. This pseudonymization process involves

3. Group Insurance Company provides health insurance to the Massachusetts state employees

replacing the direct identifying attributes in a record e.g., *Name, SSN etc*, by one or more artificial identifiers (pseudonym or pseudo-random number) while leaving the remaining attributes as-is to keep the data useful.
This area of research coined *non-interactive* [4] *Privacy-Preserving Data Publishing* (PPDP for short), studies how to thwart such kinds of linking attacks. The major goal here is to avoid linking an individual to a specific or small number of records while preserving the usefulness of the sanitized data.

1.2.1 Privacy and Utility

> *Perfect privacy can be achieved by publishing nothing at all, but this has no utility; perfect utility can be obtained by publishing the data exactly as received, but this offers no privacy.*
> *- Cynthia Dwork*

Any sanitization mechanism achieves a trade-off between privacy and utility: publishing the unaltered data in its entirety makes it extremely useful but with zero privacy guarantee while *not publishing* the data at all is perfect for privacy but it does not offer great opportunities for data analysis. The pressing question in this context is *how to design a perfect privacy sanitization mechanism which outputs extremely useful data?* This issue is however, extremely hard to address as these two ways of data publication have exact opposite requirements.
Privacy models and algorithms in PPDP facilitate the data publishers in choosing a point between these two extremes. Specifically, a privacy model formally defines the *extent of privacy* by drawing a baseline for the privacy guarantee under given circumstances. This helps the data publishers to chose a lower bound of privacy proposed by the given privacy model. The *privacy algorithms* physically transform the micro-data to a sanitized version by providing the privacy higher than the lower bound of the given privacy model and utility (as close as possible) to the *optimal* where such optimality is measured using prominent utility metrics. This transformation of micro-data to a sanitized release is called *data sanitization or anonymization*. Each sanitized dataset is then finally published and make accessible to the intended recipients.

1.2.2 Static and Sequential Data Publication

Data publication can take place in both static and dynamic settings. In static settings, it is presumed that data are static and once released, cannot be further modified. Thus, data are collected, sanitized, and then published only once. In these settings, the data

4. This thesis does not consider the interactive PPDP framework which aims at answering queries requested on a private dataset rather than sanitizing the data once for all (See Section 2.1)

privacy protection is guaranteed by algorithms designed for *static* privacy models. The static privacy models assume a simplistic scenario of one time publication. Furthermore these models do not focus on the correlation among multiple published versions of microdata. Each privacy model has its own requirements specially the kind of adversarial knowledge it can cater. Therefore the research on the privacy preservation for static datasets can be thought of as a history of progressively more refined models. Famous static models include k-anonymity [96], ℓ-diversity [75] and t-closeness [70] (See Section 2.3.1).

In more complex situations where data publisher needs to periodically republish microdata, static privacy models can only guarantee privacy upto one single release. Sequential data anonymization is naturally more complex than static publication scenario mainly due to the dynamic nature of data. It deals with publication of multiple releases each containing data from previous release(s) along with new records and/or modification in the records of previous releases. Modification in the previous records correspond to either update in any of the attribute values or deletion of a record from one release to the next one. Along with these modifications, sequential data publication is prone to several kinds of adversarial attacks that are not applicable for static data publication. This makes the static publication models inappropriate for this scenario since even if each release is individually anonymous, combining multiple releases begets the situation in which privacy can be compromised.

1.3 Thesis Contributions and Organization

The main objectives of this work are:

1. to identify advanced privacy threats in sequential data publication;
2. to highlight the complexity of dynamic data publication;
3. to key out the possible directions for improving the utility of published data in both static and dynamic settings along with providing the protocols for improved query accuracy for point and range queries;
4. to propose state of the art algorithms for static and dynamic settings that are scalable and achieve better performance than previously proposed algorithms;

In this dissertation, we initiate formal privacy definitions and propose efficient algorithms that provably guarantee privacy along with substantial increase in utility. Our main contributions are stated below:

- we present a state of the art of spatial access methods in the context of data sanitization. We evaluated several existing proposals that make use of spatial indexes for data sanitization and highlight inherent deficiencies in each of them;
- we propose yet another approximative generalization algorithm, coined *BangA*, that combines very nice features from Point Access Methods (PAM) and cluster-

ing. Hence, it achieves fast computation and scalability as a PAM, and very high quality thanks to its density-based clustering step. Moreover, BangA could incorporate background knowledge in the generalization process and the resulting public releases natively support orthogonal range queries;
- dynamic data republication poses serious threats to the privacy of individuals as it enables several attacks that are irrelevant w.r.t. static data publication. We propose a privacy model for dynamic data publishing named τ-safety that efficiently prevents from privacy breaches due to background knowledge that tracks individuals in a sequence of public releases;
- we propose a bucketization-based algorithm for sequential data anonymization, named *τ-safe m-invariant generalization* that follows τ-safety privacy model and provides better utility of final release along with improved query accuracy as compared to its predecessor i.e., m-invariance;

Below we overview the main contributions and provide thesis organization:
The first part of the thesis (Chapter 2) provides an insight into related work. Research in Privacy Preserving Data Publication (PPDP) can be categorized into *syntactic privacy definitions* and *semantic privacy definitions*. Syntactic privacy definitions have been adopted widely for the past few decades and an uncountable number of privacy models and algorithms have been proposed under the umbrella of these definitions. A lot of research is primarily dedicated to developing algorithms and notions for syntactic privacy that thwart the *re-identification attacks*. k-Anonymity is one of the first very popular syntactic technique for thwarting linking attacks. Thanks to its conceptual simplicity, k-anonymity has been widely implemented as a practicable definition of syntactic privacy, and owing to algorithmic advancement for k-anonymous versions of micro-data, k-anonymity has attained much anticipated popularity. The problem with syntactic privacy definitions is their dependence on the type of adversarial knowledge. Since it is near to impossible to estimate the amount of background knowledge an adversary can possess, these definitions have been criticized for their applicability in critical privacy applications. This opened way to *semantic privacy definitions*. Semantic privacy definitions do not make any assumptions about data i.e., it does not take into account about how data is collected or generated or what is the adversarial background knowledge but rather forces the sanitization algorithms (mechanisms) to satisfy a strong semantic property. Famous semantic privacy definitions include *differential privacy* and *zero-knowledge privacy*.
The second part of the thesis (Chapter 3) aim at developing a generalization based privacy algorithm using spatial indexes with the intent of improving utility of the sanitized release. The familiar area of spatial indexing has been shown to have a striking parallel with data sanitization [55]. Chapter 3 provides an in-depth review of spatial indexing techniques that can be used for sanitization tasks. Also, it proposes BangA generalization based algorithm that combines strong features of Point Access Methods (PAM) and

clustering to achieve scalable, efficient and highly useful public release.
Most existing works on PPDP focus on a single data release. In more complex situations, data are often released sequentially to serve various information purposes. Though there exist few works on sequential data publication (See Section 2.3.1.5), much effort is needed to cover wide range of adversarial attacks that are possible due to the complexity of handling such dynamic data. Xiao et al. [112] proposed an effective privacy model named m-*invariance* that can guarantee privacy when the dataset is encountered with insertion of records along with deletions. However, m-invariance does not cater the modification of record's attribute values between two releases. Among the few works in the literature that relate to the sequential data publication, none of them focuses on arbitrary updates, i.e. with any consistent insert/update/delete sequence, and especially in the presence of auxiliary knowledge that tracks updates of individuals. In Chapter 4, we first highlight the invalidation of existing algorithms and present an extension of the m-invariance generalization model coined τ-safety. Then we formally state the problem of privacy-preserving dataset publication of sequential releases in the presence of arbitrary updates and chainability-based background knowledge. We also propose an approximate algorithm, and we show that our approach to τ-safety, not only prevents from any privacy breach but also achieve a high utility of the anonymous releases.
We conclude this thesis with a summary and possible future perspectives in Chapter 5.

2

State of the art

Summary: *Research in privacy preserving data publication can be broadly categorized in two classes. Syntactic privacy definitions have been under the cursor of the research community for the past many years. A lot of research is primarily dedicated to developing algorithms and notions for syntactic privacy that thwart the re-identification attacks [39, 107]. Sweeney and Samarati proposed a well-known syntactic privacy definition coined k-anonymity [95, 96] for thwarting linking attacks using quasi-identifiers. Thanks to its conceptual simplicity, k-anonymity has been widely implemented as a practicable definition of syntactic privacy, and owing to algorithmic advancement for k-anonymous versions of micro-data [42], k-anonymity has attained much anticipated popularity. Even today, k-anonymity is under discussion for newly proposed areas like social networking and transactional logs. k-anonymity is the very first approach to achieve sanitization of data. Other more sophisticated approaches have emerged in recent years to address the limitations of k-anonymity. Among these approaches are ℓ-diversity [75], t-Closeness [70] and m-Unicity [112] (Section 2.3.1). Syntactic privacy definitions cover several scenarios for data sanitization including single static publication and sequential data publication. However, the problems with syntactic privacy definitions is that they can be achieved deterministically and they are dedicated to a certain type of adversarial knowledge. Each syntactic privacy definition is prone to attacks if it is exposed to other kinds of adversarial knowledge. Due to the volatile nature of these definitions, there has been a flurry of privacy models and definitions each trying to handle a new possible adversarial attack. This gave birth to semantic privacy definitions. Semantic privacy definitions do not take into account the adversarial background knowledge but rather forces the sanitization algorithms (mechanisms) to satisfy a strong semantic property by the way of random processes. Famous semantic privacy definitions include differential privacy [31] and zero-knowledge privacy [47] where the*

later focuses on privacy in social networks. Though semantic privacy definitions are theoretically immune to any kind of adversarial attacks, their applicability in real-life scenarios has come under criticism. In order to make the semantic definitions more practical, the research community has focused its attention towards combining the practicalness of syntactic privacy with the strength of semantic approaches [46] such that we may in the near future benefit from both research tracks. This Chapter provides a detail insight into both these types of definitions and also overviews several popular privacy models pertaining to each of them.

Contents

2.1 Introduction

This chapter provides the necessary background knowledge for understanding the contributions of the thesis. Several privacy definitions came into existence in the past few decades from traditional *syntactic privacy definitions*, to the most recent ones i.e., *semantic privacy definitions*. First, we overview the vast field of privacy-preserving data publication for relational data. Since this thesis contributes towards algorithmic side of privacy-preserving data publication, we try to draw a fine line between *privacy models* and their respective algorithms (we use the terms *mechanism* or *algorithm* interchangeably throughout the thesis). Then we bring to light the syntactic and semantic privacy definitions in isolation. We also overview several publication scenarios including static and dynamic data publication as seen by syntactic privacy definitions. Since our main contribution concerns dynamic data publication through syntactic privacy definitions, we provide an insight into several syntactic privacy models dealing in this complex scenario. The privacy algorithms follow specific privacy models and utility of the sanitized release is one of the most important factors of their proposition. We overview few popular quality measures that can be used to judge the utility of sanitized data. Then we detail various privacy algorithms that follow syntactic privacy models. We also overview popular semantic privacy definitions specially the *differential privacy* and its extensions. Then we elaborate the differences between syntactic and semantic privacy settings. Finally, we accentuate the open problems relating to both privacy definitions and highlight the recent research that is bringing them closer to each other.

2.2 Privacy-preserving data publication (PPDP)

The work in privacy preserving data publication spans across three dimensions (Figure 2.1) namely *i)* data model *ii)* privacy models/threats and *iii)* sanitization mechanisms/algorithms or techniques for privacy preservation. PPDP models and algorithms are generally, strongly related to each other i.e., every new model is accompanied with a proof-of-concept algorithm. Nonetheless, it is important to analyze them separately for the ease of understanding. Though this thesis tends to contribute towards algorithmic advancements in PPDP, we also present a detailed study of popular privacy models in order to apprehend the problem globally. PPDP revolves around following important aspects [67] (though not very comprehensive), necessary for the understanding of the related work.

1. **Data ownership**: As mentioned earlier, this thesis deals in centralized data publishing scenario. Throughout the thesis, we assume that publishing organization itself is trustworthy, yet it must be cautious while publishing the data externally. This is because the concerned individuals might be hesitant in providing their private information in the first place. The organizations have to chalk out compre-

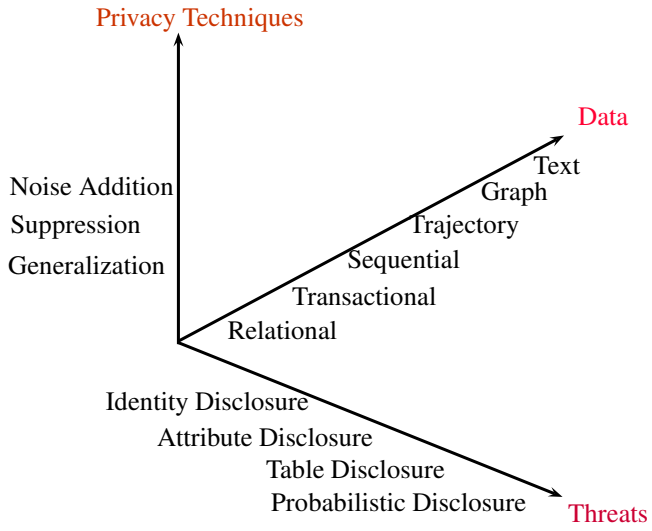


Figure 2.1: Overview of research directions in PPDP [58]

hensive publishing strategies to satisfy the concerned individuals against various

2. **Privacy Vs. Utility**: This is perhaps, the most important and intriguing aspect of PPDP. There must be a privacy policy such that the sanitized release is secure from any kind of intrusion given that it remains useful for the end user. In other words, there must be a balance between the notions of privacy and utility.
3. **Adversary's Knowledge**: The assumptions on the adversary's knowledge result in the outcome of several PPDP models and algorithms. In proposing an appropriate privacy model, it is important to study the resources available to the adversary not only in terms of externally available data but also other possible inferences.
4. **Data Model**: Another important aspect of PPDP is the kind of data to be dealt with. Much of the work done in PPDP relates to the static data publication with the philosophy of one record per individual and these records are assumed to be independent. However, data could be dynamic i.e., it is published sequentially with modifications. Other types of data include graph data, social network data and other non-relational data.

Note that *privacy-preserving data mining* and *statistical disclosure control* are closely related to PPDP. *Statistical disclosure control* aims at protecting statistical data. It allows the data to be published and analyzed by the public (mainly in the aggregated form), but protects private information of certain individuals or groups. On the contrary, *Privacy-preserving data publication* originates from the computer science society and it notably provides deep insights into adversarial models. *Privacy-preserving data mining* [25] focuses on applying some data mining tasks on a set of private databases owned by

different parties as well as focusing on privacy-preserving outputs of usual data mining tasks. In contrast, *privacy-preserving data publishing* distant itself from actual data mining task and concentrates on how to publish the data so that the anonymized data remains useful for data mining and querying. In what follows, we aim at privacy-preserving data publishing, referring neither to *statistical disclosure control* nor to *privacy-preserving data mining* anymore and invite the interested readers to consult the surveys [1] for *statistical disclosure control* and [5, 102] for privacy-preserving data mining. The privacy definitions for PPDP can be broadly classified into two categories [28]:

- *Syntactic* privacy aim at satisfying a *syntactic property* e.g., each individual in a sanitized release must be indistinguishable from certain number of other individuals in the sanitized release.
- *Semantic* privacy focus on privacy mechanisms to enforce a *semantic property* e.g., the analysis conducted on the sanitized release must be independent of the insertion or deletion of a tuple in a dataset

Below we overview popular privacy definitions and techniques relating to both categories.

2.3 Syntactic Privacy Definitions

Syntactic privacy assumes a relational table (referred to as a micro-data table) to be protected against an adversary who possesses certain amount of background knowledge for attacking the micro-data table to identify a target individual (commonly refer to as a *victim* of the adversary). Below we overview popular syntactic privacy models and techniques and refer the interested readers to [28] for in-depth analysis.

2.3.1 Prominent Syntactic Privacy Models for PPDP

The syntactic privacy models can further be classified into two categories based on the nature of the adversarial attack. In first category, an adversary is able to link a record, a sensitive information or a sanitized release to a record owner. We classify them as *identity disclosure*, *attribute disclosure* and *membership/table disclosure* respectively. In identity and attribute disclosure, the adversary knows that the record of an individual is in the sanitized release, and seeks to identify the individual's record and/or his/her sensitive information from the respective table. In membership or table disclosure, the attack consists of determining the presence or absence of an individual's record in the sanitized release. The second category encompasses probabilistic inferences and is regarded as *probabilistic disclosure*. It states that the sanitized release should provide with very little additional information apart from the background knowledge to the adversary. Table 2.1 summarizes attack models addressed by various privacy models.

Privacy Model	Attack Model			
	Identity disclosure	**Attribute disclosure**	**Table disclosure**	**Probabilistic disclosure**
k-Anonymity [95, 96]	✗			
MultiR k-Anonymity [83]	✗			
ℓ-diversity [75]	✗	✗		
Confidence Bounding [103]		✗		
(α, k)-Anonymity [109]	✗	✗		
(X, Y)-Privacy [104]	✗	✗		
(k, e)-Anonymity [117]		✗		
(ϵ,m)-Anonymity [69]		✗		
Personalized Privacy [111]		✗		
t-Closeness [70]		✗		✗
δ-Presence [82]			✗	
(c, t)-Isolation [18]	✗			✗
ϵ-Differential Privacy [31]			✗	✗
(d, γ)-Privacy [89]			✗	✗
Distributional Privacy [11]			✗	✗

Table 2.1: Privacy models [42]

2.3.1.1 The *k*-anonymity Model

To avoid identity disclosure, many organizations usually remove the uniquely identifying information like *Name*, *Social Security Number* etc. from the sanitized release. However, this sanitization of data might not be helpful in keeping the secrecy of given individuals. In the case brought to light by Sweeney and Samarati [96], it is discovered that the micro-data, even after the removal of identity information (e.g., social security number, name, and telephone number) is prone to *linking attack*. As a consequence, Sweeney was able to successfully identify the medical record of the governor of Massachusetts by linking his social information associated to the medical record (in the medical dataset) with an external data source (the voter list). A set of attributes that involves such linking attacks are termed as *quasi-identifier* of the dataset [96]. A dataset may contain several quasi-identifiers; we denote $\mathcal{QI}$ the set of quasi-identifiers hereafter.

Definition 2.1 *(Quasi-identifier (from [96])). Consider a set of attributes $\mathcal{A} = A_1, A_2, ..., A_n$ sampled from a general population. The set of attributes $QI_1, ..., QI_w \in \mathcal{A}$ is said to be a quasi-identifier if these attributes can be used via linking to uniquely identify an individual from the general population.*

In the case of the governor of Massachusetts, {*age, birthdate,zipcode*} is the quasi-identifier used for the linking attack. The voter list is termed as the *background knowledge* of the attacker. Such kind of linking attacks [22] can easily be thwarted by a simple pseudonymization scheme where quasi-identifiers are removed from the dataset. The highlighting question in this context is how much will be the information loss? In order to provide a balance between the privacy and utility, Sweeney et al [95, 96] proposed the *k*-anonymity model. The basic intuition of the *k*-anonymity model is to hide an individual in a crowd thereby blurring the link between the individuals and their respective records rather than deleting them altogether. *A table satisfies k-anonymity if every record in the table is indistinguishable from at least k - 1 other records in its public release*. This simple principle determines an equivalence relation on the data and is sufficient to prevent the disclosure of identity with a probability of $\frac{1}{k}$.

Definition 2.2 *(k-Anonymity from [75, 95, 96]). Let R be the dataset and $\mathcal{QI}$ be the set of all quasi-identifiers in it. R satisfies k-Anonymity, if for a record $t \in R$, there exist at least $k-1$ other records $t_1, t_2, \ldots, t_{k-1} \in R$ such that $t[QI] = t_1[QI] = \ldots = t_{k-1}[QI]$ for all $QI \in \mathcal{QI}$ where $t[QI]$ corresponds to the projection of t on the members of QI.*

Table 2.2 provides a toy example of a public release of 6 medical records from Table 1.1 following 3-anonymity, i.e., each public record is identical on quasi-identifiers (Age, Zip and Gender) with at least 2 other records. *A group of tuples with the same quasi-identifier value form an equivalence class*.

Definition 2.3 (X-Equivalence relation $\sim$) *Let R be a table with schema $R(X, Y)$. The X-equivalence relation $\sim_X \subseteq R \times R$ is defined as: $\forall t, u \in R$, $t \sim_X u \leftrightarrow t[X] = u[X]$.*

For a tuple $t \in R$, the X-equivalence class of t, denoted $[t]_{\sim_X}$, contains similar values on each component of X. In what follows, we refer to this QI-equivalence class as an equivalence class defined by a QI-equivalence relation. We will further explain the notion of equivalence class in Section 2.4.1.
Since the quasi-identifiers are susceptible to linking attacks, the table R is not released directly; it is first processed through a *sanitization mechanism* and then resulting table R^* is published. There exist various sanitization mechanisms in the literature e.g., *generalization, suppression, bucketization* etc. (See Section 2.4). Since this thesis employs generalization (generalization substitutes a specific value with a more general *less precise* value while preserving the data "truthfulness") as a sanitization mechanism, below we provide a definition of a generalization mechanism to achieve R^*.

Definition 2.4 (Generalization mechanism $\mathcal{A}$) *Given a micro-data table R, a generalization mechanism is a bijective function $\mathcal{A}$ defined as follows:*

Id	Age	Zip Code	Gender	Disease
1	[48-62]	441XX	*	Flu
2	[48-62]	441XX	*	Flu
3	[48-62]	441XX	*	HIV
4	[59-77]	444XX	*	Flu
5	[59-77]	444XX	*	Gastritis
6	[59-77]	444XX	*	HIV

Table 2.2: Example of a 3-Anonymous Public Release for Table 1.1

$$\begin{array}{lllll} \mathcal{A}: & R\langle ID, QI, S\rangle & \rightarrow & R\langle ID, QI, S\rangle \\ & \mathsf{I}(R) & \longmapsto & \mathcal{A}(\mathsf{J}(R)) = \mathsf{I}(R^*) = \\ & & & \{\langle t[ID], \nu, t[S]\rangle \mid t[QI] \preceq \nu \ \wedge \ t \in R\} \end{array} \tag{2.1}$$

where $J(R)$ is a generalized table of R, and ν is a generalized value of $t[QI]$ according to any pre-defined partial order over QI. For the sake of simplicity, we denote in the following by R the instance $\mathsf{I}(R)$ of R, and by R^* the instance $J(R)$ that is a generalized version of R.
Note that such a $\preceq$ partial order is basically a containment relationship. Also $\mathcal{A}(R)$ is not unique since there exist many different ways to generalize $t[QI]$ and the $\mathcal{A}(R)$ enumeration is properly combinatorics. Then, regular approaches try to optimize a utility-based objective function in the generalization mechanism. This is the underlying reason why the k-anonymization based generalization mechanisms have been proved to be NP-hard [59]
For example, the generalization in Table 2.2 partitions the records into two equivalence classes. Records 1, 2, 3 from Table 2.2 belongs to the same equivalence class and are indistinguishable one with each other. Pattern of the class is (Age=[48-62], Zip=441XX, Gender=*). Similarly, records 4, 5, 6 form the second equivalence class.

2.3.1.2 The ℓ-diversity Principle

While k-anonymity privacy model protects against identity disclosure, it does not provide sufficient protection against attribute disclosure. This is because the attributes that do not appear in the set of quasi-identifiers are not taken into account, even though these attributes contain highly sensitive information (e.g., patient diagnosis, salary, occupation etc). Machaanavajjhala et al. [75] highlight this inherent problem of k-anonymity model and propose a new privacy model, named ℓ-diversity, that aims at protecting the association between individuals and these *sensitive values*. Following the literature convention, from now on, we assume that the attributes of the dataset are made up of a single quasi-identifier (i.e., comprising of the union of the dataset's quasi-identifiers) and a single sensitive attribute.
Machaanavajjhala et al. [75] formulate the *bayes-optimal privacy* model by highlighting

the impact of sanitized release on the adversarial belief. The adversarial *prior belief* is modeled as the exact joint distribution f over sensitive values and $\mathcal{QI}$ of whole population. Given this distribution, the adversarial prior belief about the possibility of associating a given quasi-identifier q and a sensitive value s is the conditional probability to observe the association between q and s i.e.,

$$\text{Prior-belief}(q, s) = P_f(t[S] = s \mid t[QI] = q) \tag{2.2}$$

The adversarial *posterior belief* is calculated directly from the sanitized release R^* based on Bayesian probabilities and is given by:

$$\text{Posterior-belief}(q, s, R^*) = P_f(t[S] = s \mid t[QI] = q \wedge \ \exists t^* \in R^* \ , \ t \rightarrow^* t^*) \tag{2.3}$$

where t^* is generalized version of a tuple t.

Finally, the disclosure is defined as a significant difference between prior and posterior adversarial beliefs. This notion of defining the disclosure i.e., by comparing the prior and the posterior adversarial beliefs, is termed as *uninformative principle* [75]. The origin of uninformative principle is the Dalenius's early definition of statistical disclosure [26] and is the root of many influential privacy models [20, 71, 75, 89].

Definition 2.5 *(Uninformative principle [75]). The sanitized release should provide the adversary with very little additional information beyond the background knowledge. In other words, the prior and posterior beliefs should not differ much.*

Contrary to its strong properties, Machaanavajjhala et al. [75] show that bayes-optimal privacy model is impractical due to its strict restrictions and identify possible ways in which bayes-optimal privacy model can be thwarted. Consequently, ℓ-diversity privacy model is proposed as a practical alternative to bayes-optimal privacy model.

Definition 2.6 *(ℓ-Diversity Principle [75]). An equivalence class is ℓ-diverse if there are at least ℓ "well-represented" values for the sensitive attribute. A table is said to have ℓ-diversity if every equivalence class of the table has ℓ-diversity.*

The term "well-represented" in Definition 2.6 has several interpretations.

1. *Distinct ℓ-diversity* ensures that there are at least ℓ distinct sensitive values for the sensitive attribute in each equivalence class.
2. Distinct ℓ-diversity does not prevent *probabilistic inference attacks*. An equivalence class may have one sensitive value that is more frequent than the others thereby enabling the *homogeneity attacks* where an attacker can conclude that a particular individual is likely to possess that value. This motivated the need of more stronger notions of ℓ-diversity. A sanitized table satisfies *probabilistic ℓ-diversity* if the frequency of a sensitive value in each equivalence class is at most $\frac{1}{\ell}$. This ensures that an attacker cannot infer the sensitive value of an individual with probability greater than $\frac{1}{\ell}$.

3. *Entropy ℓ-diversity* states that, in each equivalence class, the entropy of each sensitive value must exceed a lower bound. The entropy of an equivalence class EC is given by:

$$Entropy(EC) = -\sum_{s \in Dom(s)} f(EC, s) \times \log f(EC, s) \tag{2.4}$$

Where $f(EC, s)$ is the fraction of records in EC having sensitive value s. A table is said to have entropy ℓ)-diversity if for every equivalence class EC, $Entropy(EC) \geq \log l$. Thus in order to have entropy ℓ)-diversity for every equivalence class, the entropy of the entire table must be at-least $\log l$ [75]. Sometimes this may be too restrictive, as the entropy of the entire table may be low if a few values are very common. This leads to the less conservative notion of recursive (c, ℓ)-diversity.

4. *Recursive (c, ℓ)-diversity* ensures that, in each equivalence class, the most frequent sensitive value does not appear too frequently, and the less frequent sensitive values do not appear too rarely. Let n be the number of sensitive values in an equivalence class EC, and $s_j, 1 \leq j \leq n$ be the frequency of j^{th} most frequent sensitive value appearing in EC. Then EC is said to have recursive (c, ℓ)-diversity if:

$$s_1 < c(s_m + s_{m+1} + \ldots + s_n)$$

A table follows the recursive (c, ℓ)-diversity if each equivalence classes in it is recursively (c, ℓ)-diverse. This ensures a smooth decrease of the privacy protection with respect to an attacker able to filter out an increasing number of sensitive values.

Other popular variants of the ℓ-diversity model include *p*-sensitive *k*-anonymity [101], (α, k)-anonymity [109] and (L, α)-diversity [94].

2.3.1.3 The Closeness Model

Though ℓ-diversity is a stronger privacy notion than *k*-anonymity, it also has limitations. Li et al. (t-closeness [70] and (*n,t*)-closeness [71]) highlight the inadequacy of ℓ-diversity principle for data sanitization when it encounters skewed data distribution. In general, ℓ-diversity is unable to guarantee privacy whenever the distribution of sensitive values within an equivalence class differs substantially from their overall distribution in the released table thereby allowing *skewness* and *similarity* attacks. For example, what would be the privacy of individuals who are in a 3-diverse equivalence class having 30% of "HIV" diseases, whereas only 1% of "HIV" appear in complete dataset. The authors of the closeness model also criticize the utility guarantees of ℓ-diversity by highlighting the fact that when the sensitive attribute is already not diverse in the complete dataset, the information loss in the sanitized dataset increases.

Li et al. [70] propose that the distribution of sensitive attribute in the complete dataset should be considered as *an auxiliary source of adversarial background knowledge*. *t*-closeness requires that the difference of sensitive attribute distribution in an equivalence class from the overall distribution of that sensitive attribute must not be more than a given threshold *t*. According to the *t*-closeness model, an adversary who knows the overall sensitive attribute distribution in the sanitized release gains only limited information about an equivalence class by learning the sensitive distribution in it. These considerations correspond to the uninformative principle (Definition 2.5) where the sensitive attribute distribution in the complete dataset makes adversarial prior belief, the distribution of sensitive attribute in each equivalence class makes the adversarial posterior belief and consequently the disclosure occurs when the difference between the two distributions exceeds the threshold *t*.

2.3.1.4 The Adversarial Background Knowledge

One of the most important problems in data publishing is to understand and reason about the adversarial background knowledge. In most cases, an adversary attempting to steal personal information of an individual from public data, has some instance-level information. For example, consider the generalized Table 2.2, and consider a curious neighbor who is able to isolate her friend *Pierre* to the second equivalence class. If she has seen *Pierre* recently, and knows that he does not have a *Flu*, then the probability of *Pierre* having HIV increases from $\frac{1}{3}$ to $\frac{1}{2}$.
k-Anonymity privacy model [95, 96] surmises that the adversary has access to some publicly-available external databases (e.g., voter list) through which she is able to nab the quasi-identifier values of the concerned individuals. The k-anonymity model also posit that the adversary possesses the information about the individual's existence in a given table. Bulk of the work following k-anonymity model presume this adversarial knowledge.
The ℓ-diversity and closeness models assume a specific form of adversarial background knowledge. The ℓ-diversity principle considers an adversarial knowledge to be the *negation statements* over sensitive values i.e., an adversary is able to discard some sensitive values from the equivalence classes whereas the closeness model restricts the adversarial background knowledge to the distribution of sensitive attribute in the sanitized release.
Martin et al. [76] initiated a formal study of the logical background knowledge in data publishing. Realistically, it is not possible for a data publisher to predict any instance-level knowledge an adversary can possess - keeping in mind the fact that there could be various such adversaries. Martin et al. [76] and Chen et al. [20, 21] propose the quantification of the adversarial knowledge such that the data-to-be-published is resilient to a certain amount of adversarial knowledge (in worst case and without the precise content of this knowledge). Specifically, Martin et al. propose a language - based on logical

sentences - for expressing the adversarial background knowledge, that consists of finite conjunctions of *basic implications* (by definition in [76], a basic implication has a form $(\wedge_{i=1,..,m} A_i) \rightarrow (\wedge_{j=1,..,n} B_j)$, where every A_i and B_j is an atom for associating a particular sensitive value to a particular individual). Subsequently, they propose a dynamic algorithm to verify safe anonymization i.e., given that the adversary is aware of maximum *k* basic implications, the probability of associating any individual to any sensitive value remains lower than a given threshold. This is termed as (c, k)-*safety* privacy model.

Chen et al. [20, 21] argue that quantifying background knowledge in terms of basic implications is not so intuitive, and propose that the interesting research idea is to consider only the real life background knowledge that can be handled efficiently. Subsequently, they propose three types of background knowledge (coined three-dimensional knowledge): *i)* knowledge about the target individual, *ii)* knowledge about other individuals in the sanitized release, *iii)* and knowledge about the relationships among individuals. Each type of background knowledge is quantified by a triplet (ℓ, k, m) which indicates that the adversary knows *i)* ℓ sensitive values that cannot be associated with the target individual *ii*) sensitive values associated with *k* other individuals *iii)* and *m* other individuals carrying the same sensitive value as that of a given individual. Then the sanitized release is such that, given the sensitive value σ and the expected amount of adversarial background knowledge about σ, the probability that an individual has σ remains lower than a given threshold. This is known as *3D-privacy model*.

Another popular adversarial knowledge model is *privacy-maxent* model [30] that focuses on expressing probabilistic background knowledge. Du et al. [30] criticized the expressive power of background knowledge in 3D-privacy model [20] since it fails to cater probabilistic background knowledge. Furthermore the authors of [30] defined the background knowledge in terms of probabilities e.g., P(*Testicular cancer* | *female*)$= 0$. Since main privacy problems arise due to the linkage of quasi-identifiers and sensitive attributes, the quantification of privacy - is therefore - to derive the probability of the linkage between any instance of sensitive attributes and quasi-identifiers with the probabilistic background knowledge. Du et al. thereby proposed privacy-maxent model [30] which formulates the deviation of the linkage probability as a non-linear programming problem.

The above mentioned approaches provide an efficient framework for defining and analyzing the adversarial background knowledge but they are unable to quantify the exact background knowledge a data publisher can possess. Li et al. [74] motivate the *kernel estimation techniques* for modeling probabilistic adversarial background knowledge. Specifically, they proposed *skyline (B,t)-privacy* model which is based on the adversary's prior and posterior beliefs. For a given skyline $(B_1, t_1), (B_2, t_2), \ldots, (B_n, t_n)$, a sanitized release satisfies *(B,t)*-privacy model if the maximum difference between the adversary's prior and posterior beliefs for all tuples in the data set is at most t_i. Wong

et al. [108] emphasized that the adversarial knowledge about the anonymization mechanism (or algorithm) can be used by the adversary to leak the sensitive information and they proposed a model coined m-confidentiality to avert such attacks.

2.3.1.5 Dynamic Data Publication

Since the emergence of k-*anonymization*, several privacy preserving paradigms have been proposed. The techniques mentioned above (commonly known as static data publication techniques or static techniques for short) ensure privacy protection up to a certain level i.e., they are focused on *single publication* of datasets. Realistically however, it is a common practice for the organizations to publish a dataset multiple times for different recipients statically or after modifications (either insertions, deletions or update) for providing up-to-date data. In dynamic data publication problem, the above mentioned techniques could provide protection pertaining to a single release. This need opens a new era in privacy preservation coined *Privacy preserving dynamic data publication*.

The problem of dynamic data publication can be broadly classified into following categories [42].

- **Multiple Release Publishing:** Multiple views of the same underlying micro-data are published once.
- **Sequential Release Publishing:** In this scenario, same micro-data is published multiple times with different recipients in mind. e.g., a hospital intends to release the person-specific data in Table 2.2 to either pharmaceutical company which needs the classification on *disease* attribute or a statistical organization which intends to apply statistical models on the attributes *(Age, Gender)*. In this publication scenario, different projections of a given micro-data table on different subsets of attributes are released
- **Continuous Data Publishing:** In this scenario, a data publisher has already released $R_1, R_2, \ldots, R_{p-1}$ and now wants to publish the next release R_p, where each R_i is a modified version of R_{i-1} in which data is inserted, updated or deleted.

Since our contribution in Chapter 4 deals in continuous data publication, we provide an insight of popular continuous data publication models and invite the interested readers for a detailed survey [42] on other dynamic publication scenarios.

Continuous data publication assumes that the data publisher has already published the releases $R_1, \ldots, R_{p-1}$ at times $(1, 2, \ldots, p-1)$ respectively and after the insertion of new records and/or modification (i.e., deletions and updates) in the previous records, he/she needs to publish R_p at time p. Also, an adversary is in possession of quasi-identifiers along with publication timestamps of her victim(s). We term a micro-data R_p a *fully dynamic dataset* if it may contain all three kinds of modifications i.e., insert/updates/deletes along its timeline (Timeline is modeled by a finite series of public releases or snapshots of R_p). Continuous data publication for fully dynamic datasets is an arduous task as compared to static publication by dint of two reasons, *i)* though

each sanitized release may be individually anonymous, the privacy of the concerned individuals could be at stake if an adversary can compare multiple releases and remove some candidate sensitive attribute values for a victim *ii)* sequential data publication brings about new adversarial attacks w.r.t single dataset publication scenario. Below we overview prominent continuous data publication models.

Sweeney in her seminal work on k-anonymization [96], identified possible inferences when new records are inserted in a dataset and proposed a couple of straightforward solutions. According to Sweeney, the records once generalized in any previous release, must remain the same or more generalized in the subsequent releases. The obvious problem with this solution is severe loss of utility. Furthermore, as explained by Bu et al. [15], this approach is vulnerable to *differencing attacks* in which the adversary may be able to filter out the records that have no correspondence with a target victim. Though such pruning of records may not breach the privacy of the target victim, it helps the adversary to narrow down to a smaller set of records which may contain the required record. The other solution proposed by Sweeney [96] is that once a dataset is sanitized and released, there should be no distinction between quasi-identifiers and sensitive attribute in subsequent releases i.e., all attributes in subsequent releases are treated as quasi-identifiers. This approach works well to defy the linking attacks but does not suffice for attribute disclosure as there might be the case when an equivalence class contains same sensitive attribute values. For instance, if each attribute in the sanitize release is considered as quasi-identifier then there is no restriction on how the sensitive values are distributed among the equivalence classes. Then, there might be a possibility that one sensitive value appears more frequently than the others in an equivalence class. Since this situation leads to homogeneity attack (Section 2.3.1.2), the privacy of any individual falling in that equivalence class is likely to be breached.

Byun et al. [15] presented the first study on the problem of continuous data publication. They identified several *inference channels* in a sequence of sanitized releases when each release is individually anonymous i.e., each release satisfies any static privacy guarantee (e.g., k-anonymity or ℓ-diversity). They present an interesting enhancement of ℓ-diversity privacy model for continuous data publication when new records are inserted. The authors of [15] propose that each continuous release must satisfy "distinct" ℓ*-diversity* (see Section 2.3.1.2). Since this instantiation of ℓ*-diversity* is prone to *homogeneity attacks*, therefore this instantiation used by Byun et al., cannot prevent attribute linkage attacks.

The authors of [15] tried to avert the inference channels in case of record insertions only. For the purpose of computational efficiency, the authors proposed to assign the incoming records directly into the previous sanitized release. The ℓ-diverse tables are internally maintained as *ungeneralized equivalence classes* and new records are thereby directly assigned to these equivalence classes subject to the following conditions *i)* R_p^* is ℓ-diverse *ii)* the quality of R_p^* is as high as possible *iii)* R_p^* is immune to possible

inferences. To improve the quality of the sanitized release, the algorithm [15] tends to specialize the data as much as possible by removing the inference channels such that if due to new records, there is a violation of given privacy requirements, the insertions are delayed for later sanitized releases. Consequently, this solution may fall into a situation in which no new records are released.
Byun et al. [14] further enhanced their previous proposition in [15] by incorporating other kinds of adversarial attacks so-called *cross version inferences*, when the new records are inserted. The authors improved their previous proposal so that it may be adopted with other generalization schemes (see Section 2.4.1 for details on generalization schemes). This improvement further took computational cost problem with the previous proposal into account. It suggested various heuristics for identifying possible inference channels and to significantly reduce the search space.

m-Invariance Byun et al. [14, 15] addressed the problem of continuous data publication in the insert-only scenario. Xiao et al. [112] identified that the continuous data publication is more complex than that. They showed that even if the continuous releases follow the principle proposed by Byun et al. [14, 15], they are vulnerable to other more sophisticated inferences. Specifically, they extended the continuous publication scenario where micro-data is modified with both insertions of new records and deletions of some previous ones. The authors of [112] discovered that the main reason behind the failure of static publication models in sequential data publication is that these models do not impose any constraint on sensitive values in the equivalence classes. Consequently, they proposed that the equivalence classes in all the public releases must contain the same set of sensitive values, the phenomenon they termed as *keeping persistent invariance* in equivalence classes. The authors of [112] further identified that if a sensitive value is deleted in the previous release, its *absence* in the current release can beget a situation where privacy breach is possible. They termed such an absence as *critical absence*. The phenomenon of critical absence occurs when a sensitive value is deleted in the previous release and the equivalence class containing that missing sensitive value is unable to possess same set of sensitive values in the subsequent release. In order to remove critical absence, the authors of [112] proposed to add fake records in place of missing sensitive values. These records are referred to as *counterfeit records*. Specifically, they proposed a *counterfeit generalization* technique coined *m-invariance*. A sequence of sanitized releases $R_1^*, R_2^*, \ldots, R_p^*$ is said to be m-invariant if

1. every sanitized release is m-unique (a sanitized release is m-unique if every equivalence class in it contains at least *m* records and all records have different sensitive values)
2. during the *lifespan* of a record (lifespan of a record is a range of timestamps in which that record exists), all equivalence classes containing that record have exactly the same set of sensitive attribute values.

An important aspect of m-invariance principle is that its space and time complexity are independent of the number of sanitized releases. This property is important in the republication scenarios where the number of sanitized tables increases monotonically i.e., the data publisher only needs last sanitized release for the sanitization of current release. We will discuss *m-invariance* scrupulously in Chapter 4.

BCF-Anonymity k-Anonymization is the pioneer model for static data publication. Since it is unable to cope with continuous publication scenario, Fung et al. [41] identified a situation in which the exact number of vulnerable records can be "cracked" by comparing a series of k-anonymous releases. Specifically, a record in a k-anonymous release is referred to as *cracked* if it cannot be picked up as a candidate record for the victim and if these cracked records are removed from the sanitized release, the resulting table could no longer follows k-anonymity. The authors of [41] identified different attack scenarios in which the records in two consecutive releases may be cracked. These attacks are termed as *Backward attacks*, *Cross attacks* and *Forward attacks* (*BCF attacks* for short). Consequently, Fung et al [41] proposed a privacy requirement, coined *BCF-anonymity*, to estimate the anonymity requirements after purging the cracked records and presented a generalization method to achieve *BCF-anonymity* without delayed records insertion or introducing counterfeit records. Since the generalization method proposed by Fung et al. does not cater the deletion scenario, it is vulnerable to the attacks due to critical absence phenomenon presented by Xiao et al. [112].

HD-Composition m-invariance assumes that quasi-identifiers and sensitive values of an individual remain the same over time. Bu et al. [12] relax this continuous data publishing scenario and assume that quasi-identifiers and sensitive values of an individual can change in subsequent releases. The authors of [12] assume that sensitive values can either be *permanent* (those cannot change over time e.g., in medical records, HIV disease has no cure available till date thus it cannot be changed in subsequent releases) or *transient* (those can change over time). The authors of [12] show that in the presence of permanent sensitive values, m-invariance principle is unable to guard the privacy of concerned individuals. They criticized m-invariance principle for its *record-based* continuous data publication approach rather than *individual-based* protection. The authors of [12] provided an efficient solution to cope with the problem of *individual protection* in the presence of permanent sensitive values efficiently. They proposed a generalization mechanism coined ***H**(older)**D**(ecoy)-composition* in the presence of *permanent sensitive values*. It carries two major roles namely *holder* and *decoy* where decoys are responsible for protecting permanent sensitive value holders. The main theme of the proposed principle is to bound the linkage probability of an individual and a permanent sensitive value by a given threshold (e.g., $\frac{1}{\ell}$). The two major partitioning principles presented by Bu et al. are: *role-based and cohort-based partitioning*. In *role-based* partitioning, for

each equivalence class, there are $\ell - 1$ decoys (those cannot be linked to permanent sensitive value) with each holder of a permanent sensitive value. Specifically, each holder is basically blended in crowd of $\ell - 1$ decoys. In *cohort-based* partitioning, there are ℓ *cohorts* from which, one belongs to the holders while the other $\ell - 1$ cohorts belong to the decoys. Also, the proposed method abjures the decoys from the same cohort to be placed in the same equivalence class. The main intent of cohorts based partitioning is to forge the information about true holders. The proposed technique is effective only in the case when the transition probability among transient values is uniform which is not often the case, the counterexample of this being the medical domain itself.

m-Distinct Li et al. [68] further assumed that while micro-data can be fully dynamic (i.e., inserts/updates/deletes), there is a certain correlation between the old values and the new ones. The authors of [68] proposed a *counterfeit generalization* model named *m-distinct*. The algorithm of *m-distinct* presents the concept of the *candidate update set* (Candidate update set for a sensitive value s - is the set of possible sensitive values to which s can be updated i.e., s can be updated to any value in its candidate update set with equal probability), taking advantage of the updates of sensitive attribute values that have the correlations between the old value and the new ones to solve the problem of continuous data publication. The rationale of *m-distinct* is that, it adopts m-uniqueness to maintain the anonymity of sensitive values in each separate publication; then in sanitizing subsequent releases, it carefully partitions the records so that the anonymity of sensitive values is still maintained. The authors termed this concept as *legal update instance* which guarantees that there is no possibility of inference via information exclusion. Though, the authors in [68] suggest that there is a certain correlation between new and old values in case of updates in sensitive attribute values, that may not be the case in many scenarios, for instance, if a person changes his/her residence then his/her new zip code is not known in advance.
Among the few works mentioned above that relate to continuous data publication, none of them focus on arbitrary updates, i.e., with any consistent insert/update/delete sequence, and especially in the presence of auxiliary knowledge that tracks updates of individuals. In Chapter 4, we present an extension of m-invariance, coined τ-safety. We show that even without any correlation between old and new values, an adversary, by exploiting the tracks of updates of individuals, can breach the privacy of the individuals. Table 2.3 provides an overview of prominent continuous data publication models with their limitations.

2.4 Prominent Syntactic Algorithms

The raw dataset must be sanitized in such a way that it satisfies certain privacy requirements. This sanitization is performed by applying a series of sanitization tech-

	inserts	deletes	updates	Support arbitrary updates with individuals tracking
Sweeney et al. [95]	✗			No
Byun et al. [14, 15]	✗			No
BCF-Anonymity [41]	✗			No
m-invariance [112]	✗	✗		No
He et al. [52]	✗	✗		No
HD-Composition [12]	✗	✗	✗	No
m-Distinct [68]	✗	✗	✗	No
τ-safety	✗	✗	✗	Yes

Table 2.3: Popular continuous data publication models

niques such as *generalization, suppression, bucketization etc*. PPDP algorithms implement certain privacy models by using these sanitization techniques.

2.4.1 Generalization-based Algorithms

Generalization:

Before starting a review of generalization-based algorithms, it is necessary to understand the major building block of these algorithms i.e., *generalization*. Generalization is a process of data sanitization by means of generalizing (coarsening) some attribute values in a micro-data table: the actual value - categorical (e.g., a profession) or numerical (e.g., age) - is substituted by a range (e.g., a group of professions, an age range). Generalization is also referred to as *recoding* in the statistical context. The bottom line is that after generalization, some records (e.g., records with Ids 1, 2, 3 in Table 2.2) would become identical when projected on the set of quasi-identifier attributes (e.g., Age, Zipcode, and Gender).
Typically, generalization uses a value generalization hierarchy (VGH) for each attribute. In a VGH, the attribute values are represented by leaf nodes, and internal nodes portray less-specific values. Figure 2.2 shows a VGH for the occupation attribute. Generalization schemes can then be defined based on the VGH that specify how the data will be generalized.

VGH basically represents the semantic information about a value v to generalize in order to obtain a general value(s) to which v can be coarsened. As in Figure 2.2, a VGH is basically a tree where leaf nodes are the domain values (e.g., if the domain comprises of a *Doctor* then *General physician* is a leaf) and root node corresponds to the most general value (e.g., often pointed out by ANY or *) and any path from root to a leaf consists of nodes representing decreasing levels of generalization. Nodes participating in generalization hierarchies are said to be *generalization nodes*. We denote the generalization

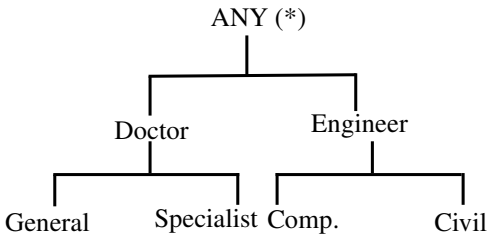


Figure 2.2: VGH for *Profession*

relationship between two nodes a and b as $a \succeq b$ i.e., a generalizes b. *Specialization* is the reverse operation of generalization and a specializes b is denoted by $a \preceq b$. It is important to mention here that these hierarchies can be either static (i.e., provided by domain experts depending upon the characteristics of the attributes) or dynamic (i.e., dynamically created by the generalization algorithms)

Equivalence class:

The generalization is normally performed on quasi-identifier attributes but sensitive attribute values can also benefit from it [100]. The main idea behind generalizing quasi-identifiers is to obtain (multi)sets of records such that all records in a (multi)set share identical generalized values on every quasi-identifier attribute. Further simplifying the idea of generalization nodes, we say that the generalization nodes are basically the generalized quasi-identifiers. A set of records accompanied by its generalization node is termed as an *equivalence class*. Depending upon the privacy model to implement, equivalence classes need to follow certain rules/constraints e.g., k-anonymity model enforces the equivalence classes to contain at least k records while (distinct) ℓ-diversity requires that each equivalence class must contain ℓ distinct sensitive values. Then, a generalization-based algorithm inputs a relation R with q quasi-identifier attributes, $QI_1, \ldots, QI_q \in QI$ and outputs a set of equivalence classes. The i_{th} equivalence class for a set of tuples t and a generalization node γ denoted by $[t]_i$ such that $\forall t \in [t]_i$ and $\forall QI_j \in QI, t[QI_j] \preceq [t]_i.\gamma[QI_j]$.

Finding the Best Generalization is Hard

It is worth noticing that a dataset can be trivially generalized to only one equivalence class having the generalization nodes consisting of roots of all the VGHs i.e., replacing each quasi-identifier with the most generalized value e.g., *ANY*. Obviously, generalization-based algorithms aim at finding a generalized release under the umbrella of a given privacy model such that utility is increased up to the hilt (e.g., one can chose the generalization nodes as close as possible to the leaves in VGH). Practically, there can be many such possible generalizations. Then, the most important question

is *"How to find an optimal generalized release?"*. Meyerson et al. [77] and Bayardo et al. [8] showed that finding the optimal generalized k-anonymous release is NP-hard owing to the combinatorial explosion in the number of possibilities for best generalized releases. Due to this reason, the above mentioned question has taken a new form: *"How to find a good approximation of the optimal generalized release"*. To answer this question, a large number of generalization-based algorithms have been proposed e.g., [6, 8, 65, 66, 77, 82, 90, 95]. We formalize the notion of a generalization mechanism (algorithm) in next section.

Generalization Algorithms

In centralized data publishing scenario, a data publisher has an input relation R containing personal data of the individuals from the population. The relation R has a schema $R(ID, QI, S)$. Following the literature convention, we assume that quasi-identifiers (*QI*) are either categorical, ordered or continuous, the *only* sensitive attribute is categorical and ID is removed before publication. For an input relation R, $\pi(R)$ and $\sigma(R)$ corresponds to the projection and selection respectively on R. For any tuple $t \in R$, $t[A]$ denotes the A field value of tuple t.
We consider the problem of transforming the input relation $R(ID, QI, S)$ into a sanitized release, denoted by $R^*(ID, QI, S)$ such that R^* is immune to linking attacks [1]. In order to achieve some privacy requirements for R^*, it is assumed that the explicit identifiers (ID) are removed in the public release and R^* can be obtained by applying the generalization mechanism as given by the Definition 2.4.
The generalization-based algorithms can either belong to *global recoding*, *local recoding* or *regional recoding* algorithms [65, 66]). In global recoding, the values are generalized to the same level of the hierarchy. One effective search algorithm coined *Incognito* for global recoding is proposed by LeFevre et al. [65]. There are several advantages of global recoding scheme:

- The strategy has conceptual simplicity.
- It is usually tractable to obtain an optimal solution [8].
- Finally, the inferences among the remaining attributes stay uniform with the original dataset.

Regional recoding [56, 66] allows different values of an attribute to be generalized to different levels. For example, Given the VGH for profession in Figure 2.2, one can generalize *Computer Engineer* and *Civil Engineer* to *Engineer* while leaving *General physician* and *Specialist* as is. Iyengar [56] used genetic algorithms to perform a heuristic search in the solution space and LeFevre et al. [66] applied a *kd*-tree approach to obtain a sanitized release.

1. Additional problems and inferences arise when micro-data is dynamic and there are multiple different sanitized versions of such micro-data. This problem is detailed in Chapter 4 which constitutes the main problem handled by this thesis

Local recoding allows the generalization of a same value to the different values in different records. For example, consider three records with the value *Computer Engineer*, this value may be generalized to *Any (*)* for the first record, *Engineer* for the second one, remains as is for the third record. Local recoding normally incurs less information loss than global recoding but it is naturally more expensive to find an optimal solution due to very large solution space which makes it a hard problem [7].

2.4.2 Bucketization Algorithms

Xiao et al. [110] proposed the bucketization algorithm to achieve ℓ-diversity by (1) minimizing the information loss (2) improving the efficiency as compared to generalization-based algorithms. bucketization surmises that there is only one sensitive attribute in a given dataset. In pursuance of minimizing information loss, Xiao et al. proposed to blur the association between quasi-identifiers and sensitive attribute values by producing the groups of records, assigning identity to each such group and then simply release the quasi-identifiers as-is and sensitive attribute values in two separate tables namely *quasi-identifier table* (QIT) and *sensitive table* (ST). The association between the QIT and ST is maintained via the identity of each group in QIT table which serves as a foreign key in ST table. The authors of [110] term this way of data publication as *anatomy* which has some similarities with *permutation* [117]. Cao et. al [16] proposed *sabre* algorithm for the implementation of t-closeness privacy model via bucketization. Generally, t-closeness privacy model forces the distribution of sensitive values in an equivalence class close to overall distribution of sensitive values (See Section 2.3.1.3). This overall distribution of sensitive values is meticulously transformed in the buckets produced by sabre bucketization algorithm thereby improving the utility and efficiency up to the hilt. m-invariance privacy model for dynamic data publication problem also makes use of bucketization. We will explain m-invariance bucketization algorithm scrupulously in Chapter 4.
While bucketization produces an effective data analysis [16, 110], it is unable to prevent membership disclosure in the sanitized release. Further studies on the bucketization approach also highlight its limitations. For example, the bucketization algorithm proposed by Xiao et al. [110] is shown to be particularly vulnerable to background knowledge attacks [73].

2.4.3 Other Algorithms

Other popular syntactic algorithms include clustering [6, 13, 23], microaggregation [29], space mapping [49], spatial indexing [55], and data perturbation [4, 89, 98]. Microaggregation [29] starts by grouping the records into small aggregates comprising of at least k records in each aggregate and finally publishes the centroid from each aggregate. Aggarwal et al. [6] proposed to achieve anonymity via clustering records into

group of at least size k and finally releasing statistics for each identified cluster. Byun et al. [13] proposed k-*member clustering* algorithm that aims at minimizing some specific cost metric. Iwuchukwu et al. [55] showed the striking similarity between spatial indexing and k-anonymity and proposed to use spatial indexing techniques for data sanitization. Ghinita et al. [49] presented a two fold solution for data sanitization. They first proposed heuristics for sanitizing *one-dimensional data* (i.e., the quasi-identifier comprising of a single attribute) and secondly, they proposed a sanitization algorithm that executes in linear time. The authors of [49] presented a space mapping technique for transforming multi-dimensional data into one-dimensional data before applying the algorithm for one-dimensional data. Below we review several space partitioning algorithms for k-anonymization and highlight their weakness in achieving efficient k-anonymous release.

Partitioning Schemes for PPDP

It is worth to notice that public release with one single equivalence class described on each dimension by the all domain is obviously k-anonymous ($k \leq n$ the number of records) but it is definitely useless for the end-user. Thus, the main challenge of k-anonymization is to compute a public release where the information loss has been minimized, in the sense of a general criteria by means of popular utility metrics discussed in Section 2.5. This optimization problem was proved to be NP-hard [77].
Hence, many approximation algorithms have been proposed in the literature since the seminal work of Sweeney [95]. Usually, Mondrian approach [66] is thought as the baseline algorithm since it has the basic good properties we could expect from such algorithms: local recoding and multidimensional partitioning. Mondrian iteratively operates a binary partitioning of the data space until every block contains between k and $2k-1$ data points. Actually, Mondrian builds a *kd*-tree over the raw data and publishes bounding boxes of the leaves as equivalence classes of the anonymous release. Construction has time complexity *O(N log(N))*, where *N* is the number of records in raw data.
Following the geometric representation of the data, Iwuchukwu et al. [55] propose to use a bulk-loading implementation of an R^+-tree, one of the most popular spatial access methods for databases, to compute the k-anonymous release. It outperforms Mondrian thanks to buffering and efficient bottom-up index construction algorithm, and it scales up to very large data sets. Furthermore, the hierarchical structure of the R^+-tree natively supports $(B^\ell k)$-anonymity for all level ℓ in the tree, with B the fanout parameter. And with an ordered leaf scan, it could support (cK)-anonymity as well, for all c in $\mathbb{N}$. Time complexity remains in *O(N log(N))*. And I/O cost for external computation is in $O(\frac{N}{B} log(\frac{N}{B}))$.
Since the R^+-tree bulk-loading algorithm is applied on a set of points rather than a set of spatial objects with an extent, it is actually a variant of a *kd-B*-tree structure where

hyper-rectangles have been shrinked to the minimum bounding boxes (MBB) of the subset of points in each equivalence class. Remind that a *kd-B*-tree is a bucket-oriented variant of a *kd*-tree where the fanout of each node is defined by a parameter B that usually fits the disk block size. The many good features of the R^+-tree approach makes it therefore the reference algorithm for k-anonymization up until now.
Many works also proposed point partitioning structures in low dimension (2-3D) for privacy preserving location-based queries [27, 48, 51, 80]. In this application domain, privacy is related to instant location of users and queries as well. Popular approaches design an anonymizer that dynamically provides a Cloaking Region to the Location-Based Service. For that purpose, Gruteser et al. [51] implements a *kd*-tree, whereas Mokbel et al. [80] uses a variant of a PR quadtree in *Casper*. Ghinita et al. [48] accommodate partitioning structures from kd-tree and R-tree to hash a database of Points Of Interest (POI) and answer approximate nearest-neighbor queries in a Privacy Information Retrieval (PIR) approach (*A PIR protocol allows a user to fetch a tuple from a database while concealing the identity of the tuple from a database server*) [24]. They also consider Hilbert space filling curves to map 2D points to single-dimensional data structures like B^+ -trees to index POIs. Actually, they argue that their PIR approach is independent from the partitioning structure as far as it provides at most $\sqrt{N}$ buckets within up to $\sqrt{N}$ POIs each. Other work [63] focused on geo-privacy in the sense of privacy-preserving location data publishing. In this context, a space filling curve was also employed to order both data points and POIs on the map. Quad-trees and space filling curves do not scale for higher dimensions, and the latter cannot guarantee non overlapping bounding boxes in the worse case.
The above short review states that every approach to geo-privacy accommodates in memory and implements well-known structure for multi-dimensional point data partitioning. k-anonymity was also studied from the cardinality constraint clustering point of view. On one hand, the anonymization algorithms were proposed [6, 13, 23] that achieve good quality, whereas neither they scale up in the size of the data set, nor they meet the basic orthogonal range query requirement since patterns are spheres (centers and radius) of each cluster. On the other hand, many grid clustering techniques ([85, 105] for a short excerpt) have been proposed. However, none of them are as fast and scalable as Point Access Methods (PAM) since external storage support and dedicated insert-delete-search operations are missing. Then, PAMs remain the preferred logical structures for the anonymization of very large data sets. In Chapter 3, we propose to use an efficient point access method i.e., Bang-file, for k-anonymization which overcomes the above mentioned problems. Extensive experimentation further strengthen our in-depth analysis favoring PAMs for PPDP tasks.

Data Perturbation

Data perturbation [4, 89, 98] is another sanitization method. It serially perturbs each record in a given dataset. Given a record t, the algorithms retain the sensitive value s of t with a probability p and replaces s with a random value from the domain of sensitive attribute with probability $1 - p$. The limitation of perturbation based algorithms is that p needs to be very small in pursuance of preserving privacy i.e., it is difficult to control noise injection. Consequently, the data may contain noise greater than expected thereby reducing the usefulness of final release for the end users [69].

2.5 Data Utility

A data publisher aims to publish the data that are not only protected but also useful. In order to provide sufficient level of data protection, privacy algorithms distort the data such that no single individual can be uniquely identified. For instance, a dataset can be trivially generalized to only one equivalence class by suppressing every quasi-identifier. This approach gives maximum privacy however resulting data becomes useless. Since sanitized data must allow search and analysis tasks, it is required to achieve good trade-off between privacy and utility. Thus, the utility of sanitized data is ostensibly measured by the degree to which it maintains the usefulness of statistical and aggregate information.
In general, the utility of sanitized data can be evaluated by two approaches. The first approach is to exploit one or more quantitative measures for information loss and the second one is to actually employ the data as input to a query, and assess the accuracy of the results. Numerous utility measures have been studied in the literature. We will discuss a small set of these measures and refer interested readers to [42] for a more detailed survey.

2.5.1 General Utility Measures

The main idea of this approach is to evaluate the extent to which the sanitized data has been distorted. The popular utility measures include generic quality measures, i.e., measures that do depend neither on the application domain nor on a specific usage of the sanitized release i.e., *discernibility penalty* [8], *KL-divergence* [59] and the certainty metric proposed in [114].

The three quality measures are explained below:

2.5.1.1 Discernibility Penalty (DCP)

Discernibility Penalty (DCP) assigns a penalty to each record based on the number of the tuples in the database that are indistinguishable from it. If a tuple belongs to an equivalence class EC of size $|EC|$, the penalty for the tuple will be $|EC|$. Thus, the penalty on the equivalence class is $|EC|^2$. The overall DCP of sanitized release R^* is given by:

$$DCP(R^*) = \sum_{EC \in R^*} |\, EC \,|^2 \tag{2.5}$$

2.5.1.2 Certainty Penalty (CP)

Certainty Penalty (CP) evaluates the loss of accuracy in the description of equivalence classes. Consider a sanitized table R^* obtained from a raw table R having q quasi-identifier attributes, $QI_1, \ldots, QI_q$. Suppose there exist a global order on all possible values in the domain of all QI attributes. If a record r in R^* has an interval $[x_i, y_i]$ on an attribute QI_i $(1 \leq i \leq q)$, then the Normalized Certainty Penalty (NCP) in r on QI_i is given by:

$$NCP_{QI_i}(r) = \frac{|y_i - x_i|}{|QI_i|}$$

where $|QI_i|$ stands for the domain of attribute QI_i. For a record r, the NCP on r is given by:

$$\sum_{i=1}^{q} w_i . NCP_{QI_i}(r)$$

where w_i correspond to the weights of attributes. Finally the CP for R is given by:

$$\sum_{t \in R^*} NCP(t) \tag{2.6}$$

2.5.1.3 KL-divergence

The discernibility penalty and certainty metric are oblivious to the overall distribution of attribute values in the data. For this reason, Kullback-Leibler or KL-divergence for short [59], is a commonly used utility metric in statistical community as it is more appropriate for measuring the information loss of sanitized data when data distribution is also a consideration. To employ KL-divergence, raw table is employed as a probability distribution p_1 i.e., for a record r, $p_1(r)$ is the fraction of records equal to r. The sanitized table is also transformed into probability distribution p_2. There are various ways of converting sanitized data into a probability distribution. We refer the interested readers to the work of Chen et al. [19] for the possible ways of achieving this

conversion. KL-divergence for p_1 and p_2 is given by:

$$KL(p_1, p_2) = \sum_r p_1(r) log \frac{p_1(r)}{p_2(r)} \tag{2.7}$$

2.5.2 Query Workload

This approach aims at measuring the utility of a sanitized release in terms of accuracy in answering aggregate queries. For answering the aggregate queries, the "COUNT" operator is considered in which the query predicate includes quasi-identifier attributes. Let R be a table with q quasi-identifiers, $QI_1, \ldots, QI_q$ where $D(QI_i)$ denotes the domain of i_{th} quasi-identifier. Then, the queries are of the form:

SELECT COUNT$(*)$ **from** R
WHERE $qi_1 \in D(QI_1)$ **AND** $\ldots$ **AND** $qi_q \in D(QI_q)$

The predicate of a query contains two important parameters (1) the query dimensionality parameter q and (2) the query selectivity θ. The query dimensionality parameter q indicates the number of quasi-identifiers used in the predicate. The query selectivity θ indicates the number of values for each attribute A_j, $1 \leq j \leq n$. Query selectivity is usually obtained as follows:

$$\theta = \frac{|T_Q|}{|R|} \tag{2.8}$$

where $|T_Q|$ is the number of tuples in the result set obtained from Q on R and $|R|$ is the number of tuples in dataset. The error for the query Q, denoted Error(Q), is the normalized difference between the result set from the evaluation of Q on raw and sanitized data respectively. Then the query error is calculated as follows:

$$\text{Error}(Q) = \frac{\text{sanitized\_count} - \text{actual\_count}}{\text{actual\_count}} \tag{2.9}$$

where the result from the COUNT query on R is denoted by *actual_count* and on R^* as *sanitized_count*.

2.6 Semantic Privacy Definitions

To attain a worthwhile instantiation of data privacy, it is important to quantify the adversarial knowledge about sensitive data that he/she gains by observing the sanitized release. These definitions are termed as *semantic* because they acquire such variation in the adversarial background knowledge. Semantic privacy in statistical context) protection has recently gained popularity for keeping the secrecy of the individuals whether they belong to a given dataset or not. For instance, consider a published dataset that can be used to compute the average taxes paid by the doctors in the city of Nantes. Consider

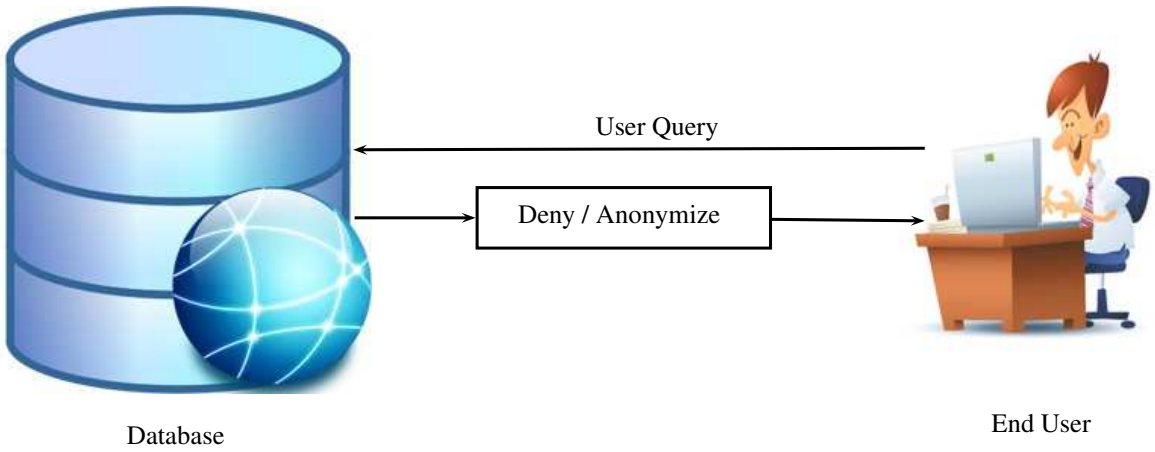


Figure 2.3: Interactive semantic privacy

an adversary who knows that his friend, who practices in Nantes, pays €1500 less than the average taxes paid by the doctors in Nantes. Although this piece of information may not be useful for the adversary, but combining this knowledge with the access to a published version of the dataset may raise privacy concerns. It is worth to notice that such privacy issues do not depend on whether the individual (the adversary's friend in the above example) may belong to the published dataset or not. Also, even with this aggregate information i.e., average taxes, it is possible to infer individual values with proper background knowledge.

Semantic privacy definitions do not make any assumptions on the background knowledge of the adversary. In semantic privacy, the quasi-identifiers and sensitive attributes get the same treatment. This is because making any such distinction is basically making assumptions about the adversarial background knowledge. Semantic privacy comes in two flavors:

- In *interactive semantic privacy setting*, the dataset is not published. The information is protected inside a database and access to the dataset is granted through an interface (via querying). The answer set returned by the interface is processed in a way to guarantee the privacy of concerned individuals. Figure 2.3 depicts the scenario of interactive semantic privacy setting.
- In so-called *non-interactive semantic privacy setting*, the sanitized dataset is published once for all while still keeping the privacy of individuals intact. Figure 1.1 is an example of non-interactive semantic privacy setting.

We provide an insight of semantic privacy in non-interactive settings and refer the interested readers to the surveys [28,32] for in-depth analysis on interactive semantic privacy approaches. In what follows, we overview differential privacy (referred to as DP afterwards), the most renowned semantic privacy approach. Since the original definition of DP is strict and it has been questioned frequently for its applicability in real life sce-

narios, we also overview several flavors of DP that try to relax the definition of original DP .

2.6.1 Differential Privacy

Though initially proposed for interactive query answering over a static private dataset [31, 35], the DP has shown great potential not only for non-interactive privacy preserving data publishing but it is also shown to preserve the uninformative principle [88]. DP has recently attracted growing attention from not only the database and security research groups [34, 54, 60–62, 88] but also from general computer science community [33, 50, 81].
Informally, DP enforces the constraint that small changes in the private data set should only incur small changes in the output distribution of a sanitization mechanism applied on the data. DP aims at characterizing the sanitization mechanisms - these mechanisms must be randomized such that given two input datasets differing by only one tuple, the output distribution from the mechanisms must not differ much. DP comes in two flavors namely, bounded and un-bounded [60].

Definition 2.7 (Unbounded DP [31, 60]) *A randomized mechanism $\mathcal{M}$ satisfies unbounded ϵ-DP if for any subset S and any pair of datasets D, D' such that D can be obtained from D' by either inserting or deleting exactly one tuple, following condition holds:*

$$\frac{Pr(\mathcal{M}(D) \in S)}{Pr(\mathcal{M}(D') \in S)} \leq e^{\epsilon} \tag{2.10}$$

Definition 2.8 (Bounded DP [35, 60]) *A randomized mechanism $\mathcal{M}$ satisfies bounded ϵ-DP if for any subset S and any pair of datasets D and D' such that D can be obtained from D' by modifying exactly one tuple.*

$$\frac{Pr(\mathcal{M}(D) \in S)}{Pr(\mathcal{M}(D') \in S)} \leq e^{\epsilon} \tag{2.11}$$

Bounded DP is also referred to as ϵ-indistinguishability.
In above definitions, ϵ is a constant specified by the end user that provide some kind of privacy budget. Intuitively, given the set of outputs S for the mechanism $\mathcal{M}$, it is hard for the adversary to infer whether the original dataset is D or D' given that the parameter ϵ is sufficiently small. Similarly, ϵ-DP also provides any individual with *plausible deniability* that her record was in the dataset [115].
The anticipative and most widely-embraced approach for the implementation of DP is Laplace mechanism [31] which incorporates random noise to obtain a randomized version of a given mechanism $\mathcal{M}$. Normally the distribution envisaged for adding the noise is Laplace distribution i.e., $Laplace(\Delta(d)/\epsilon)$ alongside a probability density function

$P(y) = exp(|y|/k)/2k$, where $k = \frac{\Delta(d)}{\epsilon}$ and $\Delta(d)$ corresponds to maximum difference between the result sets returned by a query on D and D' (that, for instance, will be 1 for count queries over D and D' since they differ by at-most 1 tuple).
The DP techniques for non-interactive settings typically publish marginals or contingency tables of the micro-data [35, 113]. The main theme of these techniques is to first compute a frequency matrix (frequency matrix - for a contingency table, is computed over all attributes, whereas for a marginal, it is calculated by projecting certain attributes) for micro-data over the domain of database. The next step is to add the noise to each count for specifying the privacy requirement. Finally these techniques publish the noisy frequency matrix. Mohammed et al [79] identified that this approach may not be efficient for high-dimensional data having large domain, due to the fact that the added noise becomes relatively large as compared to the count thereby severely degrading the utility.

2.6.2 Relaxing the Differential Privacy

The original definition of DP is very strict and in some scenarios lesser version of this definition may be acceptable for the data publisher which might be possible by relaxing some constraints while maintaining a weak form of DP. Below we summarize some well-known relaxations for DP:

(ϵ, δ)-Differential Privacy

Definition 2.9 ((ϵ, δ)-DP [37]) *A randomized mechanism $\mathcal{M}$ satisfies (ϵ, δ)-DP if for any subset S and any pair of datasets D and D', following condition holds:*

$$Pr(\mathcal{M}(D) \in S) \leq e^{\epsilon}.Pr(\mathcal{M}(D') \in S) + \delta \tag{2.12}$$

where δ is a small additive error probability to the size of the given dataset. The introduction of δ may possibly cause a higher privacy risk than ϵ-DP but on the other hand it reduces the addition of noise thereby permitting better accuracy in the sanitized release.

Computational Differential Privacy

The original version of ϵ-DP provides protection against computationally unbounded adversaries. This notion may become an overhead in the scenario in which the adversary has limited computational resources. Thus, there is room for relaxing ϵ-DP considering the realistic adversaries [78]. Consequently, this approach also results in limiting noise addition. Computational DP comes itself in two flavors:

Definition 2.10 (Indistinguishability-based Computational DP) *A randomized mechanism $\mathcal{M}$ satisfies bounded ϵ-DP for any two datasets D and D', if an (realistic) adversary is unable to characterize (with non negligible probability) the result of the evaluation of $\mathcal{M}$ over D from the result of the evaluation of $\mathcal{M}$ over D' such that D can be obtained from D' by modifying exactly one tuple.*

This definition goes back to the definition 2.8 of bounded DP and replaces an unrestricted adversary with a *computationally-bounded one*.

Definition 2.11 (Simulation-Based Computational DP) *Simulation-based approach simulates the vision of an adversary through an arbitrary randomized mechanism D'. If this simulated result is computationally indistinguishable for the real sanitizing mechanism D then D satisfies (computational) DP.*

The other popular extensions for DP include *Pan Privacy* [38], *Pufferfish framework* [61] and *Differential identifiability* [64].

2.7 Towards a Unified Approach of Syntactic and Semantic Privacy

The research community has left no stone unturned in devising strategies for both syntactic and semantic privacy definitions. The literature on privacy protection reveals that no privacy model is capable of incorporating growing demands of data publication (e.g., the adversarial background, needs of data publisher, constraints on underlying dataset etc.). Thus, despite the countless efforts, privacy protection remains an open issue.

2.7.1 Open Problems

Syntactic privacy definition, being widely studied for PPDP task, requires assumptions that make them questionable w.r.t privacy guarantees in critical applications. As described in Section 2.3, each syntactic approach is based on an attack model of an adversary and it assumes that such an adversarial knowledge is limited and is predefined. Consequently, these approaches fail to provide the promised degree of protection if the adversarial knowledge exceeds the protection level provided by the given privacy model. In short, it is *difficult to impossible* to model the adversarial background knowledge.
Semantic privacy definition e.g., DP, was introduced to overcome the inherent deficiencies in syntactic privacy approaches but its applicability in real life situation is questioned frequently.
As DP model does not make any assumptions about how data is collected and generated,

it may be insufficient to protect the privacy against the adversaries who are interested in an individual's existence in the released data. Since the deletion of a record may not hide every trace from the released table, adversary can exploit this fact to infer the individual's participation in the released data. Kifer et al. [60] clarify these underlying assumptions of DP model about data. Specifically they proposed an extension of DP that gets rid of these assumptions. Also, they showed that the DP is prone to privacy breach against arbitrary background knowledge. DP has attracted the research community for its ground breaking semantic privacy approach for static data publication [60, 61]. Employing DP for dynamic data publication (Section 2.3.1.5) may be cumbersome for the same reason as in the case of repeated queries to a differentially private system. Intuitively, by time, when the noise that must be added increases and there are bounds of privacy budget (ϵ), the adversary has ample chance to use differencing to detect and remove the noise [10, 36]. Also, Yang et al. [115] raise some major questions regarding the applicability of DP specially in dynamic settings. Firstly, it is difficult to define the DP protocols when it encounters *arbitrary updates*. Secondly, it is difficult for an end-user to choose the privacy budget (ϵ) thereby maximizing the utility of the output of a DP mechanism.

2.7.2 Relaxing Semantic Privacy Definitions for Syntactic Approaches

Recently, there is a trend of relaxing the DP so that both syntactic and semantic privacy approaches can flourish together in order to remove each others deficiencies.
Gehrke et al. [46] proposed to exploit the adversary's uncertainty about the underlying dataset. The authors of [46] stated that adding a random sampling step provides a natural way in capturing the adversarial uncertainty about the input dataset. Consequently, they initiated a new privacy definition coined *Crowd Blending Privacy* that permits to design new mechanisms having better applicability regarding utility/efficiency than differentially private mechanisms while keeping the notion of privacy intact. Moreover, they force these mechanisms to satisfy the crowd blending privacy in pursuance of achieving differential privacy when the underlying dataset is randomly sampled from the given population.
Gehrke et al. noticed that k-anonymity is based on the premonition of "blending in a crowd", since the records in a k-anonymous sanitized release are required to "blend" with at least $k-1$ other records. Ostensibly, the idea of blending in a crowd of many people is sufficient to protect the privacy of concerned individuals. However, as shown by several known attacks, k-anonymity is unable to fully capture this notion of "Crowd Blending", because it does not impose any constraint on the mechanisms used to provide the k-anonymous release.
One of the important directions given by the authors of [46] is the adaptation of generalization based k-anonymity solution to DP. They maintain that if generalization is not

done carefully, the privacy of individuals is at risk. However, they show if the generalization step is performed gingerly, these generalization-based k-anonymity algorithms can satisfy crowd blending privacy.

Definition 2.12 (Crowd Blending Privacy [46]) *A sanitization mechanism $\mathcal{M}$ satisfies crowd blending privacy if for every dataset D and every individual $i \in D$, either i ϵ-blends in a crowd of k people in D w.r.t $\mathcal{M}$, or $\mathcal{M}(D) \approx_\epsilon \mathcal{M}(D \backslash \{i\})$ (or both).*

Crowd blending privacy compels the mechanisms either to blend an individual i in a group of k individuals or do not release i's data at all. This way the mechanisms actually do not release any information about i, apart from the general properties of the crowd of k individuals. Furthermore, the authors of [46] prove that DP implies crowd blending privacy i.e., by removing the condition $\mathcal{M}(D) \approx_\epsilon \mathcal{M}(D \backslash \{i\})$ from the definition of crowd blending privacy, results in DP.
Li et al. [72] in an open publication, proposed a notion of "safe" k-anonymity and argued that safe k-anonymity preceded by a *random sampling* step satisfies (ϵ, δ)-differential privacy. The authors proposed a relaxed differential privacy definition under sampling:

Definition 2.13 ($(\beta, \epsilon, \delta)$-DP [72]) *Given a dataset D, a sanitization mechanism $\mathcal{M}$ satisfies $(\beta, \epsilon, \delta)$-DP iff $\beta > \delta$ and a mechanism $\mathcal{M}^\beta$ satisfies (ϵ, δ)-differential privacy such that $\mathcal{M}^\beta$ samples the tuple from D with a probability β.*

This interesting trend of combining DP with generalization-based approaches has given the opportunity to the researchers to blend the strength of DP with the efficiency of state-of-the-art generalization algorithms for practical privacy. Though this research area is in the initial stages, we may in the near future benefit from both research tracks. We invite the interested readers to refer to [46, 72] for in-depth analysis of these approaches.

2.7.3 Conclusive Statement

Keeping the above deficiencies of both syntactic and semantic privacy definitions, the question arises "Whether the PPDP task is forlorn?". We insist that, in order to cope with the growing demands from data recipients, several privacy models and algorithms have been proposed which take into account certain scenarios depending upon the structure of underlying data, the possibilities of inferences etc. As mentioned in Section 2.7.2, there is an encouraging trend to find an approach that might follow One-Size-Fits-All chimerical.

3

BangA

Summary: *This Chapter aims at developing a generalization-based privacy algorithm using spatial indexes with the intent of improving utility of sanitized release.*
One of the major reasons for the popularity of non-interactive PPDP is the belief that the data could be sanitized with very little information loss; this turns out to be true if removing identifying attributes only, guarantees ample privacy. However, when micro-data has to satisfy a stronger privacy criteria, a substantial loss of information may incur. For instance, Aggarwal [3] has shown that sanitizing sparse high-dimensional data (data with large number of attributes) adversely affects the overall utility of the final release. Hence, any such research on privacy preserving data publication that does not take into account the enforcement of privacy guarantees and the utility of sanitized data simultaneously, is incomplete in nature.
The familiar area of spatial indexing has been shown to have a striking parallel with data sanitization [55]. This Chapter starts by providing an in-depth review of spatial indexing techniques that can be used for data sanitization. Point Access Methods (PAM) are logical structures that efficiently organize a set of points for enhancing search facilities. The PAMs have many desirable features that are suitable for the problem of data sanitization. We argue in a detailed study that Nested Hyper-Rectangular based Bucketed Point Access Methods, NHR-based BPAMs for short, happen to be the most effective and efficient logical structures for PPDP tasks. To follow on the analysis, we also review the clustering systems like GRIDCLUS [92] and BANG-clustering [93] since they provide extremely efficient clustering solutions by combining clustering with PAMs namely Grid File [84] and Bang File [40] respectively. The remaining part of Chapter 3 proposes BangA, an efficient sanitization algorithm based on BANG-clustering. Extensive experimentation shows that BangA outperforms traditional k-anonymization algorithms thanks to its effective structure and ability to scale up. Since it is based on a spatial index, BangA can be used

as-is for sequential data anonymization (in a limited capacity). Also, it is capable to incorporate more sophisticated generalization models e.g., ℓ-diversity with slight change in its splitting strategy. At the end, it makes BangA a first-class algorithm for a large family of sanitization tasks.

Contents

3.1 Introduction

Organizations may release their microdata for the purpose of facilitating useful data analysis and research. For example, patients medical records may be released by a clinic for research organizations. Releasing this kind of data about individuals without risking their privacy has been an important problem. To obviate personal identification, many organizations usually remove the uniquely identifying information like name, SSN from the published data. However, this sanitization of data might not be helpful in guarding the secrecy of given individuals as it is still possible to link released records back to their identities by matching some combination of quasi-identifier attributes. This gave rise to the need for robust sanitization methods to publish sensitive individual data keeping their privacy intact. The seminal k-anonymization paradigm [95] was proposed to achieve this goal by means of a generalization model. Basically, anonymization based on generalization consists in decreasing the accuracy of values from quasi-identifiers. For instance, 44100 Zip code would become 44XXX and 70 pounds would be said to range between 50 and 80 pounds.

The k-anonymity model proposed by Samarati and Sweeney [90] provides a practical solution for Privacy Preserving Data Publication (PPDP) and has been studied extensively in the last two decades. Anonymization via generalization and/or suppression is able to protect the privacy of individuals, but at the cost of information loss especially for high-dimensional data. This is due to the fact that generalization based k-anonymity is impeded by the curse of dimensionality as shown by Aggarwal [3]. Furthermore, in order to achieve an effective generalization, the tuples in the same equivalence class ought be close to each other so that the generalization may not lose too much information. Nevertheless, high-dimensional data forces greater amount of generalization to satisfy basic requirement for k-anonymity even for relatively smaller value of the parameter k. Hence, it is important to consider deeply the trade-off between privacy and information loss. Thus, the motivating question in this context is *how to minimize the information loss in the course of generalization specially for high-dimensional data*.

This Chapter presents *BangA*, a new generalization algorithm that meets generic PPDP features as described in 3.2, and that offers several new desirable features in regard to many other existing approaches, and especially compared to the R^+-tree based anonymization algorithm [55].

Though BangA generalization algorithm can be extended to achieve any generalization model e.g., ℓ-diversity or *t*-closeness, we implemented BangA to achieve k-anonymous public release for the following reasons:

- k-anonymity is conceptually simple;
- k-anonymity does not enforce any constraint on the distribution of sensitive values in public release. This is one of the main reasons it can be extended to achieve more stronger notions of privacy e.g., *differential privacy*(DP). As mentioned in Section 2.7.2, there is an encouraging trend of combining DP style privacy with

Id	Age	Zip Code	Gender	Disease
1	[48-62]	441XX	*	Flu
2	[48-62]	441XX	*	Flu
3	[48-62]	441XX	*	HIV
4	[59-77]	444XX	*	Flu
5	[59-77]	444XX	*	Gastritis
6	[59-77]	444XX	*	HIV

Table 3.1: 3-Anonymous public release

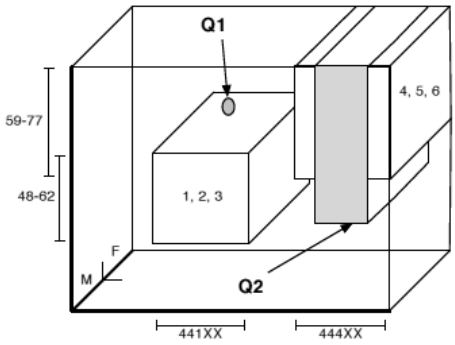


Figure 3.1: 3D spatial representation of the anonymous public release from Table 3.1 with point query Q_1 and window query Q_2.

syntactic approaches. Specifically the works in [72] and [46] provide interesting directions to achieve DP for k-anonymity based generalization approaches.

This Chapter is organized as follows: Section 3.3 states the sanitization problem along with the assumed adversarial knowledge. Section 3.4 discusses the main recipe for BangA. Sections 3.5 and 3.6 explain how the raw data can be transformed to anonymous public release using BangA. Section 3.7 evaluates the effectiveness of the proposed approach. Section 3.8 discusses various extensions for BangA.

3.2 Spatial Indexing Techniques for PPDP

This section explores the relevance of point access methods for privacy-preserving data publication and provides the foundation for Chapter 3. We also overview several privacy algorithms that make use of point access methods to achieve data sanitization. The sanitized release must support exploration, analysis and scientific studies. The very first and popular processing of sanitized release is then to search and filter tabular data by means of *point queries* and *window queries*. Indeed, regular database records can be geometrically interpreted as points in a multidimensional space where each dimension is a column of the raw table. Point coordinates are then defined by the attribute values. Transformation is obvious for numerical and ordinal variables. Categorical variables could also be equipped with a total ordering, except that without any native ordering, the process is driven by the application domain and background knowledge. Thus database queries are transformed into queries against a *set of points*.

Once the microdata is sanitized, its records become *hyper-rectangles* in a multi-dimensional space, where each dimension is a field in the set of quasi-identifiers. For

instance, sanitized records from Table 2.2 are cuboids in the 3-dimensional space (Age, Zipcode, Gender) as shown in Figure 3.1. As a point query example Q_1, user would filter data to retrieve possible patient's record designated by **Age=62** AND **Zip Code=44120** AND **Gender=F**. For sanitized release, Q_1 can be expanded for Zip Code and Gender. For instance, Zip Code can be replaced with its value in the increasing levels of its Value Generalization Hierarchy i.e., Zip Code=44120 OR Zip Code=4412X OR Zip Code=441XX OR Zip Code=44XXX OR Zip Code=4XXXX OR Zip Code=*. Similarly, Gender=F or *. Consequently, the result set for Q_1 comprises of 1, 2, 3 from Table 2.2. Similarly, a window query Q_2 for **Zip Code IN [4442X,4447X]** AND **Age>=50 AND Gender=*** comprises of the result set 4, 5, 6 from Table 2.2.

To achieve such querying scenario, the sanitized records are mutually disjoint spatial objects with a *rectangular extent* and window queries are *orthogonal range queries*. Any record that overlaps/lies within query region is a member of the result set. There exist many efficient algorithms and data structures [2] to compute such orthogonal range queries against the spatial representation of the anonymous database. Furthermore, since any orthogonal range query can be decomposed into a several 1-dimensional range queries, it is then easy to manage filters on the tabular representation of the public release within a basic spreadsheet or web-client technologies as well. Query Q_2 over Table 2.2 gives an example of such straightforward decomposition. Those practical features are very useful in lots of iterative exploration processes that would support analysis and scientific studies. *Then, we argue that the axis-parallel rectangular coding of anonymous records is a strong requirement for a generic PPDP task.*
Other kinds of window queries are defined by the shape of query region: sphere, half-space, simplex, polytopes. Sphere range queries, so-called *nearest-neighbor queries* have been extensively studied and there also exist efficient algorithms to compute such popular queries especially on rectangular objects. However, none of these range queries satisfies the decomposition property that makes anonymous releases human-friendly under tabular representation. To sum-up the above discussion, we argue that every generic PPDP task should meet at least the following theoretical and practical requirements in order to be valuable for the end-user:

1. Indistinguishability principle − to achieve generalization;
2. Mutually disjoint equivalence classes − to preserve quality of the anonymous public release;
3. Multidimensional point partitioning − to support point and range queries on the anonymous public release;
4. Hyper-rectangular coding of equivalence classes − to allow decomposition of orthogonal range queries.

In what follows, we review existing spatial structures that support anonymization algorithms, and present features of the main logical structures eligible for a PPDP task.

Then we focus on a singular kind of structures, so-called *nested hyper-rectangle-based bucketed point access methods*, that have very nice features for the anonymization.

3.2.1 Point Access Methods

Point Access Methods (PAMs) are logical structures that organize a set of points for efficient searching. We will see in this section that PAMs have features that are suitable for the anonymization problem, and as such, we argue in the following that they are the preferred data structures to support generalization algorithms.

Comparative Analysis of PAMs

For an insight into multi-dimensional Point Access Methods, we invite the interested reader to refer to the first chapter of [91]. In Table 3.2, we present a short comparison between the most popular PAMs that could be of interest for PPDP task. For the sake of simplicity, we omit the multiple extensions of each structure, available in the literature, since the main criteria of our comparison are inherent to each structure such that they remain valid whenever the extension. Basic criteria are as follows:

- *bucket?*: decides whether the PAM is bucketed or not, i.e. each element of the logical structure has a parametrized size rather than a fixed-length size. Bucket PAMs are those that could be used as spatial indices for databases since the bucket size B is set to the disk page size and then, the I/O cost of such structures is controlled. Those structures are *external or secondary storage structures* and then, they can grow as much as the size of the data set requires to, without main memory limitations;
- *orientation:* separates PAMs into 2 categories: those that decompose the underlying space, and those that aggregate the data points. The former are *top-down* since they iteratively divide the space to build the blocks, and the latter are *bottom-up* since they operate from the data to the blocks;
- *shape :* blocks of the partitioning could have various shapes in the space. The most simple but popular one is the *hyper-rectangle* (HR);
- *grid? :* decides whether pre-defined scales support the PAM or not, such that every partition line follows a grid in the space. PAMs with such feature adopt regular decomposition.
- *done :* already used into an anonymization approach? (see Section 2.4.3 above for a review).

The first 5 rows in Table 3.2 refer to in memory structures. Except the *BSP* tree, all of them build (nested) hyperrectangular ((N)HR) blocks, thus they meet the PPDP requirements as stated above. The NHR property will be discussed further later. The BSP tree builds convex polytopes that do not allow to decompose orthogonal range queries then it is not eligible for a PPDP task. The only *BD*-tree, that builds nested HRs, does not

	bucket?	orientation	shape	grid?	done
kd -tree	No	top-down	HR	No	√
kd- trie	No	top-down	HR	Yes	√
BD- tree	No	top-down	NHR	No	−
BSP tree	No	top-down	CP	No	×
PR quadtree	No	top-down	HR	Yes	√
kd-B -tree	Yes	top-down	HR	No	−
kd-B -trie	Yes	top-down	HR	Yes	−
Grid file	Yes	top-down	HR	Yes	−
R^+-tree	Yes	bottom-up	HR	No	√
hB -tree	Yes	top-down	NHR	No	−
BV -tree	Yes	bottom-up	NHR	No	−
BANG file	Yes	top-down	NHR	Yes	−

Table 3.2: Comparison of index structures for multidimensional point data. *HR* stands for HyperRectangle, *NHR* is *Nested HR*, *CP* means Convex Polytope.

support any anonymization process. All remaining rows are Bucketed PAMs (BPAMs) that are indexing structures for point databases. Among them, the 4 first structures generate HR blocks, whereas the 3 last ones provide nested HRs. The only R^+-tree was used for PPDP until now. Moreover, we claim that bottom-up spatial indexing is not systematically more efficient than top-down approaches as opposed to the conjecture from [55]. This claim is supported by our own experiments comparing in the same running environment R^+-tree approach (bottom-up) with the BANG file (top-down) in Chapter 3. Following usual analysis on spatial access methods, we claim that the performance is mainly dependent on the *splitting strategy*. In the BANG file, we use regular decomposition following the grid whereas the original R^+-tree grows by means of a quadratic procedure comparing pairwise distances of elements in an overflow bucket. Those strategies determine a constant factor (w.r.t. N, the number of points) in time complexity that makes the execution time slower for the R^+-tree. Hence, both top-down and bottom-up approaches deserve to be studied in the context of PPDP. Finally, the grid-based PAMs have the ability to support background knowledge in the space decomposition process by means of dimensional scales. Consequences are multiple. First, the block splitting strategy is straightforward since scales have been pre-defined over each dimension, so that the algorithm performs very fast. Second, the privacy requirement is governed by the user by means of the grid resolution rather than any predefined parameter e.g., the parameter k for k-anonymous public release. Obviously, grid resolution could be adapted to match any given parameter value when needed.

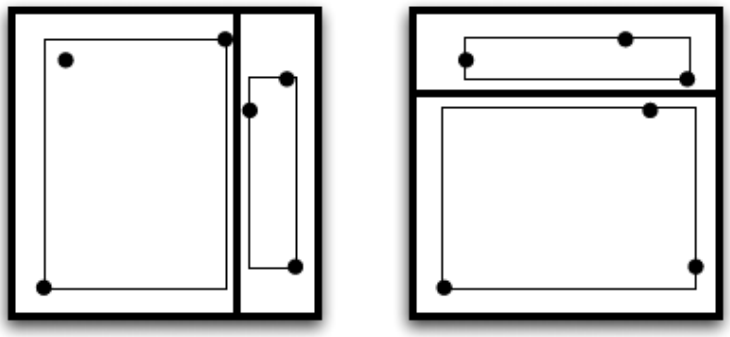


Figure 3.2: Low quality binary partitioning of a set of 6 points into blocks of at least 3 points, following either (a) X-axis, or (b) Y-axis.

Focus on Bucketed PAMs

Bucketed PAMs (BPAMs) are well-suited for the anonymization task. The very first reason is that BPAMs fulfill the basic requirements for PPDP as stated above. But BPAMs have many other nice features that could be of interest in the context of PPDP. First, since they support spatial indexing techniques in databases, they leverage 30-years research and experience in effective and efficient multidimensional partitioning data structures built from very large data sets. Thus, they scale up and perform very well. Next, BPAMs natively offer basic insert-delete-search operations that straightforwardly make the anonymization process *incremental*. It then supports dynamic updates of the dataset *before* the generation of the anonymous public release, and it provides a framework to study the open issue of continuous publication. Moreover, BPAMs require a search operation to perform at least in *O(log N)* to be efficient. Thus, they all develop a hierarchical structure, so-called *tree directory*, that makes possible *multi-granular anonymization* with partitioning extraction at any level in the tree. The only exception would be the Grid file that performs in *O(1)* such like *linear hashing*, having the main drawback of a low filling rate in each block and a large and sparse directory.

Features of NHR-based BPAMs

we argue that :

> *NHR-based BPAMs are the most sophisticated and suitable logical structures to support PPDP tasks.*

NHR-based BPAMs operate an axis-parallel space partitioning by means of nested hyper-rectangles rather than disjoint hyper-rectangles only. This singular feature allows to improve expressive power of patterns compared to other HR-based BPAMs. For example, given a set of 6 points in a 2-dimensional space, as shown on Figure 3.2;

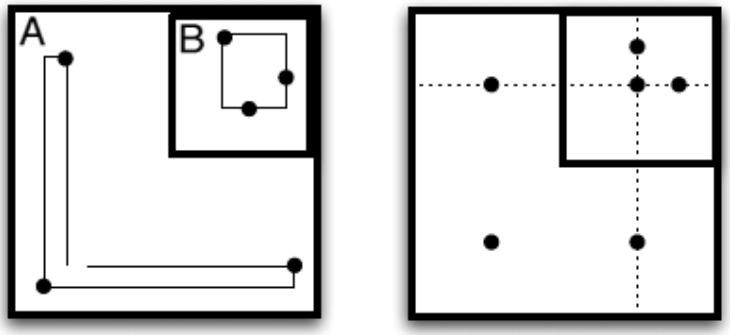


Figure 3.3: BANG file (NHR) partitioning with cardinality constraint (≤ 3 points), (a) on points from Figure 3.2, and (b) where HR partitioning fails.

assume we are trying to 3-anonymize the data set. Then, the alternative HR partitionings are those drawn on Figure 3.2. It also provides the MBBs of each block as the R^+-tree do. Similarly, Figure 3.3 shows (a) the partition obtained by a NHR-based BPAM for the same problem, and (b) a set of 6 points that can even not be partitioned with a HR-based BPAM but that can be divided within nested hyper-rectangles. Both pictures of Figure 3.3 show an outermost region A and a nested region B. Space spanned into $A - B$ forms one block, denoted by $[A]$, assigned to an equivalence class of the public release, whereas points that lie into B are the second block $[B]$. Hence, NHR-based BPAMs are known to better observe clustered values into data and also to improve the filling rate of each block since there are more flexible in the space decomposition as shown respectively on Part (a) and (b) of Figure 3.3.

Point and Range Queries against NHRs

Remind that one of the PPDP requirements is to provide user-friendly descriptions of anonymous data set to ease point and range searching in very simple but popular environments such like spreadsheets. Remind that point queries and orthogonal range queries both have the property of being decomposable into *conjunctive queries*. HR-based BPAMs are obviously tailored to fulfill such requirement. We argue in the following sections, that anonymous public releases built with NHR-based BPAMs could also support point and orthogonal range queries, without disregarding quality and efficiency of the anonymization process.

3.2.2 Synthesis

We advocated the use of Bucketed Point Access Methods for Privacy-Preserving Data Publishing tasks. We reviewed existing approaches based on multidimensional point partitioning. Then, we presented an almost comprehensive list of PAMs eligible

to the PPDP task. We ultimately claim that Nested Hyper-rectangle based BPAMs are the most promising structures to support PPDP. As a result of this study, we propose yet another generalization algorithm coined BangA, that combines very nice features from Point Access Methods and clustering. Hence, it achieves fast computation and scalability as a PAM, and very high quality thanks to its density-based clustering step. Moreover, BangA could incorporate background knowledge in the generalization process and the resulting public releases natively support orthogonal range queries.

3.3 Problem Definition

The data publisher wants to release a person-specific table such that the privacy of individuals remains intact. Recall that the microdata table R has the format:

$$R\langle ID,\, QI,\, S\rangle \tag{3.1}$$

The data publisher releases two types of information. The first being the sensitive attribute S e.g., Disease. The second type is the quasi-identifier attributes *QI*. We assume a single QI which is a combination of attributes such as *QI = Age, Zipcode, Gender*. *ID* is not released. The data publisher uses the anonymization mechanism $\mathcal{A}$ given in Equation (2.1) to generate a generalized table R^*:

$$\mathcal{A}(R) = R^* \tag{3.2}$$

where R and R^* being instances of $R\langle ID, QI, S\rangle$.

3.4 General Overview

BangA relies on an index structure so-called BANG file [40]. It operates on Axis-parallel space partitioning by means of nested hyper-rectangles rather than disjoint hyper-rectangles only. This singular feature allows to improve expressive power of patterns compared to *kd*-B-trees, including variants like R^+ -tree. BangA also supports background knowledge in the space decomposition process by means of a grid. Consequently, it has a simple splitting strategy as scales are pre-defined over each dimension thereby substantially increasing the efficiency of algorithm. BangA is also able to provide multi-granular anonymous release. To this end, it performs a density-based clustering step on blocks of the BANG file to build a dendogram. Then, a cut in the dendogram could yield high quality public releases and provide very flexible settings within one single run.
Below are the various other practically useful features of our approach.

1. Spatial indexing technique − to leverage 30-years research and experience in effective and efficient external multi-dimensional partitioning data structures built from very large data sets;
2. BANG file revisited − to accommodate a well-studied logical structure to the generalization problem;
3. Axis-parallel coding of equivalence classes − to ease orthogonal range queries for the end-user;
4. Nested hyper-rectangles − to improve quality of the anonymous public release keeping the axis-parallel coding feature up;
5. Grid-based partitioning − to make the computation faster and to control the privacy requirement by means of knowledge about the data;
6. Density-based clustering of blocks − to enforce quality of the anonymous public release in the process of block merging;
7. Multi-granular anonymization − to allow different settings for the k value with a single run of the algorithm.
8. Methodology for point and orthogonal range queries on non hyper-rectangular tabular data − to support exact match and basic range searching against anonymous public releases into spreadsheets.

Features 1, 3 and 7 are shared with at least the R^+ -tree based approach, whereas features 2, 4, 5, 6 and 8 are unique to BangA.

3.5 From Raw Data to the BANG Directory

3.5.1 Data Space Partitioning

Mapping n-tuples to the unit hypercube $[0, 1]^n$

The very first step of the overall anonymization process in BangA is to map n-dimensional raw data to the unit hypercube $[0, 1]^n$ where the BANG file is going to be defined. Records are n-tuples $\langle x_i \rangle_{1 \leq i \leq n}$ over the set of quasi-identifiers. And each field value x_i is an element of an attribute domain D_i. Mapping records $[0, 1]^n$ consists in normalizing all the n domains. The operation is then dependent from the kind of variable. For instance, a straightforward linear transformation could be used for interval variables. Ratio and additive variables are also easy to manage. Ordinal variables, isomorphic to the natural numbers, could be handled as well, whereas nominal variables would require more effort to achieve the mapping, especially to define an ordering. Here, the application domain supports the definition of the right ordering. H. Samet reminds in

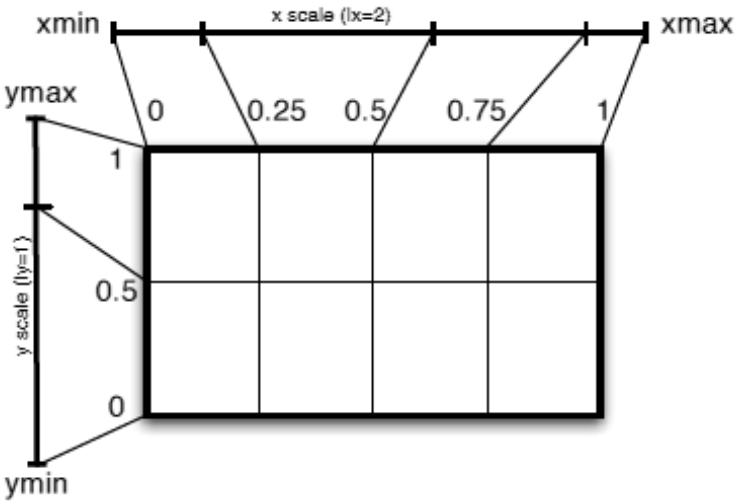


Figure 3.4: Grid partitioning of the data space by means of regular decompositions following dimensional scales.

the introduction of his book [91] that "it should be clear that finding an ordering for the range of values of an attribute is not an issue; the only issue is what ordering to use!" Then, any background consideration should be made, such like reasoning from domain taxonomies, to help finding the right ordering for categorical values.
The second strong requirement of the user-defined mapping to $[0,1]^n$ is to provide a partitioning of domain D_i within 2^{l_i} ranges, $l_i \geq 1$. The l_i parameter set up the resolution of the dimensional scale that will be used for space decomposition into the BANG file. Similarly, the unit interval $[0,1]$ is partitioned on the ith dimension into equal-sized ranges $[\frac{k}{2^{l_i}}, \frac{k+1}{2^{l_i}}]$, $0 \leq k \leq 2^{l_i}$ that map to the 2^{l_i} ranges from D_i. Since there is no constraint on the mapping from $\Pi_{1 \leq i \leq n} D_i$ to $[0,1]^n$, background knowledge could be incorporated into scales in order for instance to fix undesirable data distribution or to emphasis portions of the data space in the transformation.

Grid partitioning and resolution

The n scales define a grid on the multi-dimensional data space as shown by Figure 3.4 for a 2-dimensional space. Furthermore, as many grid-based structures, the BANG file divides the data space within a hierarchy of regions where the leaves are the finest grained grid regions corresponding to the grid resolution. The BANG file performs iterative binary partitioning to develop the hierarchy from the entire data space (root) to the grid regions (leaves). Scales are used as partition lines for regular decomposition and the process is *cyclic through the dimensions.*

Each region in the hierarchy, including grid regions, is identified by a unique pair

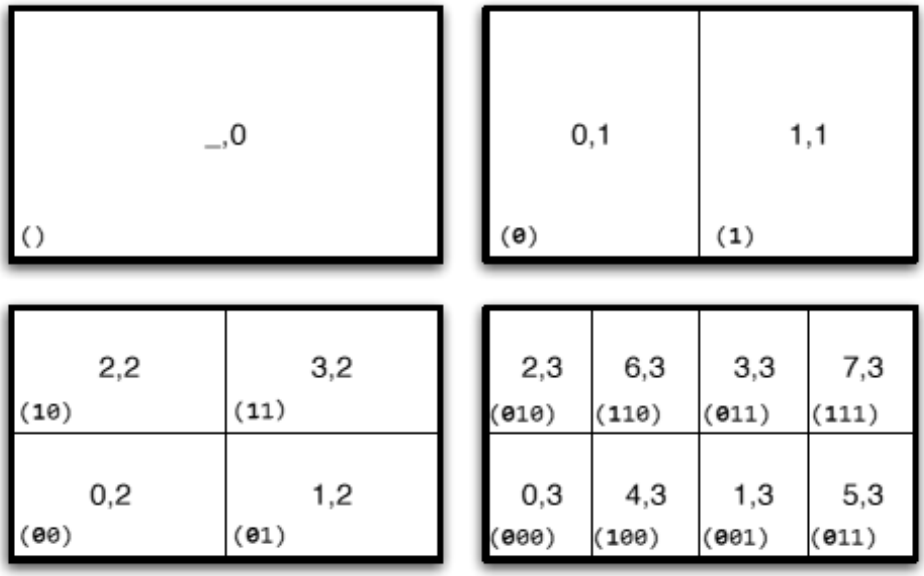


Figure 3.5: Block region numbering scheme.

(r, ℓ) where r is the region number and ℓ is the granularity or level number. A key feature of the BANG file [40] is the region numbering scheme. It relies on a *bitstring* representation of regions that provides very efficient search facilities.
Figure 3.5 shows the numbering scheme. The outermost region is not given any identifier, whereas each non-root block region is identified by (r, ℓ) with r being the value of a string of binary digits, e.g. **010** is assigned to region $(2, 3)$. Each subspace of a binary partitioning is given value 0 (left/below part) or 1 (right/above part) and regions are identified by the sequence of values of binary partitioning.

About shape

The BANG file relies on 2 axioms stated by Freeston [40]:

1. The union of all sub-spaces into which the data space has been partitioned must span the data space.
2. If two sub-spaces into which the data space has been partitioned intersect, then one of these sub-spaces completely encloses the other.

The second axiom states the existence of *nested regions* in the data space partitioning. Hence, the BANG file removes the requirement that the portions resulting from decomposition of the underlying space that are spanned by a region be hyper-rectangles. The consequence is that the sub-space spanned by a bucket of point data, so-called *a block*, is a combination of an enclosing region minus a set of enclosed regions. Then it could be either an hyper-rectangular region, or an axis-parallel concave portion of the space or

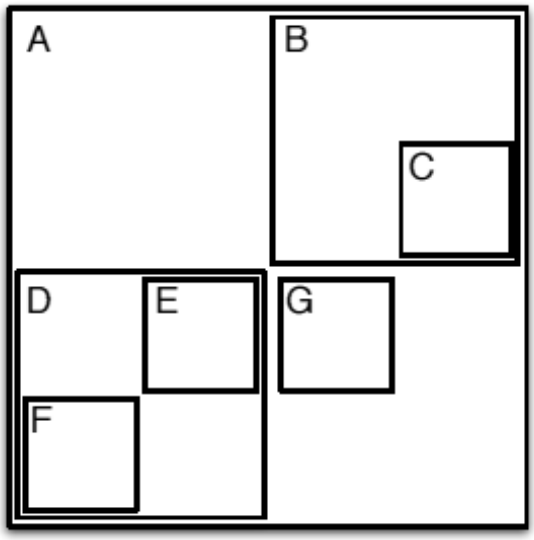


Figure 3.6: Example of a 2D data space partitioned with the BANG le into 7 nested block regions A, B, C, D, E, F, G.

even a disjoint set of sub-blocks. In the following, we denote by X an hyper-rectangular region of the space given by the regular decomposition of the BANG file. The block enclosed into X is itself denoted by $[X]$. For instance, on Figure 3.6, the block $[A]$ is assigned to region A and is defined as the sub-space spanned by region A minus regions B, D and G. Only the innermost regions such like C, E, F and G on Figure 3.6 coincide with their corresponding blocks $[C]$, $[E]$, $[F]$ and $[G]$ respectively. Otherwise, the general definition of a block is as follows:

$$[X] = X - \bigcup_{Y \in \mathcal{H}_X} Y \tag{3.3}$$

where $\mathcal{H}_X$ is the set of pairwise disjoint X-enclosed regions at the first level.
For the sake of simplicity, we also use $[X]$ to denote the data bucket from which points lie into the sub-space spanned by block $[X]$. Thus, $[X]$ is a subset of points and a complex shape as well, depending on the contextual meaning and without any ambiguity. The advantages of the block definition compared to hyper-rectangle-based kd-B-tree structures are a better observation of inherent clusters into data and also a higher filling rate of buckets. Building algorithm as described by Freeston [40] guarantees the balance among buckets by redistribution thereby making way for clustered value sets.

3.5.2 Mapping Scheme

Both insertion and searching into the BANG directory require to map data point coordinate to a block where the point lies by the way of the enclosing region number. To

this end, the BANG file defines a set of hash functions [40] from data point coordinate $\langle x_i \rangle_{1 \leq i \leq n}$ to region number r at scaling level k_i, by means of enclosing region coordinate $\langle d_i^{k_i} \rangle_{1 \leq i \leq n}$

$$d_i^{k_i} = \frac{\lfloor 2^{l_i}.x_i \rfloor}{2^{l_i - k_i}},\ 0 \leq k_i \leq l_i \tag{3.4}$$

Then, the convenient bitstring representation of region numbers allows to concatenate dimensional $d_i^{k_i}$ coordinates at levels k_i to one single $(r, \ell = \sum_i k_i)$ value. Let's take the following example:

$$d_1^2 = (10)_2$$
$$d_2^4 = (0110)_2$$
$$d_3^3 = (110)_2$$

$$r = (101|011|10|0)_2$$

This very efficient mapping is valid if regular decomposition of the space is cyclic through the set of dimensions. Offsets correspond to different dimensional scales depending on each attribute domain.

3.5.3 BANG directory

Despite historical proximity with the Grid file and its DYOP variant, directory of the BANG file is a tree rather than an array (grid). It follows H. Samet's claim [91] who states that the BANG file is a variant of the kd-B-trie, that is a kd-B-tree with regular decomposition. Figure 3.7 shows an example of a BANG tree directory from partitioning of Figure 3.6. Blocks are in the leaves whereas inner nodes contain entries of the form (subspace spanned by a child node, reference to child node). The subspace spanned by a child node is defined as an outermost hyper-rectangle and zero or more nested regions to remove. On Figure 3.7, we denote by X a simple hyper-rectangle (region) and $X!$ a complex shape built as follows:

$$X! = X - \bigcup Y \tag{3.5}$$

Where Y is an X-nested region that occurs in the path from the root to the current node. For instance, the root of the BANG tree directory from Figure 3.7 contains 2 entries: D and $A!$. D is the sub-space spanned by the left child node whereas $A! = A - D$ is the sup-space spanned by the right child node. Note that the second $A!$ of the tree is defined as follows:

$$A! = A - (B \cup D) \tag{3.6}$$

The partitioning algorithm is simple yet efficient. As any $kd-B$-tree, its time complexity is $O(NlogN)$. For external data, I/O cost still remains in $O(\frac{N}{B}.log\frac{N}{B})$ with B the disk block size. It performs incremental insertion of data points in a top-down manner. Enclosing grid region identifier is first computed thanks to the mapping scheme discussed before. The all path up to the root (the entire space) is also retrieved. Then,

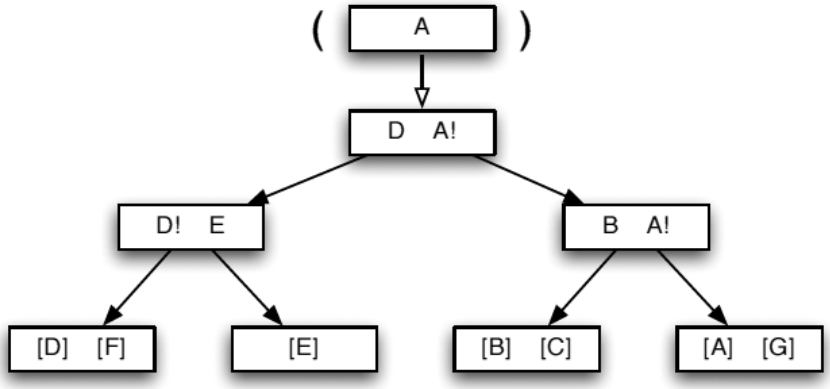


Figure 3.7: BANG directory of partitioning from Figure 3.6 represented as a kd-B-trie with fanout $B = 2$.

the BANG directory is searched for the smallest recorded region that encloses the data point. It is then assigned to the corresponding bucket.
When data bucket overflows, the algorithm operates splitting to balance the distribution of points between buckets. Splitting is done by iteratively halving the space spanned by points in the bucket until the best balance is achieved. It gives birth either to a buddy region or to a new enclosed region. The iterative halving strongly differs from the R^+-tree splitting strategy. Indeed, the BANG file operates from the entire space to blocks (topdown) whereas the R^+-tree operates from points to blocks (bottom-up). This distinct feature will be discussed further in the Performance section (see Section 3.7.2).
Finally, in order first, to encompass anonymity requirements and second, to obtain the highest quality partitioning for the anonymization process, we set up the minimum filling rate to the lowest page size (M) value, depending on the grid resolution. Then, splitting is performed whenever the bucket size reaches $2M + 1$. The second parameter B (the fanout of the tree directory) is set to the page size such that it allows to build an external tree structure that scales up to potentially any size of data sets.

3.6 From BANG Directory to Anonymous Public Release

Remind that we chose to achieve k-anonymity, based on BangA generalization algorithm, this optional post-processing step merges buckets from the BANG file to build equivalence classes of the anonymous release with the desired parameter $k \leq M$. Although the brute BANG directory already achieves very high quality in the public release thanks to non hyper-rectangular blocks, this additional processing increases the usefulness of the anonymous public release for higher k values. Actually, it could be performed on any other axis-parallel partitioning for anonymization and can be performed independently to any PAM algorithm, such like the R^+-tree approach.

3.6.1 Density-based clustering

BangA performs a density-based clustering on data buckets in a way similar to BANG-clustering [93]. BANG-clustering is a grid clustering approach that relies on a main memory kd-tree accommodated from the BANG file. It is a direct descendant from GRIDCLUS [92] based itself on the Grid file.
The algorithm computes density index for each block assigned to a data bucket and it creates dendogram by merging neighbor blocks having closest density indices. However, BANG-clustering as well as GRIDCLUS compute density indices for each block $[X]$ by means of $card([X])$ the number of data points it contains, and $V(X)$, the spatial volume of the enclosing region only. This approach does not leverage the non hyper-rectangular blocks built by the BANG file. Thus, we refine the previous proposal and define the spatial volume $V([X])$ of a BANG directory entry $[X]$ as follows:

$$V([X]) = \prod_{1 \leq i \leq n} e_X^i - \sum_{Y \in \mathcal{H}_X} \prod_{1 \leq i \leq n} e_Y^i \tag{3.7}$$

where $X - \bigcup_{\mathcal{H}_X} Y$ is the block where lies the data points from $[X]$, and X and elements of $\mathcal{H}_X$ are hyper-rectangular regions. The e_α^i are extents of the region α on the ith dimension.
The density index $\mathcal{D}([X])$ of block $[X]$ is then given by the ratio:

$$\mathcal{D}([X]) = \frac{card([X])}{V([X])} \tag{3.8}$$

Next, the algorithm performs a sort on blocks according to their decreasing density. The ranking of blocks supports the construction of a *dendogram* obtained by merging iteratively pairs of *neighbor* blocks with the highest density indices, creating new clusters otherwise. Neighborhood is defined as a shared $(n-1)$-dimensional hyperplane between two block regions. The algorithm is detailed in [92].

3.6.2 Multi-granular anonymity

BangA allows multi-granular anonymity in a single run. However, instead of working directly on the index structure, we leverage the above *dendogram* by means of computation of a cut. The main purpose is to allow the end-user to set the k value on the fly, without the need for scanning raw data. For the basic $k = M$ setting, leaves of the dendogram are straightforwardly the equivalence classes for the anonymous public release thanks to the filling requirements on the BANG file.
If we consider higher k values, then we could perform a top-down depth- first traversal of the dendogram until we reach maximally specialized k- filled blocks in each and every branch. The result cut draws the anonymous public release. Since we compute the cut on the dendogram rather than on the index structure (kd-B-trie), then the k value is not restricted to M^ℓ settings. Indeed, the approach gives BangA the ability to perform cM-anonymity, for any natural number c. In R^+-tree based approach [55], this feature is offered thanks to an ordered leaves scan of the kd-B-tree that gives low quality releases compared to BangA since adjacent leaves could be merged even if they belong to very different branches of the tree which bottom line for any clustering technique.

3.6.3 Point and Range Queries

Section 3.2.1 states that one of the PPDP requirements is to provide user-friendly descriptions of anonymous data set to ease point and range queries in very simple but popular environments such like spreadsheets. Fortunately, BangA was tailored to fulfill such requirement, without disregarding quality and efficiency of the anonymization process.
To this end, each equivalence class of the public release is encoded by its enclosing hyper-rectangular region such that nested regions are allowed in the table. And the level of each region is provided in an additional column. Hence, it becomes very easy to process point queries in the anonymous table:

1. define filters on each dimension;
2. rank the intermediate result on decreasing region level;
3. keep only the records with the lowest value on the region level.

The above procedure works since the intermediate result returns nested regions only, where the innermost region is the right answer. Then, comparing levels suffices to remove false positives that are enclosing regions. For instance, a point query in block E on Figure 3.6 returns the intermediate result set $(A, 0), (D, 2), (E, 4)$. Then, since levels are 0, 2, 4 resp. for A, D and E, the remaining block is E and the answer of point query is the set of records from $[E]$. Orthogonal range searching is slightly more difficult to manage. Indeed, if we follow the above point query processing, defining range filters rather than exact match filters, then we are left with *false negatives* since

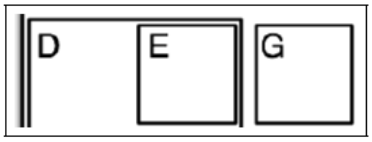


Figure 3.8: Example of a sub-space spanned by a range query on the partitioning of Figure 3.6.

enclosing regions could be partly covered by the range query. At the contrary, if we stop at step 2, then there could be *false positives* in the answer set.
Then, we propose the following methodology to manually perform orthogonal range searching in anonymous public releases. The query is first decomposed into elementary range queries that cover the entire query space with small cuboids that correspond to the finest resolution of regions in the public release. The resolution can be determined by means of the highest level value. Obviously, the resolution depends on the k value for a given public release. Then, each elementary range query is performed in the same way than point queries, except that filters on dimensions are ranges rather than exact matching. Finally, the answer set is the union of all the elementary range query results.
For instance, assume a range query Q that spans the sub-space of A shown on Figure 3.8. Step 2 of point queries with range filters returns the intermediate result set (A, 0), (D, 2), (E, 4), (G, 4) whereas step 3 gives (E, 4), (G, 4). In the former result set, (A, 0) is a false positive, and in the later result set, (D, 2) is a false negative. To fix this wrong behavior, the above methodology for range searching first decomposes the query into 3 elementary queries Q_1, Q_2 and Q_3 that span respectively the sub-part of D, region E and region G. Values of dimensional filters are given by the examination of bounds in each column of the equivalence classes. Next, Q_1 is computed as a point query (with range filters) and gives the intermediate result set (A, 0), (D, 2). Then the answer is $[D]$. The process is repeated for Q_2 and Q_3 and it returns resp. $[E]$ and $[G]$. Union of the 3 result sets is the answer to Q.
Obviously, the BANG directory tree remains available for very large data sets and could be used as a regular database access method for any kind of range queries over the leaves of the index structure (the M-anonymous release). Any other Spatial Access Method could also be considered to deal with the cM-anonymous releases built from the dendogram. Axis-parallel polytopes, that are blocks, are then indexed and they could be efficiently retrieved by usual spatial database operators.

3.6.4 BangA and other Syntactic Generalization Models

Though we employed BangA to achieve k-anonymity generalization model, it can be directly exposed to any sophisticated syntactic generalization model e.g., ℓ- diversity [75] and t-closeness [70]. As its R^+-tree counterpart, BangA would be able to

incorporate constraints from the definition of the various existing generalization models in its anonymization process. The only accommodation would be to redefine the assignment and splitting strategies such that both resulting blocks satisfy the generalization model. For instance, to make the anonymous release ℓ-diverse, it requires that at least ℓ sensitive values are "well represented" in each equivalence class. Thus, BangA would incorporate checking on sensitive values in its splitting decision to only create new ℓ-diverse blocks from old ones. And it would add constraint on assignment of a new point into an existing block such that the resulting block still satisfies the ℓ-diversity, otherwise the algorithm would locally redistribute points into blocks.

3.7 Experimental Validation

This section provides an extensive experimentation on BangA generalization algorithm to achieve k-anonymous public release. The choice of k-anonymity using BangA is due to recent striking works on combining k-anonymity and differential privacy (See Section 3.1).

3.7.1 Preparation and Settings

We conducted experiments on two datasets for the empirical evaluation of BangA. Efficiency and effectiveness was addressed according to time cost of the computation and quality of the public release, respectively. We also implemented R^+-tree approach for k-anonymization espoused in [55] for comparison with BangA, since it is commonly admitted that it is the reference algorithm. Though we had no access to the source code, we implemented the closest possible solution for the given approach strictly observing the requirements in [55] and adopted the same architecture than BangA for the sake of equity in the comparison analysis.
We used the popular "Adults" dataset taken from U.C. Irvine Machine Learning Repository. This dataset, also known as "Census Income" dataset, contains the data about individuals in the USA. We purged all records with missing values and were left with a table containing 1 million tuples. We used the attributes *Age, Zipcode and Education level* as quasi-identifiers. Second dataset was "Voter list" taken as is from the experiments conducted by Sweeney [95] in her seminal work on k-anonymization. It contains 54,803 records (tuples with missing values are already removed). We used *Age, Zipcode and Salary* as quasi-identifiers. For stress testing and to study the behavior of both the approaches for high dimensional data, we used third dataset named "Customer" which was synthetically generated using a data generator tool [1]. This dataset contains 1 million tuples with 15 attributes and all of them are used as quasi-identifiers.

1. Datanamic data generator: http://www.datanamic.com/datagenerator/index.html

Category	Description
Compiler	Microsoft Visual C++ 2005
Database	Postgre SQL
Operating System	Windows 7
CPU	Intel Xeon CPU W3520 2.67 Ghz
Memory	4096MB
Hard disk	500GB

Table 3.3: Experimental setup

Dataset	*Size*	*Quasi-identifiers*	*R^+-tree*	*BangA*
Voter list	*54,803*	*3*	*360s*	*300s*
Adults	*1 million*	*3*	*1554s*	*1116s*
Customer	*1 million*	*15*	*Out of memory*	*2765s*
Customer	*1 million*	*7*	*1314s*	*906s*

Table 3.4: Time cost (in seconds) of BangA and R^+-tree with B = 5 and M = 5.

To conduct the experiments in real database environment, we first populated a PostgreSQL database with all the three datasets. And for convenience and code efficiency, data has been normalized (see Section 3.5.1) on the database level using advanced query facilities provided by PostgreSQL DBMS. We also used database statistics for query optimization. We applied R^+-tree and BangA approaches on all the quasi-identifiers of Adults, Voter list and Customer datasets. In the R^+-tree approach, the anonymization process is followed as in [55]. Table 3.3 gives a short description of the system configuration used in all the experiments.

3.7.2 Performance

Execution time of R^+-tree and BangA was measured on Voter list, Adults and Customer datasets, thus, from 50K to 1 Million records. Block size and page size was set to 5, in order to evaluate the performance of each algorithm under stress of very small bucket and fanout values. Many runs with different settings have been done and results always confirm those presented here. In this experiment, we evaluated the BangA algorithm with *dendogram* construction, against the brute R^+-tree construction, i.e. w/o leaf scan or cut extraction. Results are presented in Table 3.4. It shows a 17% lower time cost in favor of BangA compared to R^+-tree for the Voter list data set. And the difference still increases for large data sets like Adults since it spends up to 30% less execution time for the public release computation. For high dimensional data in Customer dataset with 15 quasi-identifiers, BangA simply outperforms R^+-tree based anonymization as the later is unable to cope with such high dimensional data. In order to make a veri-

fiable comparison of both approaches, a set of 1 million tuples with 7 quasi-identifier attributes was randomly sampled from the Customer dataset (having 1 million tuples and 15 quasi-identifier attributes). With slightly large number of quasi-identifiers, results in Table 3.4 indicate 32% lower time cost in favor of BangA compared to R^+-tree. This shows that whatever the number of dimensions, BangA out performs its counterpart.
We did not compare the efficiency of BangA with previous proposal such as Mondrian [66], since experiments in [55] have shown that $R+$-tree anonymization outperforms all the previous algorithms. Thus, those experiments validate the very good behavior of BangA regarding performance and scalability. Moreover, the second storage structure of the BANG file guarantee it could handle very large data sets without any drop in the performance.
The second result is as follows: we could argue that bottom-up spatial indexing is not systematically more efficient than top-down approach as conjectured in [55]. This result is given by our own experiments comparing in the same running environment R^+-tree approach (bottom-up) with BangA (top-down). Following usual analysis on spatial access methods, we claim that the performance is mainly dependent from the splitting strategy. In BangA, we use regular decomposition following the grid whereas the original R^+-tree grows by means of a quadratic procedure comparing pairwise distances of elements in an overflowed bucket. Those strategies determine a constant factor (w.r.t. N) in time complexity that makes the execution time slower for the R^+-tree as shown on Table 3.4.
We also empirically evaluated the influence of input parameters on the process. We compared *fine-grained* and *coarse-grained* block sizes both for the R^+-tree and BangA. The results indicate that varying M parameter, for a given output k value, does not affect the quality of data but it reduces the execution time of both algorithms as there would be less partitions in both cases. For instance, to build a 100-anonymous release, it is fast and safe to set M = 100 and to build the public release as the set of leaves of the tree directory. However, in this case the construction of any k-anonymous release, $k < 100$, requires a new run. Thus, for a generic anonymization process, it is much better to set M to a quite small value and next to perform a cut in the dendorgram given an online k parameter.

3.7.3 Quality of the Public Release

Since the k-anonymity problem relies on the trade-off between privacy of individuals and utility of the public release, we computed and compared quality of the public releases built respectively with BangA and with the R^+-tree, by means of several measures of information loss. The main idea is to evaluate the extent to which the dataset has been distorted when generalizing records. We adopted generic quality measures, i.e. measures that do depend neither on the application domain nor on a specific us-

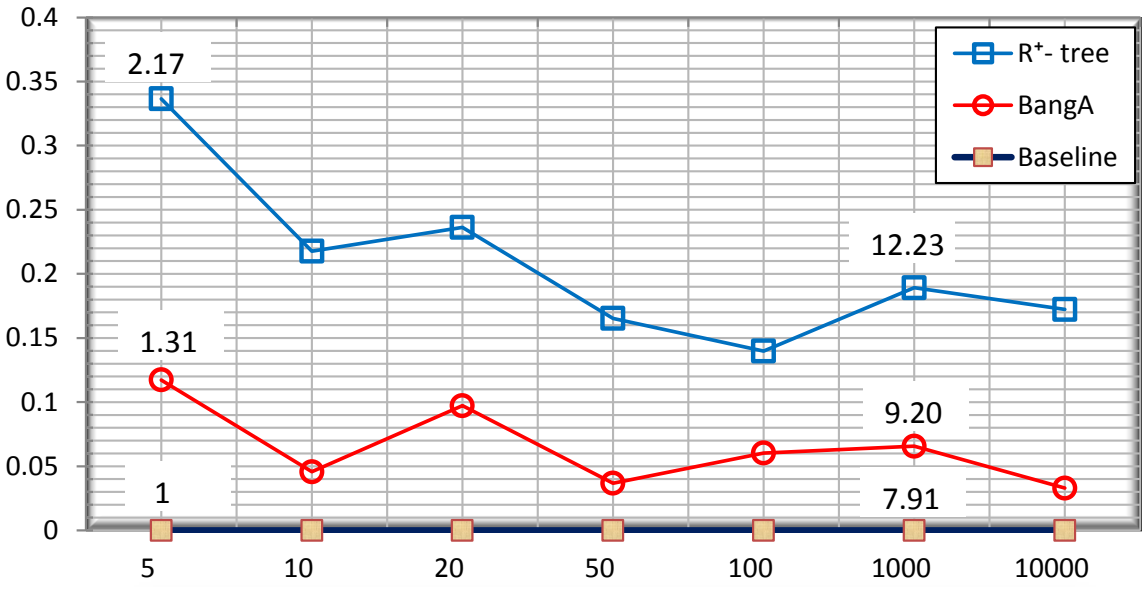


Figure 3.9: KL-divergence (on Y-axis, normalized by log ratio on baseline) according to k parameter (X-axis) in 5, 10, 20, 50, 100, 1000, 10000.

age of the public release. We then first followed the experimental protocol described by Iwuchukwu et al. [55], with 3 different measures: the Discernibility Penalty, KL-*divergence* and the *Certainty Metric* (See Section 2.5). We conducted experiments on the Adults and Customer datasets. Results for Adults dataset are presented on Figure 3.10, Figure 3.11 and Figure 3.9 for certainty metric, discernibility penalty and KL-divergence respectively. Roughly speaking, all the experiments show that BangA provides higher quality public releases than the R^+-tree since BangA curves systematically remain lower than those from the R^+-tree and quality measures are actually "penalty" measures.

Next, curves are all increasing since the higher the k parameter, the lower the overall quality. We could notice that the gap between the R^+-tree and BangA increases with k in the discernibility penalty. Here we face the usefulness of the density-based clustering of BangA since merging elementary blocks give birth to very accurate equivalence classes even for higher k values, compared to the R^+-tree. To focus on specific values rather than analysis trends in large scale curves, we consider numbers for $k = 100$ since it represents a descent rate of 0.01% of the size of the data set. Here, we observe 5% better quality in CM, 8% in DP an 9% in KL-divergence always in favor of BangA. For instance, in Figures 3.9 and 3.11, we normalized the values respectively for KL-divergence and DCP for R^+-tree and BangA w.r.t. baseline values (original values are marked for $k = 5, 1000$ in Figure 3.9 and for $k = 5$ in Figure 3.11) in order to highlight the gain achieved by BangA over R^+-tree based anonymization. Those val-

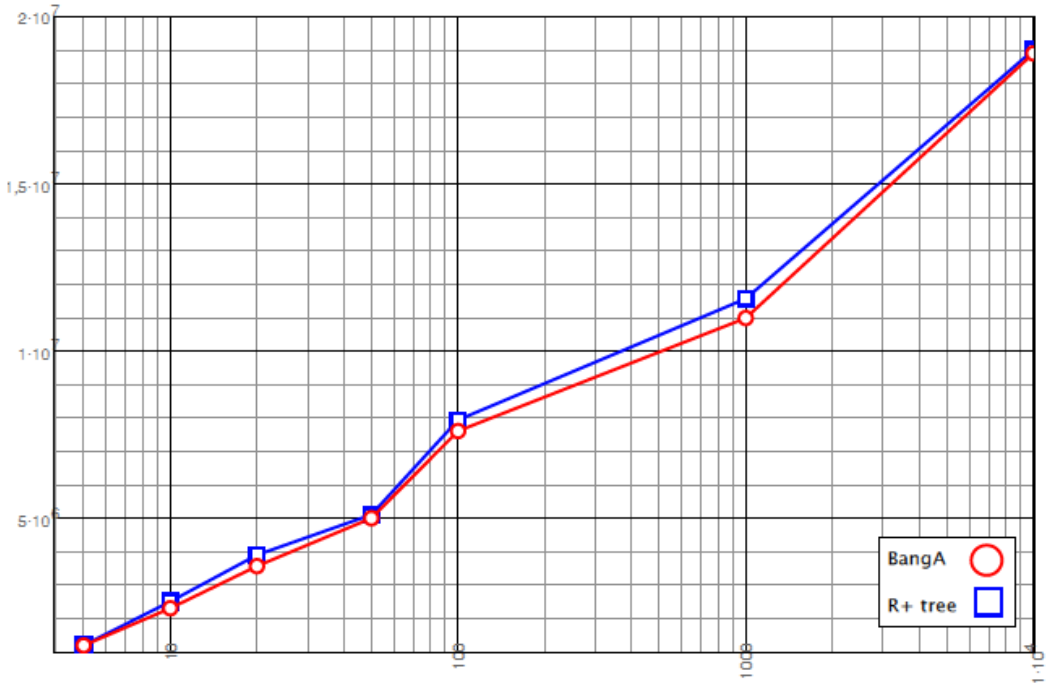


Figure 3.10: Certainty penalty (Y-axis) according to k parameter (X-axis) in 5, 10, 20, 50, 100, 1000, 10000 on a log-linear scale.

ues are prototypical of the average gap between BangA and R^+-tree with a varying k value. Moreover, if we consider the baseline of *DP*, then the improvement of BangA with respect to the R^+-tree is more than 48%. Finally, it is worth to notice that *CM* is not designed to take into account non hyper-rectangular blocks since it aggregates dimensional range values. Thus, we only computed estimated values based on enclosing regions for BangA.

3.7.4 Query Accuracy

Apart from studying the quality of data through "penalty" measures and KL-divergence metrics, the utility of the anonymized data is also studied in terms of *relative query error*. In this section, we focus on *point and window queries* as they are important building blocks for statistical analysis and many data mining applications (e.g., association rule mining and decision trees). We used the randomly sampled Customer dataset containing 1 million tuples and 7 quasi-identifier attributes for these experiments and followed the procedure detailed in Section 2.5.2.
The point queries are relatively easier to handle (See Section 2.5.2 for details). The window queries are of the form:

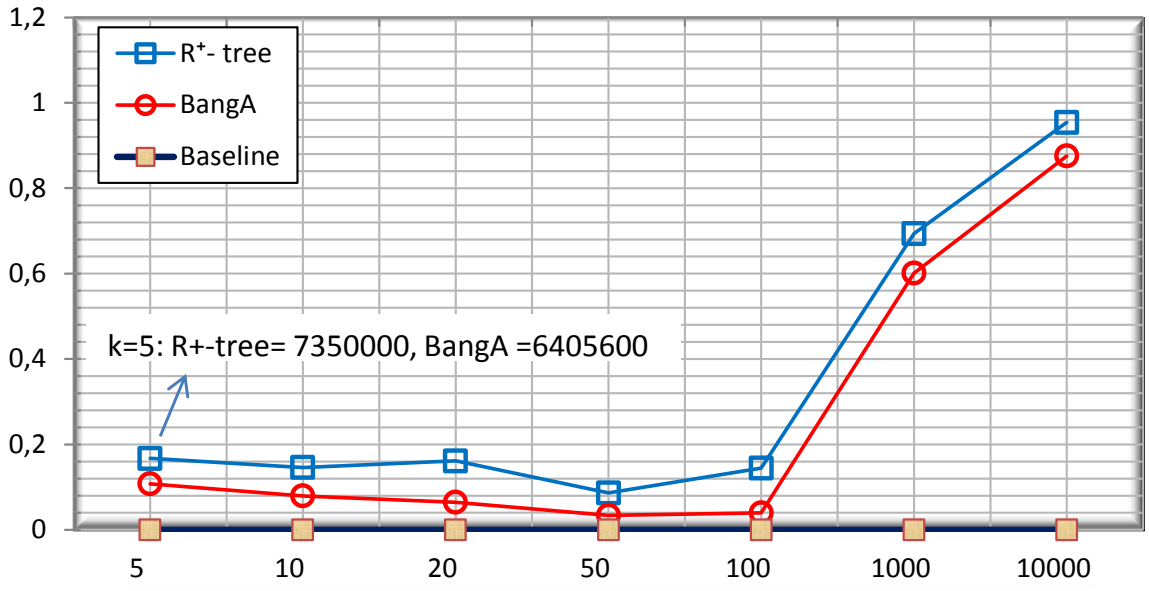


Figure 3.11: Discernability penalty (on Y-axis, normalized by log ratio on baseline) according to k parameter (X-axis) in 5, 10, 20, 50, 100, 1000, 10000.

SELECT COUNT$(*)$ **from** R
WHERE $R.QI_1 \geq qi_1$ **AND** $R.QI_1 \leq qi_2$
AND
. . .
AND
$R.QI_7 \geq qi_7$ **AND** $R.QI_7 \leq qi_7$

The above mentioned 7-dimensional query is dynamically created by using the upper and lower bounds on the range of each participating attribute. These bounds are defined as follows:
A *COUNT* query Q on the anonymized data set R^* fetches the count of tuples matching the query Q. For point query, the result set contains those tuples with the lowest value on the region level (See Section 3.6.3).
A window query Q returns a count of the records in R^* that matches Q. A tuple $t \in R^*$ is said to be a matching tuple for Q if region spanned by t and the query Q have a non-null intersection i.e., t must intersect Q on all quasi-identifier attributes.
We conducted the experiments using query dimensionality parameter (See Section 2.5.2). The query error rate is calculated using Equation (2.9). We considered 300 randomly generated queries for conducting these experiments and calculated the average relative error.
For these experiments, we anonymized the Customer dataset on all 7 quasi-identifiers and varied the query dimensionality parameter i.e., the number of QI attributes in query

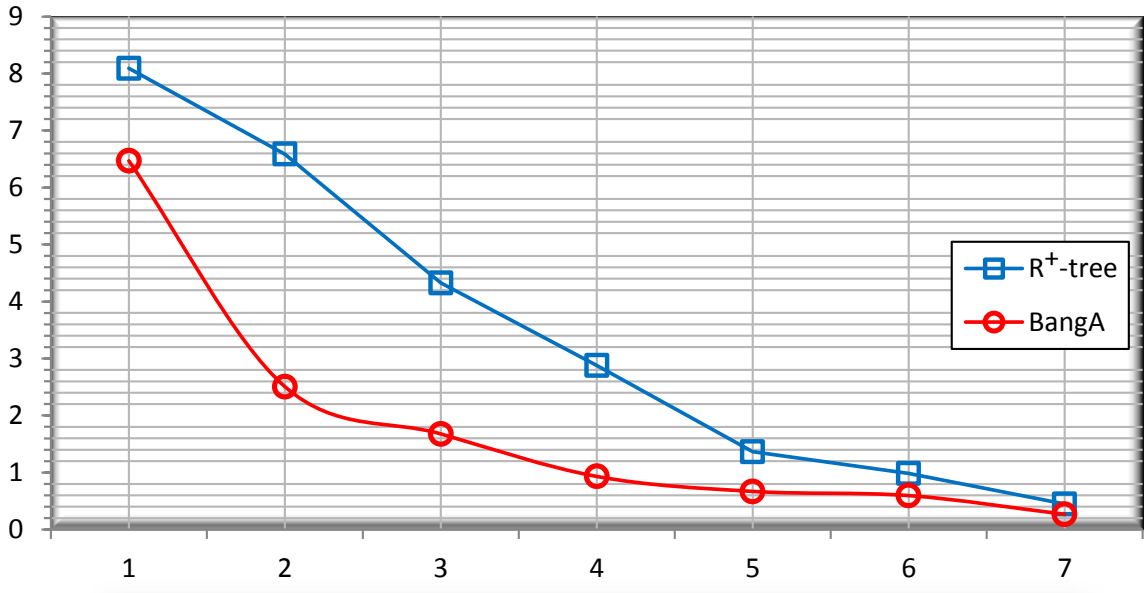


Figure 3.12: Error (Y-axis) for point queries according to varying query dimensionality (X-axis).

predicate. The results for point and window queries with varying query dimensionality are shown on Figures 3.12 and 3.13. As the query dimension increases, average relative error rate decreases. Thus the anonymized data performs better for queries with a larger query dimensions. BangA tends to be more stable than R^+ based approach showing less relative error rate for any query dimension.

3.8 Extensions

BangA generalization algorithm has shown to achieve significant gain both in terms of efficiency and utility. Along with the features mentioned above, BangA can be extended in various directions. Below we highlight few important extensions that are applicable to BangA.

3.8.1 Compaction Procedure

Iwuchukwu et al. [55] propose a compaction procedure that simply shrinks the envelop of each block to its MBB as shown on Figure 3.2. The R^+-tree approach natively computes such MBBs for every block. Consequently, the average volume of the blocks is minimized. However, BangA operates a top-down decomposition of the space such that the union of all the blocks spans the entire space. Obviously, a compaction of each

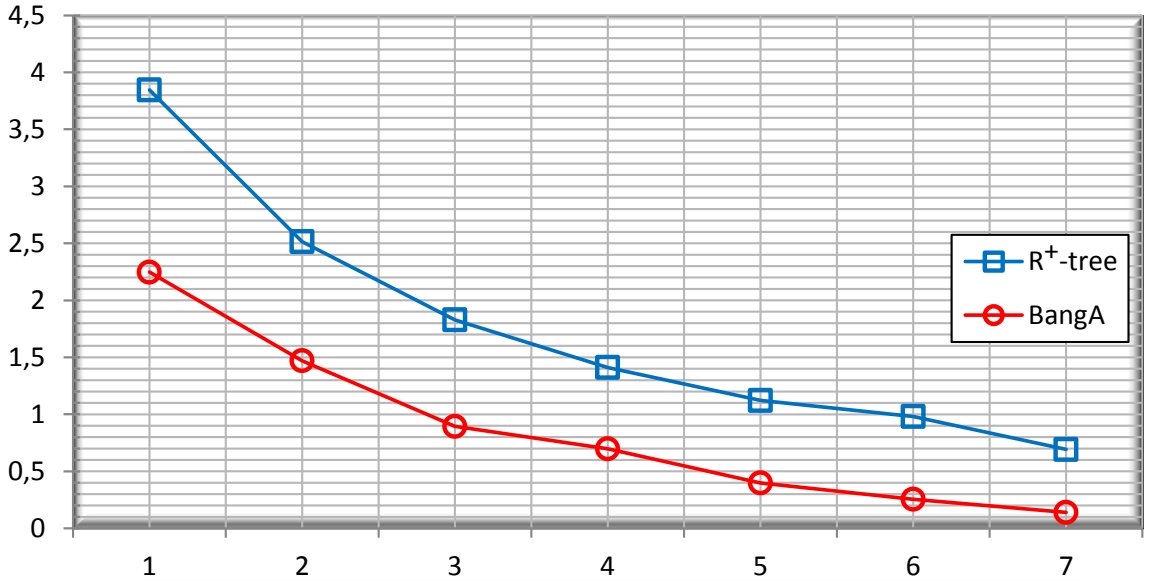


Figure 3.13: Error (Y-axis) for window queries according to varying query dimensionality (X-axis) with dimensions = 7.

block would yield to a more accurate anonymous public release, and would still increase its quality w.r.t the R^+-tree numbers. Thus, it can be considered as a straightforward improvement of BangA, even if computation of non hyper-rectangular "MBB" such like those on Figure 3.3 must be carefully defined.

3.8.2 BangA and Differential Privacy

As described in Section 3.6.4, BangA can be directly applied to any syntactic generalization model. Quite recently, *Differential Privacy* DP has emerged as a state-of-the-art semantic privacy paradigm that offers strong theoretical privacy guarantees. Due to its inability of achieving practical implementation, there is a surge of works nowadays that tend to combine the practicalness of syntactic approaches with the effectiveness of DP (See Section 2.7.2 for details).
BangA can be extended in following directions to achieve DP style privacy:

BangA and Crowd Blending Privacy

In Section 2.7, we highlighted several relaxations of DP. Specifically, Gehrke et al. [46] proposed crowd-blending privacy to strictly relax the notion of DP. They emphasize that if generalization is done safely then any generalization based algorithm may be extended to achieve crowd blending privacy. As shown by the experiments,

BangA is an extremely efficient generalization algorithm and remains suitable candidate to achieve crowd blending privacy.

BangA and Differential Privacy through RPS framework

Very recently, Qardaji et al. [86] in an extended abstract, propose multi-dimensional partitioning to achieve DP. Specifically, the authors in [86] propose a framework coined *RPS (Recursive Partitioning and Summarization)* to achieve DP. In RPS framework, the tuples in micro-data are treated as points in multidimensional space. To achieve DP via multi-dimensional partitioning, an RPS algorithm specifies three subroutines:

1. how a region can be partitioned
2. when to stop partitioning
3. how to summarize the tuples in partition

For an RPS framework to be differentially private, all the three subroutines must follow some form of DP. Specifically, the authors propose a multidimensional partitioning based k-anonymity solution that satisfies DP via the RPS framework mentioned above. Since BangA employ extremely efficient multidimensional partitioning, it is an interesting candidate to be a part of RPS framework.

3.8.3 BangA and Incremental Data Anonymization

The data sanitization based on k-anonymity model has been extensively studied for the past few decades. However this intensive research on k-anonymity is limited to the scenario where it is assumed that the entire dataset is available at the time of release. In other words, much of the work done on k-anonymity model focus on static data. This assumption leads to severe shortcomings both in terms of utility and privacy as data nowadays are continuously collected (thus continuously growing) and there is increasing demand for up-to-date data frequently. Previous k-anonymization techniques can be employed on a dataset as a whole i.e., they take a raw dataset as input and output the anonymized version. If new records are added to the dataset, the only solution is to anonymize the whole dataset including the new records.
Since the spatial indexes are designed for frequent updates, BangA can easily be employed without any modification to previously anonymized data in this dynamic setting. New records can be added to previous equivalence classes without breaking the k-anonymity. Also the utility of resulting public release remains good as described in [55]. Remind that this approach is limited to *Insert-Only* scenario where there are no deletions and modifications in the previous version of raw data.

3.9 Synthesis

In this Chapter, we proposed a new anonymization method called BangA. Based on the BANG file indexing structure, it performs very well and provides non hyper-rectangular blocks assigned to the equivalence classes of the public release. Furthermore, BangA allows to incorporate background knowledge in the dimensional scales that are used for regular decomposition. A post-processing step provides a density-based clustering of the blocks in order to achieve a high quality anonymization regardless of the k value. And since the result of such post-processing is a dendogram, then, it offers the opportunity to build on demand the desired k-anonymous release without scanning the raw data. And to support the exploration of non hyper-rectangular blocks, we also provided a methodology for point and range searching in nested equivalence classes of the anonymous public release. Along with usual benefits, BangA can easily be extended to adopt the compaction procedure to achieve better utility of data. Also BangA can incorporate other generalization models like ℓ-diversity by making slight adjustments in its assignment and splitting strategies. Last but not least, without any loss of generality, BangA can be served as-is for incremental data anonymization. Quite recently however, Differential Privacy (DP) has received much attention from the research community and BangA generalization algorithm is an interesting candidate to achieve DP style privacy.

4

τ-safety

Summary: *BangA is a first step towards sequential data anonymization. Sequential data anonymization is obviously more complex than static publication scenario mainly due to cross-release inference channels. It deals with publication of multiple releases each containing data from previous release(s) along with new records and/or modification in the records of previous releases. Modification of previous records are either update in any of the attribute values or deletion of a record from one release to the next one. Along with these modifications, sequential data publication is prone to several kinds of adversarial attacks that are not applicable for static data publication. This makes the static publication models inappropriate for this scenario since even if each release is individually anonymous, combining multiple releases begets the situation in which privacy can be compromised. BangA is able to provide the required privacy in the scenario in which there are only new records to manage. Since record deletion or update brings about a complex problem, more sophisticated privacy models are required. Among the few works in the literature that relate to sequential data publication, none of them focuses on arbitrary updates, i.e. with any consistent insert/update/delete sequence, and especially in the presence of auxiliary knowledge that tracks updates of individuals all along the series of releases. In this Chapter, we first highlight the invalidation of existing algorithms and present an extension of the m-invariance generalization model coined τ-safety. Then we formally state the problem of privacy-preserving dataset publication of sequential releases in the presence of arbitrary updates and chainability-based background knowledge. We also propose an approximate algorithm, and we show that our approach to τ-safety, not only prevents from privacy breach but also achieve a high utility of the anonymous releases.*

Contents

4.1 Introduction

The work in Chapter 3 focuses on the problem of minimizing the risk of identifying the individual record holders in a person-specific table. A set of quasi-identifying attributes *QI* is generalized to a coarser representation such that each individual is grouped with a certain number of other individuals (e.g., in k-anonymization each equivalence class contains at least k records). In this context, the data in the person-specific table is static and is aimed for one time publication. In more complex scenarios, a data publisher needs to publish the micro-data multiple times with frequent updates i.e., new records are inserted, deleted and updated. The publication of micro-data with such frequent updates brings about several privacy scares. Previously, most of the work in privacy preserving data publication caters only static data publication. In dynamic setting however, data are modified and published multiple times. Sequential publication is obviously more challenging as it raises new kinds of attacks w.r.t. the single publication scenario.

4.1.1 Motivation

Dynamic data republication poses serious threats to the privacy of individuals regarding two kinds of updates in data sets [68]. *External updates*, intuitively, are the updates comprising of *first time insertions and deleted records* as they affect the total number of records in the resulting dataset and *Internal updates* correspond to either the modifications in each record's attribute values or re-insertion of a record. We assume that the internal updates in dynamic data sets are *arbitrary i.e.* old values may not have any correlation with the new ones. In other words, a sensitive attribute value can be internally updated to any other value within its domain. For example, if a person is admitted to a hospital for *flu*, it is not necessary that if at later time she is admitted to the hospital, she will have *flu* or other respiratory disease i.e. her new disease is not dependent on the previous one.

Suppose the hospital publishes Tables 4.1, 4.2 and 4.3 (original values in brackets) following *ℓ-diversity* principle at times 1, 2 and 3 respectively in which the attributes *Age and Zipcode* are *QI* and *Disease* a *sensitive attribute*. We further categorize internal updates as *QI updates (modifications in QI)* and *sensitive updates (modification in sensitive attribute)*. Any individual who belongs to this publication series has an (logical) *event list* associated with him/her. This event list contains the information about how the data of that individual has evolved by time. For example, an individual *p* appears for the first time in the release R_1. Then, before the publication of R_2, he contracted a new disease and R_2 reflects this change. So *p's event list* has the information that he first appeared in the dataset at time 1 and his record gets changed at time 2. This event list contains sensitive information about p and if an adversary (e.g. friend of p) owns this information, the privacy of p is at stake.

Keeping the above ideas in mind, we explain possible inferences due to these event lists and in the presence of (internal and external) updates. In Table 4.2, the records of p_2 and p_6 are internally updated (italicized), record of p_7 is first time insertion (bold) and record of p_4 is deleted. In Table 4.3, the record of p_4 is inserted again (under-bar) i.e she is hospitalized for the same disease at time 3. Identifiers of the individuals (ID in this case) are not included in the public release. They are shown here for the ease of understanding. Even though each release is individually anonymous, the privacy requirement could be compromised by the comparison of different releases by event list and discarding some possible sensitive values for a victim.

Invalidation of existing methods for static datasets

The main problem with static data approaches when they are employed in dynamic settings is that these approaches do not take into account the distribution of sensitive values in the previous public release(s). For instance, ℓ-*diversity* requires that each equivalence class contains ℓ well-represented sensitive values. Although each and every public release is 2-diverse, adversary may be able to identify an individual's sensitive value by comparing any two releases. The privacy of the individuals can be breached as shown by the following scenarios. We assume that the adversary has access to all previously published releases and knows the exact *QI* value and event list of each individual.

Scenario I: Suppose the adversary (an acquaintance of p_1) is looking for the sensitive value of p_2 in Table 4.2. By using the event list, the adversary knows that p_1's sensitive value is unchanged in both releases though she is not aware of p_1's sensitive value. The adversary can argue as follows: p_1 and p_2 must be in first equivalence class in both releases. They must have contracted *{cataract, pneumonia}* in first release and *{cataract, diarrhea}* in the second one. Since the only unchanged value is *cataract*, it is the sensitive value of p_1. Thus p_2 contracted *pneumonia* in first release and *diarrhea* in the second one.

Scenario II: Suppose the adversary is looking for the sensitive value of p_2 in Table 4.3. The adversary knows that p_2 belongs to the first equivalence class of all the published releases. p_2 must have contracted *{cataract, diarrhea, glaucoma}* at time 3 and *{cataract, diarrhea}* at time 2. Also, adversary (through event list) knows the fact that p_2's sensitive value is unchanged at time 2 and 3. By comparing these two releases, the adversary is able to *exclude glaucoma* as p_2*'s* disease at time 3. Thus the probability that p_2 has *diarrhea* at time 3, increases from $\frac{1}{3}$ to 0.5 due to an internal update. By using this knowledge and using the first published release, the adversary can further narrow down to breach the privacy of p_2.

Name	Age	Zipcode	Disease
p_1	21-22(21)	12k-14k(12k)	cataract
p_2	21-22(22)	12k-14k(14k)	pneumonia
p_3	23-24(24)	18k-25k(18k)	flu
p_4	23-24(23)	18k-25k(25k)	glaucoma
p_5	41-42(41)	20k-34k(20k)	flu
p_6	41-42(42)	20k-34k(34k)	gastritis

Table 4.1: 2-Diverse R_1^*

Name	Age	Zipcode	Disease
p_1	21-23(21)	12k-4k(12k)	cataract
p_2	*21-23(23)*	*12k-40k(40k)*	*diarrhea*
p_3	24-26(24)	18k-34k(18k)	flu
p_7	**24-26(26)**	**18k-34k(34k)**	**gastritis**
p_5	41-42(41)	20k-35k(20k)	flu
p_6	*41-42(42)*	*20k-35k(35k)*	*gastritis*

Table 4.2: 2-Diverse R_2^*

Scenario III: Suppose the adversary is looking for the sensitive value of p_4 in Table 4.3. Since the adversary knows that p_4 has records in Tables 4.1 and 4.3 and her sensitive value is unchanged in both these releases, the adversary can argue as follows: p_4 belongs to second equivalence class at time 1 and first equivalence class at time 3. She must have contracted *{flu, glaucoma}* at time 1 and *{cataract, diarrhea, glaucoma}* at time 3. Since the only unchanged value is *glaucoma*, p_4 has that disease at both times she belonged to the public release.

In the scenarios mentioned above, the information in the event list of individual is used to link two published releases in the presence of arbitrary updates. We denote the event list by τ and term such attacks as τ-*attacks*. For instance, in Scenario I, the τ-*attack* became possible when event list (τ) of p_1 is used to link the public releases at time 1 and 2. It is important to notice that without the event list, this problem can be reduced to several independent problems for static dataset because then, the arbitrary internal updates of sensitive values will lead to entirely different publications with no correlation whatsoever i.e., R_1 and R_1 are completely independent [12, 68].

Invalidation of m-invariance due to internal updates

m-invariance [112] is the seminal work in dynamic dataset republication that can only handle external updates. Briefly, the requirement of *m*-invariance is that if a record

Name	Age	Zipcode	Disease
p_1	21-23(21)	12k-4k(12k)	cataract
p_2	21-23(23)	12k-40k(40k)	diarrhea
p_4	21-23(23)	12k-40k(25k)	glaucoma
p_3	24-26(24)	18k-34k(18k)	flu
p_7	24-26(26)	18k-34k(34k)	gastritis
p_5	41-42(41)	20k-35k(20k)	flu
p_6	41-42(42)	20k-35k(35k)	gastritis

Table 4.3: 2-Diverse R_3^*

occurs in two consecutive releases then it must bear the same set of sensitive values in both releases.

As an example, p_2's disease in first release is *pneumonia*. In first release, p_2 is in the equivalence class with the set of sensitive values *{cataract, pneumonia}*. She is successfully cured and admitted to the hospital for *diarrhea*. According to m-invariance, the equivalence class of p_2 in current release must be *{cataract, pneumonia}* but due to internal update of *pneumonia* to *diarrhea*, p_2's equivalence class cannot be as in the previous release. Thus the requirement of m-invariance is not manageable. Also, m-invariance does not keep track of record's sensitive values in previous releases. Thus if a previously deleted record is re-inserted at some later point, m-invariance considers it is a new record and consequently, raises τ-attack threats.

4.1.2 Contributions

We propose an extension of m-invariance for the sequential publication of fully dynamic datasets in the presence of τ-attacks. Within our proposal, Table 4.3(a) and Table 4.3(b) are published at time 2 and Table 4.4(a) and Table 4.4(b) at time 3. Table 4.3(a) contains a generalized version for each tuple from raw micro-data and consists of four equivalence classes. Tuples with names c_1 and c_2 are the *fake tuples*(fake tuples are used to counter the problems that arise due to deletion or updation of tuples) and Table 4.4(a) contains four equivalence classes and contains only one fake tuple i.e. c_1. Tables 4.3(b) and 4.4(b) contain basic statistics that show the equivalence classes 1 and 2 at time 2 and equivalence class 1 at time 3 have fake tuples. To the best of our knowledge, it is the first work that investigate the problem of sequential data publication with arbitrary updates in the presence of *chainability*-oriented auxiliary knowledge (such knowledge enables cross-release inference channels by tracking the individuals in multiple releases). Moreover, in this domain, data utility has not been a major concern in the previous literature, whereas it is a first-class citizen in our approach.

Our main contributions are as follows:

(a) R_2^*

Name	GID	Age	Zipcode	Disease
p_1	1	21-22 (21)	12k-13k (12k)	cataract
c_1	1	21-22	12k-13k	pneumonia
p_3	2	24-25 (24)	18k-19k (18k)	flu
c_2	2	24-25	18k-19k	glaucoma
p_5	3	41-42 (41)	20k-35k (20k)	flu
p_6	3	41-42 (42)	20k-35k (35k)	gastritis
p_2	4	23-26 (23)	34k-40k (34k)	diarrhea
p_7	4	23-26 (26)	34k-40k (40k)	gastritis

(b) Counterfeits

GID	Count
1	1
2	1

Table 4.4: τ-safe 2-invariant Generalization R_2^*

1. We propose the τ-safety paradigm, defined after m-invariance, for the sequential publication of anonymous releases from dynamic dataset in the presence of arbitrary updates and under the threat of τ-attacks.
2. We shift from *record-based privacy* paradigm to *individual-based privacy* such that serial data publication mechanism becomes safer.
3. Assumptions about adversary's knowledge are severe such that she knows tracks of individual's modification. We then take care about *chainability* within the background knowledge model.
4. We designed and implemented an approximation algorithm to show by intensive experiments that τ-safety has immediate practical impact.
5. We draw a general framework for such a problem and give opportunities for future independent contributions on many open issues. For instance, the trade-off between utility and fake tuples is properly stated as well as various optimality criteria.

4.2 Problem Foundation

Let $\mathrm{T} = (R_1, R_2, \ldots, R_p)$ be a set of micro-data tables generated at times $1, 2, \ldots, p$ respectively. R_j is an instance of micro-data table at time j $(1 \leq j \leq p)$ and has the schema $\langle ID, QI, S \rangle$. Denote by $R = \bigcup_{1 \leq j \leq p} R_j$, the union of all the records t that occurs in T. Let $\mathcal{X} = \bigcup_{i=1}^{p} \pi_{ID}(R_i)$ be the set of individuals x where *ID* is the identifier of records in T. At time j, each individual x is associated with an "event list" of size j which holds the series of operations performed on x till time j. We denote by $\mu = \{\mathbf{i}_{\text{(nsert)}}, \mathbf{u}_{\text{(pdate)}}, \_{}_{\text{(unchanged)}}, \mathbf{d}_{\text{(elete)}}, \mathbf{r}_{\text{(e-insert)}}\}$ the alphabet of operations that can be performed on x. The event list for an individual x is denoted by $\tau(x)$. It is a valid sentence of the

(a) R_3^*

Name	GID	Age	Zipcode	Disease
p_1	1	21-22(21)	12k-13k(12k)	cataract
c_1	1	21-22	12k-13k	pneumonia
p_3	2	23-24(24)	18k-25k(18k)	flu
p_4	2	23-24(23)	18k-25k(25k	glaucoma
p_5	3	41-42(41)	20k-35k(20k)	flu
p_6	3	41-42(42)	20k-35k(35k)	gastritis
p_2	4	23-26(23)	34k-40k(34k)	diarrhea
p_7	4	23-26(26)	34k-40k(40k)	gastritis

(b) Counterfeits

GID	Count
1	1

Table 4.5: τ-safe 2-invariant Generalization R_3^*

Symbol	*Meaning*
x	*an individual*
$\mathcal{X}$	*Historical union of individuals*
τ	*event list*
μ	*operations list*
[t] or [x]	*QI-group*
Sig([t])	*signature of a QI-group*
Del	*delete list*
m	*parameter for m-invariance*
i, j, k, l	*time stamps*
$\tau(x)[j]$	j_{th} component of $\tau(x)$

Table 4.6: Notations

grammar defined by the following regular expression:

$$\tau := (\_*, (i, (\_ * | u)*, (d, \_*, (r, (\_|u)*, d)*, (r, (\_|u)*)?)?)?) \tag{4.1}$$

It mainly states that we cannot delete before having inserted or re-inserted, and other basic such rules. For example, $\tau(x) = (i, \_, u, d, r)$ indicates that an individual x is inserted at time 1, remains unchanged at time 2, has been updated at time 3 and deleted at time 4 and then, inserted again at time 5. $\tau(x)[j]$ denotes the j^{th} element in $\tau(x)$.

4.2.1 The Preliminaries

Notations in Table 4.6 are used throughout the Chapter. The definition of fully dynamic dataset has been evolving since [15] first proposed the idea of dynamic datasets republication. Intuitively, the data in a dynamic dataset do not remain the same in

each subsequent release. Fully dynamic dataset contains two kinds of updates namely external and internal updates:

Definition 4.1 (External Update:) *For all j, an individual x is said to be an external update in R_j iff one of the following conditions hold:*

1. *deletion:* $\tau(x)[j] = d$
2. *insertion:* $\tau(x)[j] = i$

External update correspond mainly to the insertion or deletion of records.

Definition 4.2 (Internal Update:) *$\forall j$, an individual x is said to be an internal update in R_j iff one of the following conditions hold:*

1. $\tau(x)[j] = u$
2. $\tau(x)[j] = r$

Internal updates correspond to the re-insertion or modification of QI values in any tuple. As mentioned in section 4.1.1, internal updates can be categorized in *sensitive updates* and *QI updates*. If an individual x is a sensitive internal update at time j, then we consider $\tau(x)[j] = d$ and incorporate y such that $\tau(y) = (\_, \ldots, \_, i)$. We also fix $t[ID] = x$ to $t[ID] = y$ in each of the following releases. As a consequence, lifespan of x cannot extend after time $j-1$ and lifespan of y starts from time j. We then treat sensitive internal updates as first time insertions. Indeed, when sensitive value is arbitrarily updated, then track of the individual is basically reseted. It is important to note here that the concept of re-insertion of a tuple cannot be thought of as an *(external) deletion and (external) insertion* because then it will be considered as a new tuple thereby making it vulnerable for τ-attack.
A generalized version R^* of a micro-data table R can be obtained by applying a generalization mechanism as defined in Definition 2.4 such that $\mathcal{A}(R) = R^*$. A generalized table series T^* of $\mathrm{T} = (R_1, R_2, , ..., R_p)$ is an instance of $(R_1^*, R_2^*, ..., R_p^*)$. Note that ID column is obviously discarded in public releases.

4.2.2 Adversarial Background Knowledge

At time *p*, the adversarial knowledge consists in:

- the generalized series $\mathrm{T}^* = (R_1^*, R_2^*, ..., R_p^*)$.
- the publicly available external relations ET_j, $1 \leq j \leq p$ that gives QI values for any ID value at time j , as in Table 4.7 e.g. voter list as used by Sweeney [96]. Then, the adversarial knowledge includes a series of $ET = (ET_1, ET_2, ..., ET_p)$.
- multivalued modification function τ that gives the event list $\tau(x)$ for each individual x occurring in T.

(a) ET_1

ID	Age	Zipcode
p_1	21	12k
p_2	23	14k
p_3	24	18k
p_4	23	25k
p_5	41	20k
p_6	42	34k

(b) ET_2

Name	Age	Zipcode
p_1	21	12k
p_2	23	40k
p_3	24	18k
p_4	23	25k
p_5	41	20k
p_6	42	35k
p_7	26	34k

Table 4.7: External tables

- the $\preceq$-join that makes all the possible matchings between each entry in ET_j and $\langle QI, S\rangle$ values in R_j^*.

To sum-up, the adversarial background knowledge is the quadruple:

$$\mathcal{BK} = (ET,\ \mathrm{T}^*, \tau,\ \preceq) \tag{4.2}$$

At time j, $(1 \leq j \leq p)$, adversary's knowledge is enforced by the join:

$$BK_j = (R_j^* \underset{R_j^*[QI] \preceq ET_j[QI]}{\bowtie} ET_j) \tag{4.3}$$

Adversary can further narrow down the acquired knowledge by applying several joins on the set of previous BK_j knowledge:

$$\mathcal{BK}_p = BK_1 \underset{ID,S}{\bowtie} BK_2 \underset{ID,S}{\bowtie} \dots \underset{ID,S}{\bowtie} BK_p \tag{4.4}$$

And then, an iterative join process allows to gain further knowledge up to a fix-point. Roughly, candidate set of sensitive values for any single individual is compared to the one from the previous step until there is no more reduction. We term $\mathcal{BK}$ as *chainability-oriented auxiliary knowledge* as it chains the knowledge from the previous public releases and can be used to track an individual through T.

4.2.3 Privacy Disclosure

Privacy breach occurs when an adversary is able to gain certainty about sensitive value of an individual.

Definition 4.3 (Privacy risk:) *Let T^* be a published series and $\mathcal{BK}$ the adversary's background knowledge against T^*. The privacy disclosure risk of an individual $x \in \mathcal{X}$ is given by:*

$$\textit{risk}(x) = P(x[S] \mid \mathcal{BK})$$

where $P(x[S] \mid \mathcal{BK})$ is the probability that the individual x is linked to its effective sensitive value $x[S]$, given the knowledge of an adversary $\mathcal{BK}$.

Definition 4.4 (δ-Privacy:) *Given a published series* T^* *and* $\delta = [0, 1]$*, we say that* δ*-privacy is satisfied if* $\mathrm{risk}(x) \leq \delta$ *for all individuals* $x \in \mathrm{T}^*$

δ-Privacy is the basic privacy requirement that any sequential anonymization algorithm for fully dynamic dataset must follow in order to guard the privacy of individuals. For instance, the privacy models like m-invariance and m-Distinctness follow $\frac{1}{m}$ privacy with different settings of background knowledge.

4.3 Problem Statement

4.3.1 *m*-invariance revisited

m-invariance [112] is a baseline for dynamic data re-publication with external updates only, such that lifespan of a tuple is necessarily a consecutive range of timestamps. We require first to define QI-groups and signatures before we are able to provide a definition for the m-invariance mechanism.

Definition 4.5 (QI-group:) *Given* R *an instance of a database, and* $\mathcal{A}$ *a generalization mechanism; a QI-group in* $\mathcal{A}(R)$ *is an equivalence class defined by the equivalence relation* $\sim$ *such that the quotient space* $\mathcal{A}(R)/\sim$ *provides a partition of records in* $\mathcal{A}(R)$ *and* $t \sim u \Leftrightarrow t[QI] = u[QI]$.

For any tuple t*,* $[t]$ *is the QI-group that contains* t*. By straightforward extension,* $[x]$ *is the QI-group of an individual* x *by the way of* $x = t[ID]$*. All the tuples in a QI-group share a single QI value, whereas they may have distinct S values that form the signature of a QI-group.*

Definition 4.6 (Signature:) *[112] Let* $[t]$ *be a QI-group in* R^**; the signature* $\mathrm{Sig}([t])$ *of* $[t]$ *is the set of distinct sensitive values in* $[t]$.

Definition 4.7 (Candidate Sensitive Set (CSS):) *[112] Let* $[x]$ *be a QI-group of an individual* x *in* R_j^**; for an individual* $x \in [x]$*, the candidate sensitive set of* x *at time j denoted by* $x.CSS[j]$*, is the union of sensitive values in* $[x]$.

Xiao et al. [112] proved that for an individual x having lifespan *[i, j]*, risk(x)=1 if there exist a single element in $x.CSS[i] \cap x.CSS[i+1] \cap \ldots \cap x.CSS[j]$. This is the main reason behind the failure of conventional static data publication models e.g., k-anonymity, ℓ-diversity etc. when they are employed in dynamic settings. In order to

prevent such situation, the authors of [112] enforce the constraint on each QI-group such that each republication must ensure a sufficiently large $\cap_{k=i}^{j} CSS(k)$ at each publication timestamp. They term such constraint as *persistent invariance*. The idea of persistent invariance led to the proposition of m-invariance privacy model.
The m-invariance mechanism relies on a strict generalization model for static releasing of micro-data. It has been coined m-uniqueness.

Definition 4.8 (m-unique:) *[112] A generalized table R^* is m-unique iff each QI-group in R^* contains at least m tuples, and all tuples in the group have distinct sensitive values.*

Definition 4.9 (m-invariance:) *[112] A sequence of published relations R_1^*, R_2^*, ..., R_p^* is m-invariant if the following conditions hold:*

1. *$\forall j\ (1 \leq j \leq p)$ R_j^* is m-unique.*
2. *For any tuple t with lifespan $[i, i+k](1 \leq i \leq p)$, $k \geq 0$ we have $\mathrm{Sig}([t]_i) = \mathrm{Sig}([t]_{i+1}) = \ldots = \mathrm{Sig}([t]_{i+k})$, where $[t]_j$ denotes the QI-group of t at time $j \in [i..i+k]$*

The core idea of m-invariance is to preserve the same set of candidate sensitive values for each tuple within its entire lifespan. However, this idea faces the problem of *critical absence* whenever some previous sensitive value is missing in one release. This important issue will be extensively discussed later on.

m-invariance was proved to resist republication-based attacks under the external update assumption. It is shown in the next section that m-invariance is not sufficient to prevent from τ-*attacks*.

4.3.2 τ-Attacks

In this section we present the idea of τ-attack as the most sophisticated threat in sequential releasing with arbitrary updates. The τ-attacks are closely related to the composition attacks [43] of an adversary. Consider a nosy neighbor who is able to track her friend in each public release. With every public release, she gains the information about how the data of her friend is evolved by time or she is building the event list for herself. By keeping the event list for each individual, we can keep track of the adversarial knowledge at each time. Thus, the event list is handful in thwarting such kind of adversarial knowledge.

Ganta et al. [43] identify the composition attacks in partition based schemes such that these attacks are spread over several releases and in the presence of external knowledge. Though composition attacks [43] are focused on single static anonymization techniques, the τ-attacks correspond to the same category and target multiple releases in

which an adversary can either guess the exact sensitive value of an individual as in Scenario I and III (*exact sensitive value disclosure* [43]) or the adversary can *locate* the set of sensitive values the victim may be assigned to, as in Scenario II (*locatability* [43], where locatability is a process of pruning the set of sensitive values that might not relate to the target victim). Keeping in mind the scenarios discussed in Section 4.1.1, we then define the τ-attacks in a simple manner.

Definition 4.10 (τ-attack:) *For any individual x in $\mathcal{X}$, there is a τ-attack if an adversary with $\mathcal{BK} = (ET, T^*, \tau, \preceq)$ can precisely infer $x[S]$.*

The elaborated part of the definition comes from τ in $\mathcal{BK}$ that allows for original disclosures requiring new generalization models for sequential releasing. As explained earlier, though τ-attack can be performed on many "one shot" models, this problem may get worse in dynamic scenarios. This is because, with each republication, the adversary may be able to perform τ-attack by combining multiple releases specially when they are not consecutive. Let us take an example when m-invariance is prone to τ-attack. m-invariance imposes on each tuple to keep signature unchanged from one release to the next one. Thus, if a record t is deleted at time i and then re-inserted at time $i + k$ ($k > 0$), then this constraint does not affect t. And even if t is unchanged from i to $i + k$, its signature may be different at time $i + k$, i.e., $\text{Sig}([t]_i) \neq \text{Sig}([t]_{i+k})$ such that $\frac{1}{m}$-privacy is no more guaranteed and could yield to τ-attacks.

4.3.3 τ-safety

In this section, we introduce a new paradigm for privacy preserving in dynamic data publication namely *τ-safety*.

Definition 4.11 (τ-safety:) *A sequence of anonymized releases $T^* = R_1^*, R_2^*, \ldots, R_p^*$ is said to be τ-safe iff it satisfies the following conditions:*

1. *At any time j ($1 \leq j \leq p$), R_j^* is m-unique.*
2. *For each individual x and each consecutive lifespan $[i..i + k]$ in $\tau(x)$ of any individual x, signature of $[x]$ must remain the same.*
3. *Whenever $\tau(x)[i] = r$ for an individual x, $\text{Sig}([x]_i) = \text{Sig}([x]_{i-k-1})$ such that the last deletion of x occurred at time $i - k$.*

Condition 1 ensures the indistinguishability of the sensitive values in each QI-group. Violating m-uniqueness may result in homogeneity attacks due to duplicate sensitive values in QI-groups. The larger is the m value, the more difficult is the disclosure. Condition 2 states that if an individual's sensitive value is unchanged during her lifespan then she must bear the same set of signature throughout until she is deleted from the

dataset. It is an m-invariance-like privacy enforcement. Condition 3 states that if an individual has been re-inserted, then the signature of her QI-group must be redrawn from the last QI-group she was previously assigned to. This condition ensures the individual-based protection where each record has a "memory" and an adversary cannot infer any sensitive information even after combining multiple non-consecutive releases.

Lemma 4.1 *If the anonymization mechanism follows τ-safety , then at any time p, $risk(x) \leq \frac{1}{m}$ $\forall x \in \mathcal{X}$.*

Proof The proof of this lemma follows directly from m-invariance principle (Lemma 3 in [112]). Since τ-safety handles all the τ-attacks such that the *persistent invariance* [112] is maintained in each QI-group, it conforms to the privacy guarantee provided by m-invariance. Then it satisfies $\text{risk}(x) \leq \frac{1}{m}, \forall x \in \mathcal{X}$.

4.3.4 Enforcing τ-safety

In this section, we elaborate the enforcement of τ-safety for a sequence of public releases.

Definition 4.12 (Sequential publication of dynamic data:) *Given $T = (R_1, \ldots, R_p)$, $p > 1$, a sequence of raw releases of a fully dynamic dataset; given $T^* = (R_1^*, \ldots, R_{p-1}^*)$ the series of $p-1$ first anonymous public releases from T such that it satisfies δ-privacy ($\delta \in [0,1]$) under $\mathcal{BK}$; the problem of dynamic dataset republication in the presence of arbitrary updates and given chainability-based background knowledge $\mathcal{BK}$, is to publish the p^{th} release R_p^* in T^* such that δ-privacy is satisfied for T^*.*

Such a publication series $(R_1^*, \ldots R_p^*)$ is called a τ-safe m-invariant series. We also say that this series satisfies τ-safety.

Let us revisit the situations in which the adversary attempts to apply the τ-attack on Tables 4.3(a) and 4.4(a) in the three scenarios mentioned in Section 4.1.1

Applying the τ-attack in scenario I, the adversary knows that p_2 has records in equivalence classes 1 and 4 in Tables 4.1 and 4.3(a) respectively. Despite this knowledge, adversary will not be able to identify the sensitive values of p_2. The adversary will try to reason as follows: in first release, p_2 is in the equivalence class with sensitive values *{cataract, pneumonia}*. In second release, she is in the equivalence class with sensitive values *{diarrhea, gastritis}*. Based on this knowledge and the event list of p_1 and p_2, there is not even a minute possibility for the adversary to disclose p_2's sensitive attribute values as both sets are entirely different. Similarly for scenario II, following the condition 2 in definition 4.11, since the signature of QI-group of p_2 remains the same from time 2 to 3, the adversary is unable to filter out any sensitive value. Applying the τ-attack in scenario III on Tables 4.1 and 4.4(a) for p_4, the adversary won't be able to breach the privacy of p_4. Since Table 4.4(a) conforms to τ-safety, it handles the re-insertion of p_4 accordingly.

4.3.5 About Counterfeits

Critical absence is a side-effect of any sequential publication of dynamic dataset with persistent invariance property such like m-invariance or τ-safety. And *counterfeits* or fake records, are the preferred parade.

Definition 4.13 (Critical Absence:) *Let S_i and S_{i+1} be the multisets of sensitive values in consecutive micro-data tables R_i and R_{i+1} respectively. Then critical absence holds in R_{i+1} iff $S_i - \Delta Q \not\subseteq S_{i+1}$, with ΔQ the multiset of sensitive values coming from QI-groups that have been totally removed from time i to time $i+1$.*

Intuitively, the series of S_i's should be inflationary to prevent from critical absence, since persistent invariance requires to keep signatures unchanged. Exception to this rule raises with ΔQ such that it allows for non inflationary S_i's as far as the missing values come from newly discarded QI-groups.
Counterfeits are necessary for overcoming the problem of critical absence. The lack of missing sensitive values must be removed by adding fake records. Though studied only for external updates, the number of these counterfeit records highly depends on the distribution of sensitive values in QI-groups. Xiao et al. [112] empirically show that the percentage of counterfeit tuples added to the public release is well below 0.1%. This is because, by time, when new records are inserted, they fill the missing sensitive values thereby replacing the counterfeits from resulting publication. Internal updates may give rise to the same situation as deletions, apart from the fact that they are able to fill other missing sensitive values or replace the counterfeit values. Then, the ultimate problem is to find a τ-safe R_p^* such that multiset S_p is optimally partitioned in QI-groups w.r.t counterfeit tuples and utility.

4.4 Analysis for Achieving Optimal τ-safe Release

In this section, we analyze the problem of achieving an optimal τ-safe release. Any sequential data publication model that aims to limit the privacy risk must take into account the distribution of sensitive values in sequential releases in order to satisfy δ-privacy. Consider the multisets S_{i-1} and S_i of sensitive values in consecutive micro-data tables R_{i-1} and R_i respectively. Figure 4.1 depicts a general view of sensitive values between S_{i-1} and S_i at time *i*. The multiset $S_i^=$ corresponds to the sensitive values that are common to both S_{i-1} and S_i (the sensitive values in $S_i^=$ are unchanged from $i-1$ to i). The multiset S_i^+ contains the sensitive values that are entirely new at time i. The multiset S_i^- (dark gray shaded area) contains the sensitive values that are deleted from $i-1 \rightarrow i$ and do not have any corresponding entry in either $S_i^=$ or S_i^+. For example, in Table 4.2, S_i^-{*glaucoma, pneumonia*}, $S_i^=${*cataract, flu, flu, gastritis*} and S_i^+{*diarrhea, gastritis*}. Figure 4.2 depicts this distribution.

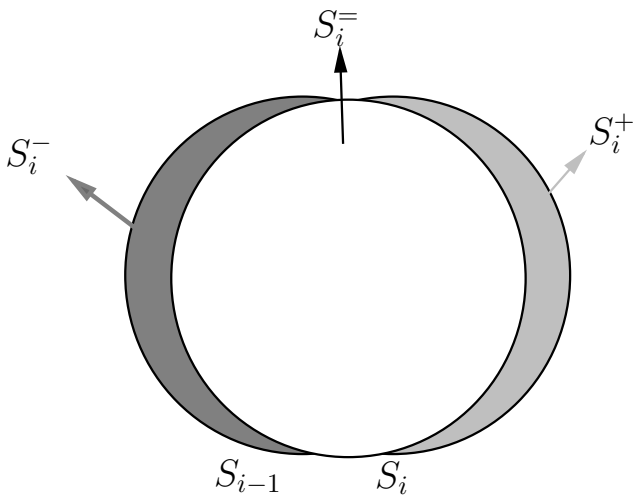


Figure 4.1: Venn diagram of sensitive values in S_{i-1} and S_i

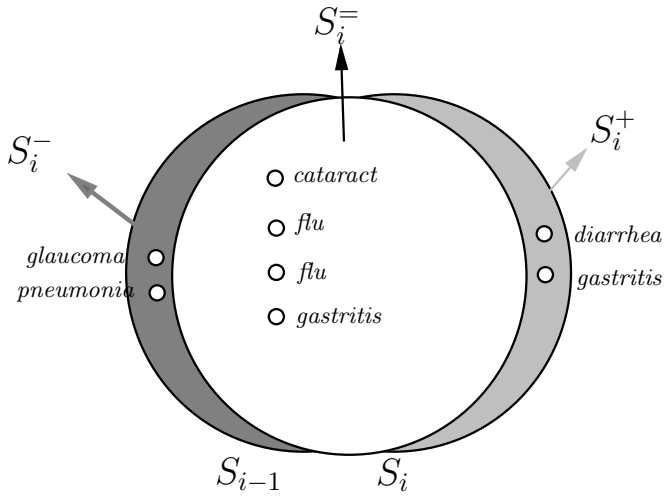


Figure 4.2: Example of sensitive values updates at time i from Table 4.2

As explained in Section 4.3.5, the multiset S_i^- basically contains the critical absences for which we need counterfeit records and for enforcing the persistence invariance in each QI-group, the sensitive values in $S_i^=$ and S_i^+ are used to populate old QI-groups or creating new ones if necessary. The question arises "*how to optimally distribute the sensitive values in* $S_i^=$ *and* S_i^+ among the QI-groups?"
We try to answer this question using the example in Figure 4.3 for Table 4.2. There

are three QI-groups that need to keep their signatures unchanged in the second release, in order to maintain persistent invariance among themselves. Figure 4.3 portrays the situation of these QI-groups (namely G_1, G_2 and G_3) and the multisets S_i^-, $S_i^=$ and S_i^+. Since the groups G_1 and G_2 contain the sensitive values from S_i^- i.e., *pneumonia and glaucoma* respectively, counterfeit records can directly be assigned to them. These groups can thus easily be populated by simple assignment of their remaining sensitive values from either $S_i^=$ or S_i^+. For the group G_3 however, several assignments are possible. The arrows on Figure 4.3 highlight these assignments. These assignments may be decided by exploiting several properties of the tuples having those sensitive values e.g., by calculating the QI-based distance between the two records (See Section 4.5.2). This problem of assignment is highly combinatorics.
An interesting aspect about the counterfeit records is that *they may be used to further increase the utility of final release*. For instance, in the above example, the group G_2 needs one *flu* to be completed and $S_i^=$ contains 2 *flu*s, either of which can be assigned to G_2. Since G_2 contains a counterfeit record (remind that the counterfeit records have a minimal effect on final generalization since they have null values on their QI attributes [112]), we can choose one *flu* for G_2 which will minimize generalization in G_3 (remind that G_3 also needs one *flu* to be completed). This way the counterfeit in G_2 will help reducing the generalization in both G_2 and G_3. The above analysis reveals

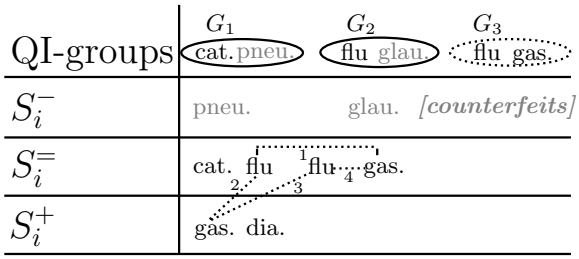


Figure 4.3: All possible assignments in G_3 indicated by numbered dotted lines

that there exist several ways of partitioning S_i in QI-groups such that finding the optimal partitioning depends on the assignment of sensitive values in previously defined QI-groups and new QI-groups. In what follows, we present our approximate solution to achieve τ-safety.

4.5 τ-safe m-invariant Generalization

Since m-invariance is the state-of-the-art in preventing the τ-attacks in the presence of external updates, we enforce m-invariance principle in our counterfeit anonymization using a variant of m-invariance bucketization algorithm [112]. m-invariance bucketization algorithm classifies the records (*into so called buckets*) depending upon their

sensitive values in the previous release. As the ultimate goal of m-invariance is not the quality of resulting publication, it does not take into account the proximity of records before assigning them to proper buckets. In this section, we explain that the utility of public release could be substantially increased if we consider the proximity of records in *m*-invariance algorithm [112] while assigning them to proper buckets. This section details the procedure for the publication of R_p^*. We start by elaborating the main contents of the proposed solution. The subsequent sections describe different building blocks of the solution.

4.5.1 A Bucketization Algorithm

The goal of the algorithm is to sustain the privacy of individuals while attaining higher utility. By high utility, we mean two major goals *i)* the generalization of the QI values must be as minimum as possible *ii)* the number of counterfeit records are kept to minimum. We classify the records-to-be-published as follows:

1. X_p^{new}: $\forall x \in R_p$, if $\tau(x)[p] = i|r|u$ then $x \in X_p^{new}$
2. X_p^{same}: $\forall x \in R_p$, if $\tau(x)[p] = -$ then $x \in X_p^{same}$
3. Del : $\forall x \in R_p$, if $\tau(x)[p] \neq i|r|u|-$ then $x \in \mathsf{Del}$.

The interesting features of the proposed algorithm are:

- The algorithm is incremental and thus it does not require to scan history of public releases to anonymize current release.
- The space and time complexity of the proposed algorithm is independent of the number generalized tables. This property is important in the republication scenarios where the number generalized tables increases monotonically.
- The algorithm substantially improves on the utility w.r.t. m-invariance algorithm by taking into account the proximity of records before assigning to them to proper QI-groups.
- The algorithm upgrades the privacy guarantee of m-invariance to τ-safety specifically making the resulting publication immune to the adversarial attacks based on event lists.
- The m-invariance algorithm does not permit the p_{th} publication if the records in X_p^{new} are not m-*eligible* (i.e., at most $\frac{1}{m}$ of the records in X_p^{new} have the same sensitive values). The proposed algorithm provides an added flexibility to the data publishers by removing this blocking constraint.
- Last but not the least, the algorithm ensures individual based protection rather than record based protection.

The first step of the algorithm prepares Del.

4.5.1.1 Preparing Del

To handle the event list τ for each individual, we propose the use of a "delete map" denoted by Del. Del is managed internally and is updated on the arrival of new micro-data. The core idea of the delete map is to maintain a memory of individuals with their previous signatures. Suppose an individual *x* is deleted from micro-data at any time *i*. She will be added to Del with the signature of QI-group he last appeared in. The schema of Del is given as:

$$\text{Del}: \quad \begin{matrix} key = t[ID] \\ value = Sig(t[ID]) \end{matrix} \tag{4.5}$$

where *t[ID]* is an individual identifier through which she can be tracked, *Sig(t[ID])* is the signature of her last QI-group. Precisely, the signatures of deleted records are taken from previously anonymized releases and are kept in Del for further processing.
In order to keep check on the size of Del, it is updated during anonymization. The worst case space complexity for Del is equal to the size of dataset itself.

4.5.1.2 Phases of τ-safe m-invariant generalization

As stated before, the m-invariance algorithm [112] does not permit the p_{th} publication if the records in X_p^{new} are not m-eligible. This prerequisite is used withing the heuristic of the bucketization algorithm for m-invariance achievement. According to our analysis, this is a sufficient condition not a necessary one. The only side-effect, if X_p^{new} is not m-eligible is that, there will be more counterfeits. Thus, we remove the constraint on X_p^{new} to be m-eligible. For publication of p_{th} release i.e., R_p^* , we only need previously anonymized release R_{p-1}^*, micro-data R_p and Del. The overview of τ-safe m-invariant anonymization is presented in Algorithm 1. We also follow our example in Tables 4.3(a) and 4.4(a).

Algorithm 1: τ-safe Generalization

Require: R_p, R_{p-1}^*, Del
Calculate: $X_p^{new} = R_p - R_{p-1}, X_p^{same} = R_p \cap R_{p-1}$
Fix-reinsertions(X_p^{same}, X_p^{new})
BUC := Classify(X_p^{same}, Del, R_{p-1}^*)
R_p^*:= Balance(BUC)| Finalize-Assignment |Partition |Generalize
Publish(R_p^*)

We divide the τ-safe generalization algorithm in following major phases:

1. Fix X_p^{same}
2. Classification
3. Balancing

4. Finalize-Assignment
5. Partition
6. Generalize

Fix X_p^{same}: This phase prepares X_p^{same} by moving the re-inserted records from X_p^{new} to X_p^{same}. A record $t \in X_p^{new}$ is moved from X_p^{new} to X_p^{same} if $\tau(t[ID])[p] = r$ i.e. *t* is re-insertion in R_p. After this step, all the reappearing records, without any modification in their sensitive values since their last deletion, are moved to X_p^{same}. This will make sure that the signatures of the re-inserted records remain the same. For a re-inserted record *t*, if *t[S]* is modified such that *t[S]* is *subsumed* by its previous signature, *t* is still moved to X_p^{same}. *Consequently, the record t will maintain its previous signature thereby minimizing the possibility of any possible inference on t.* Remind that in Section 4.2.1, we emphasized that the re-insertion of a record cannot be thought of as an external deletion and an external insertion otherwise the signature of the re-inserted record may not remain the same in the current release thereby allowing τ-attack.
In Figure 4.4, after the publication of R_2^*, Del contains a single entry for p_4 because of its deletion from R_2. At time 3, during the phase of preparing X_p^{same}, X_p^{new} contains only p_4 since it is the only insertion. Since p_4 exists in Del already, it is moved to X_p^{same} and thus at time 3, X_p^{same} contains all the tuples while the entry for p_4 has been deleted from Del.

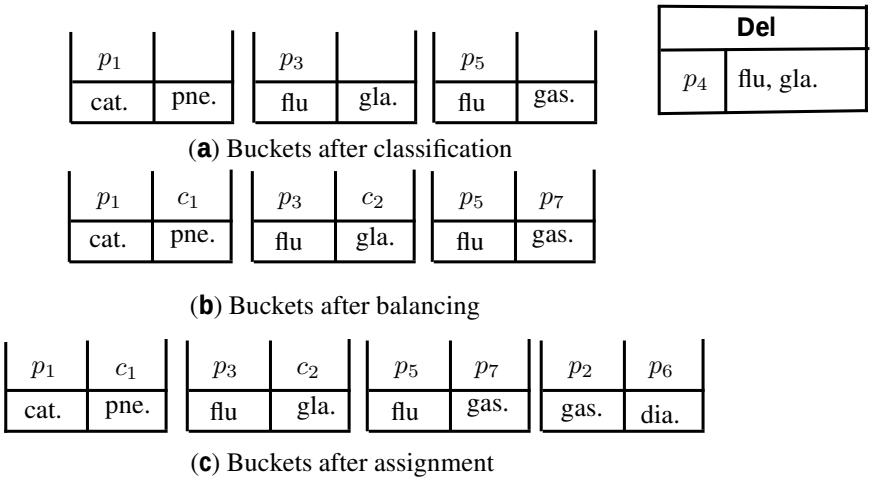
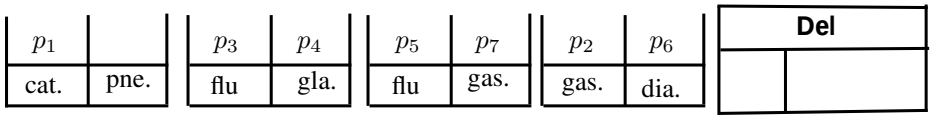
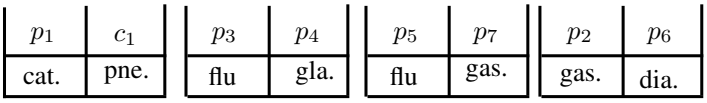


Figure 4.4: Illustration of τ-safety for R_2^*

Classification: This phase creates new buckets from records in X_p^{same} by using R_{p-1}^* and Del such that the QI-groups in R_{p-1}^* make the signature of each bucket. If a record *t* is a re-insertion, it has no corresponding entry in R_{p-1}^*. Subsequently, new bucket is



(a) Buckets after classification



(b) Buckets after balancing

Figure 4.5: Illustration of τ-safety for R_3^*

Algorithm 2: Classification

Require: X_p^{same}, Del,R_{p-1}^*
  Initialize BUC := $\oslash$
  for all records t in X_p^{same} **do**
    if $\tau(t[ID])[p] = r$ **then**
      B := Create-Bucket(Sig(Del.t[ID]))
      Delete-Entry(Del.t[ID])
    else
      B := Create-Bucket(Sig($[t]_{p-1}$))
      put(B ,t)
    end if
    if $B \notin BUC$ **then**
      BUC := BUC $\cup$ {B}
    end if
  end for
Ensure: **return** BUC

created using Del (t is located in Del since it is a re-insertion). A *bucket* is the major building block of our algorithm. A bucket *B* consists of *m* or more *entries and an entry* $e_i \in B$ *contains a sensitive value s and a set of records such that t[S]=s. We say that the* ***signature of a bucket B*** *(denoted onwards by Sig(B)) is a set of sensitive values that can be assigned to it and it contains at-least* m *sensitive values.* At the end of this phase, we have all the buckets for the records in X_p^{same}. Also, the Del is updated in this phase. After the creation of buckets for reinserted records, there is no need of keeping an entry for them in Del. Thus each reinserted record is deleted from Del in classification phase. The complexity of this phase is O($|X_p^{same}|$). Algorithm 2 depicts the process of classification.

Figure 4.4(a) and 4.5(a) depict the classification phase at times 2 and 3 respectively. At time 2, X_p^{same}={p_1, p_3, p_5}. From Table 4.1, classification phase creates buckets

from the signature of records in X_p^{same} as shown in Figure 4.4(a). Similarly at time 3, X_p^{same}=$\{p_1, p_2, p_3, p_4\ p_5, p_6, p_7\}$, this phase creates buckets from R_2^* for the records in X_p^{same} except p_4. Since p_4 is a re-insertion, its signature from Del is used to create a separate bucket. Figure 4.5(a) depicts this process.

Balancing: This phase takes as input a set of buckets created from the classification phase and balances them. *A bucket is said to be balanced if every sensitive value in its signature is associated with the same number of records.* This phase focuses on individual sensitive values rather than signatures and starts by identifying a set of unbalanced buckets. The buckets are balanced by either assigning a counterfeit tuple or by choosing a record from X_p^{new}. For missing sensitive values the algorithm simply assigns counterfeit records because they do not have a corresponding entry in X_p^{new}. The remaining buckets are then balanced by using X_p^{new}.
Figure 4.4(b) and 4.5(b) depict the balancing phase at times 2 and 3 respectively. At time 2, since X_p^{new} does not contain the sensitive values for *pneumonia and glaucoma*, counterfeit tuples are added directly to the first two buckets in Figure 4.4(b). X_p^{new} contains *2 gastritis and 1 diarrhea* and thus can be used to balance the third bucket. Thus at time 2, after balancing phase, X_p^{new} contains *1 gastritis and 1 diarrhea* and all the buckets are now balanced. Similarly at time 3, all the buckets are already balanced except the first one. Since X_p^{new} does not contain *pneumonia* for balancing the first bucket, counterfeit record is assigned to it. Thus all the buckets in Figure 4.5 are now balanced.

Finalize-Assignment: *The fundamental problem we are emphasizing here is how to optimally partition the multiset of sensitive values S_p into buckets such that i) any bucket created by using S_p is balanced ii) if new buckets are created, they satisfy m-uniqueness property defined in Definition 4.11.*. This phase assigns the remaining records in X_p^{new} *to the respective buckets*. A tuple $t \in X_p^{new}$ can be assigned to a bucket *B* if $t[S] \in Sig(B)$. If a bucket does not exist, a new bucket is created which must follow m-uniqueness constraint.
Remind that we lifted the restriction on X_p^{new} being m-eligible. Thus, this step starts by making X_p^{new} m-eligible so as to ensure m-uniqueness constraint. The obvious advantage of relaxing this constraint is the liberty we offer to the data publisher for releasing any kind of micro-data but at the cost of more counterfeits. The algorithm adds a counterfeit record in X_p^{new} by choosing a random sensitive value *s* from *Dom(s)*. A counterfeit record has null value for each of the quasi-identifier attributes. For example, a counterfeit record c_1 in Table 4.3(a) is of the form:

$$c_1 = \langle \oslash, \oslash, \oslash, pneumonia \rangle \tag{4.6}$$

The maximum number of added counterfeits to make X_p^{new} m-eligible is $m-1$. Remind that if X_p^{new} is already m-eligible, there is no need to add any counterfeits.

We introduce two variables for this phase namely α and β. β is used to manage the signature of a new bucket and α helps in assigning correct number of records to the newly created bucket in order to ensure that the it remains balanced. Once X_p^{new} is m-eligible, the algorithm runs iteratively to move a set X_B of $\alpha.\beta$ tuples from X_p^{new} to the buckets containing $\beta \geq m$ sensitive values. Note that we follow the same procedure for computing the values of α and β as described by Xiao et al. [112] in the assignment phase of m-invariance algorithm. This helps in assigning all the records in X_p^{new} to the balanced buckets (Lemma 5 in [112]). Then, β is used to form a signature for a bucket, *say B* where *B* is created if it does not exist. The values of α and β are computed by making use of three inequalities (See Algorithm 3). Once the values of α and β are determined, the following strategy is used to build the set X_B for the assignment to bucket *B*: Let $\mathcal{S} = (s_1, s_2, \ldots, s_\lambda)$ be the list of distinct sensitive values in X_p^{new}. At the start of each iteration, $\mathcal{S}$ is sorted descendingly on the count of sensitive values such that the most frequent sensitive value is the first to appear in $\mathcal{S}$. The algorithm picks β sensitive values from $\mathcal{S}$ for the signature of bucket *B* such that the *B* has signature $(s_1, s_2, \ldots, s_\beta)$. The algorithm then picks α tuples from X_p^{new} for each entry in *B* by using a distance function (See Section 4.5.2). For each $s_i \in \mathcal{S}$, the algorithm randomly moves α tuples with sensitive value s_i from X_p^{new} to X_B. The process continues for each sensitive value in $s_1, s_2, \ldots, s_\beta$ and at the end of each iteration, records in the X_B are moved to *B*. After the assignment phase, all the records in X_p^{new} are assigned to the balanced buckets. Algorithm 3 depicts the procedure for finalizing the assignment of the records in X_p^{new}.

Figure 4.4(c) depicts the assignment phase at time 2. After the balancing phase, X_p^{new} = {p_2, p_6}. Since assigning any of these records to previously defined buckets will break their balance, the algorithm simply creates a new bucket from X_p^{new} since $|X_p^{new}| = m$ i.e., X_p^{new} is m-eligible. Then after the assignment phase, every bucket remains balanced.

Partition : This phase takes as input the set of buckets from the previous phase and splits them to achieve better generalization. As all the buckets are balanced, they actually contain a number of records that is a multiple of the total number of entries in a bucket. Let n denote the number of entries in a bucket. The buckets are split such that for every bucket B and for any entry e_i in B, $|e_i| = 1$ i.e. there exist exactly *one* record in each entry of every bucket. Splitting further improves the quality of generalization because the resulting buckets are then as small as possible and ready for straightforward generalization.
Each bucket is inspected in turn. If the total number of records in B i.e $count(B) > n$, B is split into two child buckets such that the resulting buckets are still balanced and each child bucket has size n. Totally $\frac{count(B)}{n}$ splits are performed.

Algorithm 3: Assignment

Require: X_p^{new}, BUC

1: Initialize: λ = total number of distinct sensitive values in X_p^{new}
2: **if** X_p^{new} is not m-eligible **then**
3: add counterfeits in X_p^{new}
4: **end if**
5: **while** $|X_p^{new}| \neq 0$ **do**
6: γ := $|X_p^{new}|$
7: calculate $\mathcal{S} := (s_1, s_2, \ldots, s_\lambda)$ i.e., $s_i (1 \leq i \leq \lambda)$ where s_i is the i_{th} most frequent sensitive value in $\mathcal{S}$
8: $\beta := m$
9: α := largest positive integer that satisfies the inequalities below
10: **if** ! ($\alpha \leq s_\beta$ and $s_1 - \alpha \leq \frac{(\gamma - \alpha * \beta)}{m}$ and $s_{\beta+1} \leq \frac{(\gamma - \alpha * \beta)}{m}$) **then**
11: $\beta = \beta + 1$
12: goto line 9
13: **end if**
14: Create-bucket B with Sig(B)= $(s_1, s_2, \ldots, s_\beta)$
15: BUC := BUC $\cup$ {B} (Create bucket if does not exist in BUC)
16: **for** i = 1 to β **do**
17: randomly move α tuples with sensitive value s_i from X_p^{new} to *B*
18: **end for**
19: **end while**
20: **return** BUC

For a bucket B, the process starts by randomly picking a record *t* from its first entry. In the next step, the distance of *t* from each record in the next entry is calculated. The record with minimum distance is kept and the *mean* of *t* and chosen record is calculated. The process continues picking up a record from each entry of B based on minimum distance from the *mean* (*mean* is updated on every selection of a new record from the next entry). Finally, a child bucket B_{new} is created such that $Sig(B_{new}) = Sig(B)$ and all the chosen records are inserted into the corresponding entries of B_{new}. The process continues until the required condition is met i.e. $count(B) = s$.

Generalization After the partitioning phase, the algorithm simply performs generalization on each QI attribute of each bucket.

Publication Identifier attributes are removed and R_p^* and counterfeit statistics are published.

4.5.2 Distance Function

Consider the micro-data in *n*-dimensional euclidean space where *n* is the number of QI attributes. The distance between the two records in this *n*-dimensional space can be calculated using any *distance* function. This distance is instantiated by a basic euclidean distance in the multidimensional Euclidean space. The main purpose of the distance function is to reduce the amount of generalization. Instead of randomly picking any record to assign to any bucket, the function calculates the distance between two records thereby gathering closer records. Remind in Section 4.5.1.2 where this function is used all along the algorithm. The usual euclidean distance can be used between the records t_1 and t_2. It is given by:

$$\text{Euc-distance}(t_1, t_2) = \sqrt{\sum_{i=1}^{n} (t_1(i) - t_2(i))^2} \tag{4.7}$$

where i denotes the i_{th} QI attribute value for t_1 and t_2 and n is the number of QI attributes.

4.6 Experimental Validation

In this section, we present the experimental results to check the performance of our approach and provide a comparison with the m-invariance algorithm. The quality of public releases is tested with various quality measures. Moreover, the variation in counterfeit counts have been tested under various settings.

Category	Description
Compiler	Microsoft Visual C++ 2005
Database	PostgreSQL
Operating System	Windows 7
CPU	Intel Xeon CPU W3520 2.67 Ghz
Memory	4096MB
Hard disk	500GB

Table 4.8: Experimental setup

4.6.1 Preparation and settings

The experimental setup is given in Table 4.8. We used "Adults" dataset taken from U.C. Irvine Machine Learning Repository. This dataset, also known as "Census Income" dataset, contains the data about individuals in the USA. We purged all records with missing values and randomly chose 160,803 tuples for our experiments. *We used the attributes age, capital-gain and fnlwgt as quasi-identifiers and occupation as sensitive attribute*. All the attributes are discrete and have domains respectively 94, 5, 127 and 50 distinct values. Since m-invariance does not permit the anonymization of current release if new records are not m-eligible, we prepared the experimental protocols such that the new records are always m-eligible for fair comparison.

Though we did not have access to their code, we implemented the algorithm in [112] keeping it as close as possible to the original one. Since our main purpose in these experiments is to highlight the problems caused by *internal updates* in *m*-invariance, we define two separate parameters to verify our results:

- **External update frequency** : We initially took 60,000 rows for our first release R_1 , chosen randomly from the raw dataset. Then, for each subsequent release R_i, we randomly deleted 3000 rows from R_{i-1} and put them in delete pool and then inserted 5000 tuples randomly selected from the remaining tuples. The dataset was republished 20 times.
- **Internal update frequency:** Since our main task was to study arbitrary internal updates, we set a parameter defining the internal update frequency. By default it is set to 5000 (out of which 1000 tuples are taken from delete pool and they correspond to re-insertions). The internal updates from $i-1$ to i in the given dataset have been managed as follows:
  - age grows by 1 until it reaches 120.
  - fnlwgt and capital-gain can remain the same or be updated to any other value in their domain.
  - as our focus is internal updates on sensitive values , we allow arbitrary updates in occupation to any other value in its domain i.e. occupation of a person can remain the same or be modified to any of the remaining 49 values in its domain.

4.6.2 Failure of *m*-invariance and Other Generalization Models

Since Xiao et al [112] have shown that existing generalization models are unable to cope with external updates, in the first set of experiments, we apply internal updates randomly to find out the vulnerable records in case of m-invariance. ***Vulnerable records in m-invariance refer to those records which are unable to keep their signatures constant in following releases due to either modification in their sensitive values or being re-insertions in the current release.*** As the update rate increases, there is a dramatic increase in the number of vulnerable records. Also, as the number of public releases increases, we have gradual increase in the number of vulnerable records. Furthermore, an interesting aspect is the variation of the parameter m is that with higher m, the vulnerable record count is low which is due to the fact that modified records might fall into the same group as the size of the group is quite large thereby keeping the signatures same. Thus, higher m lowers the number of vulnerable records (caused by internal updates).

4.6.3 Anonymization Quality

These sets of experiments focus on quality of resulting releases. The main idea is to evaluate the extent to which the dataset has been distorted when generalizing records. We adopted generic quality measures, i.e. measures that do depend neither on the application domain nor on a specific usage of the public release. We then evaluate the anonymized releases with three different measures: *certainty penalty (CP) [114] , discernibility penalty (DCP) [9]* and KL-divergence [59] See Section 2.5.1.

The CP evaluates the loss of accuracy in the description of equivalence classes, whereas the discernibility penalty quantifies the extent to which the size of the equivalence classes is close to the parameter m. KL divergence provides an entropy measure that estimates the information loss in the public release.

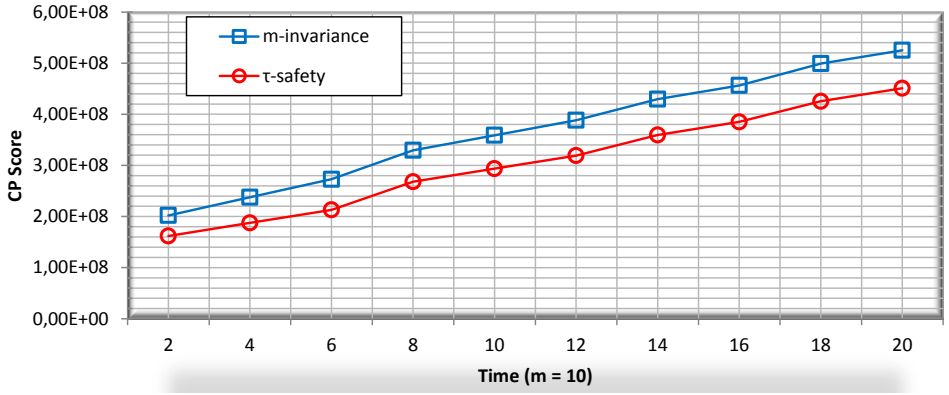


Figure 4.6: Certainty Penalty (CP)

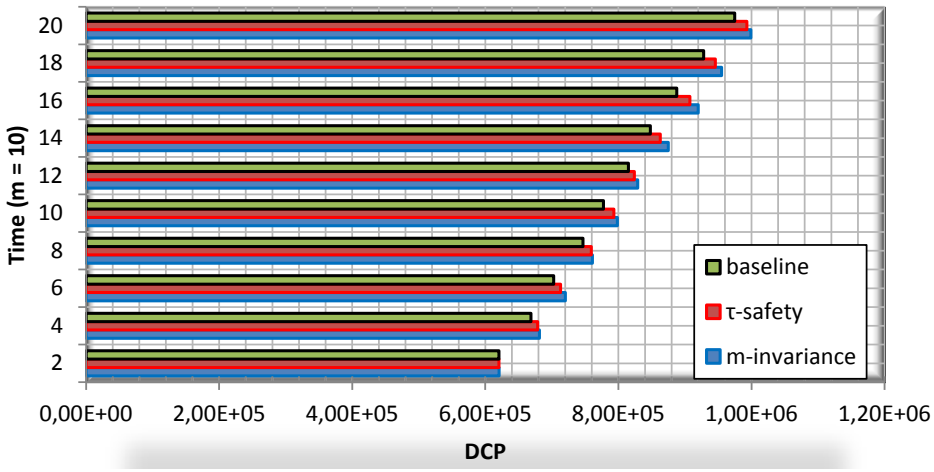


Figure 4.7: Discernibility Penalty (DCP)

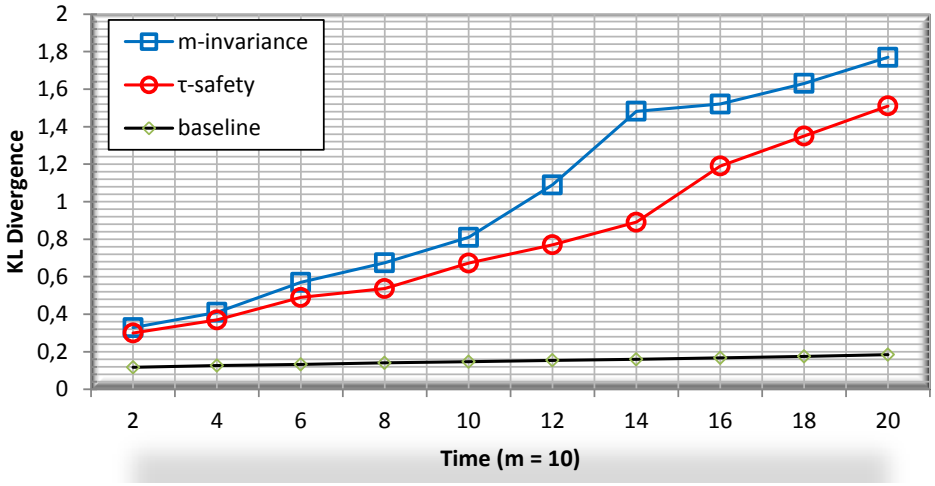


Figure 4.8: KL divergence

The results are presented in Figure 4.6, 4.7 and 4.8 for certainty penalty, discernibility penalty and KL divergence respectively. As both m-invariance and τ-safety focus on minimizing the size of QI-groups by specifying the value of m, DCP score for both are thus not very far. But the main difference can be seen from CP score. CP score for m-invariance is much higher than that of τ-safety. CP score suggests that the intervals created by m-invariance algorithm are unable to control the generalization because m-invariance assigns the records randomly in the buckets. On the contrary, τ-safety assigns records in the buckets based on the distance thereby resulting in better CP score. Similarly, τ-safety shows less information loss than m-invariance as measured by KL divergence.

4.6.4 Query Accuracy

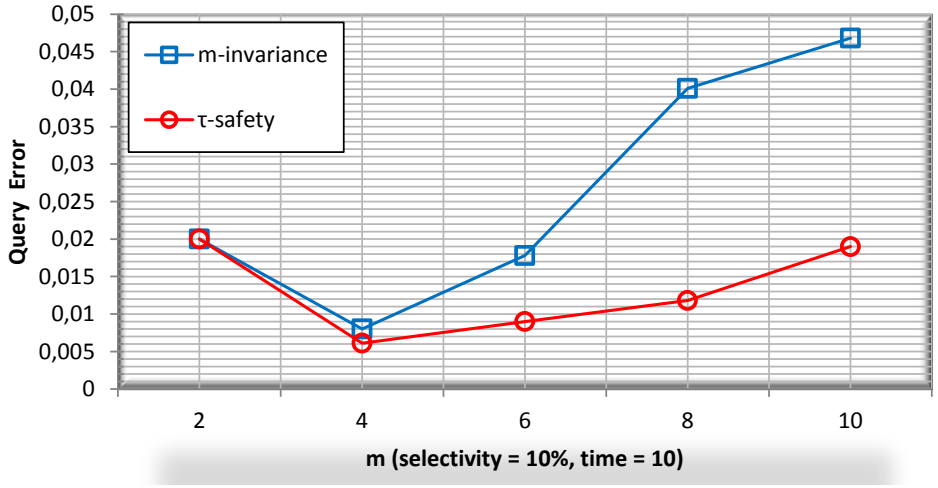


Figure 4.9: Query Error with varying m

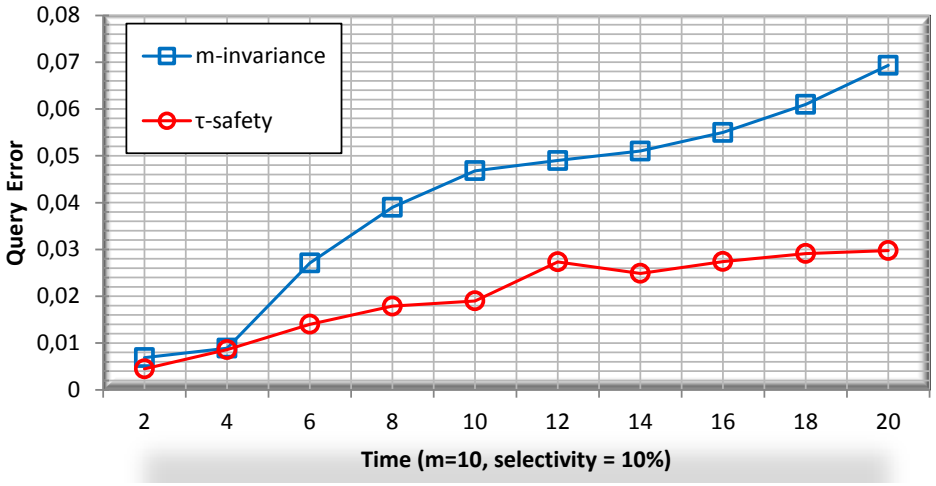


Figure 4.10: Query Error with varying Time

Relative query error rate is a commonly used method to measure utility [68, 112]. We computed the relative query error by using the protocols discussed in Section 2.5.2. We compare the utility of m-invariance and τ-safety using the relative query error rate of 1000 randomly generated range queries . The query error increases smoothly as time evolves (Figure 4.10), because newly inserted records are assigned to a QI-group based on distance which means reduction on the intervals of QI values, as a result the error will not increase anymore when re-publishing enough times. While the error rate goes

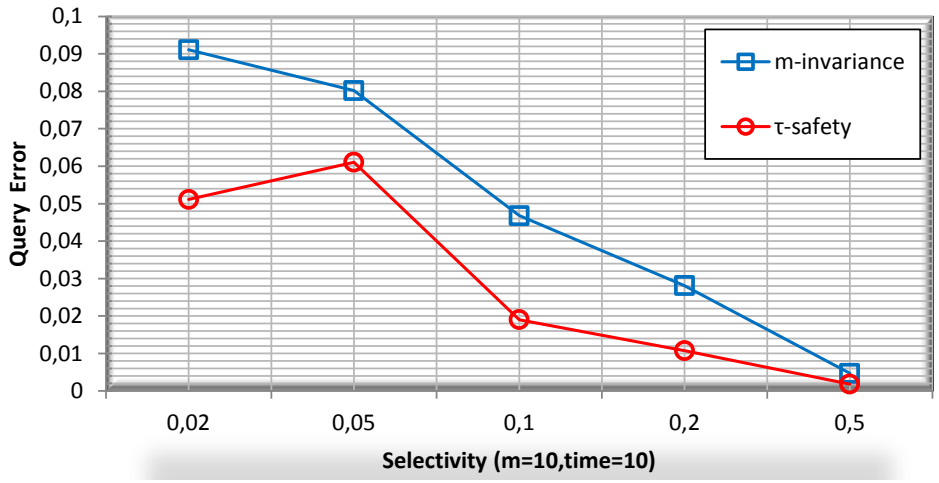


Figure 4.11: Query Error with varying Selectivity

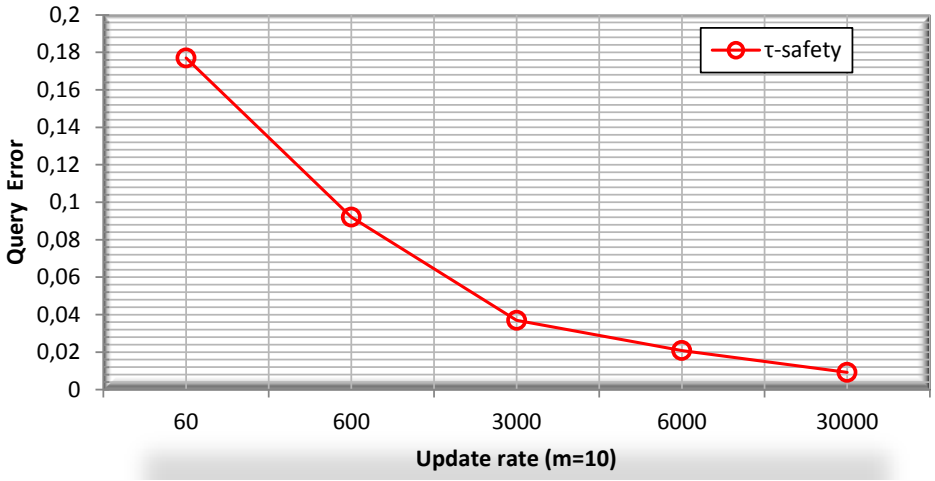


Figure 4.12: Query Error with varying Update rate

up for more selective queries i.e. Figure 4.11, τ-safety tends to produce better results than its counterpart. Figure 4.9 shows the error rate with varying m . At last, in Figure 4.12, we show that with higher update rate, the error rate reduces. A larger update rate indicates more flexible sensitive values assignment to the buckets. As a consequence, more records can be assigned to a bucket, thereby facilitating efficient generation of QI-groups and improved query accuracy.

4.6.5 Counterfeits

The second set of experiments focus on comparing the number of counterfeits produced by both algorithms. Figures 4.13 and 4.14 depict that even with a small number of internal updates, many releases do not even need counterfeit tuples. As can be seen, the counterfeits produced by τ-safety reaches the baseline for the minimum number of counterfeit tuples. In contrast, due to strict implementation of m-eligibility, m-invariance encounters the situations in which more counterfeits are required than the baseline. . This is an encouraging result, for it indicates that τ-safety algorithm provides the required privacy with better utility and with minimum possible counterfeits. Figure 4.15 shows the variation in counterfeits with varying update rate. For a smaller update rate, the value is higher due to the fact that more QI-groups are short of sensitive values. As the update rate increases, the number of counterfeits becomes smoother because internal updates in some sensitive values replace the counterfeits in the subsequent releases. Then, in the presence of both internal and external updates, the number of counterfeits is always low.

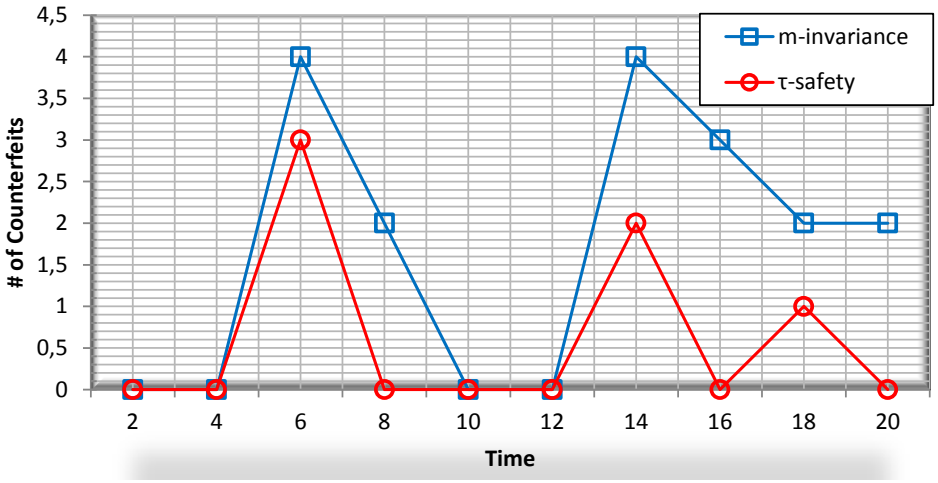


Figure 4.13: Counterfeits with varying Time

4.6.6 Anonymization Efficiency

As per experimental setup, the number of records for each publication is incremental as time evolves. We report the time cost for the publication of R^*_{10} in order to present precise time for one single republication. Figure 4.17 demonstrates the computation cost with varying update rates. With larger update rate, we can achieve higher utility but the cost is higher because more records are assigned to the same bucket, which results in higher cost when splitting and generating QI-groups in the last phase. Figure 4.16

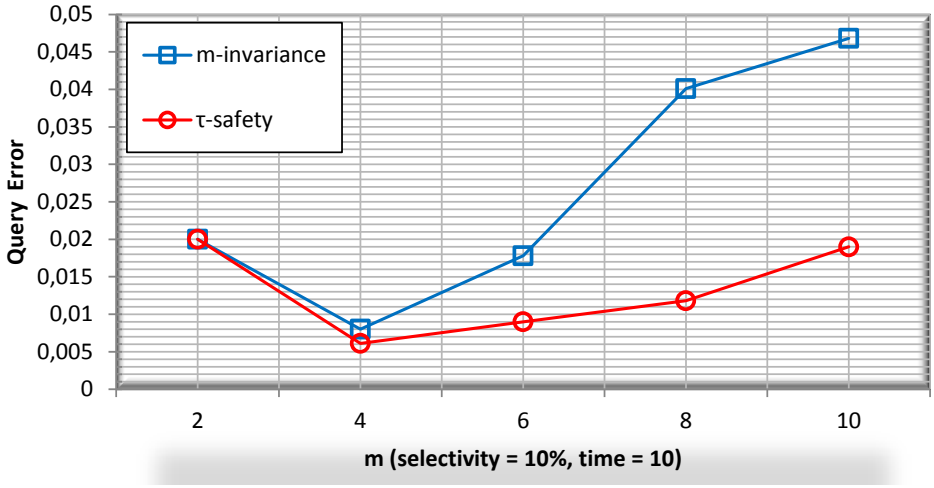


Figure 4.14: Counterfeits with varying m

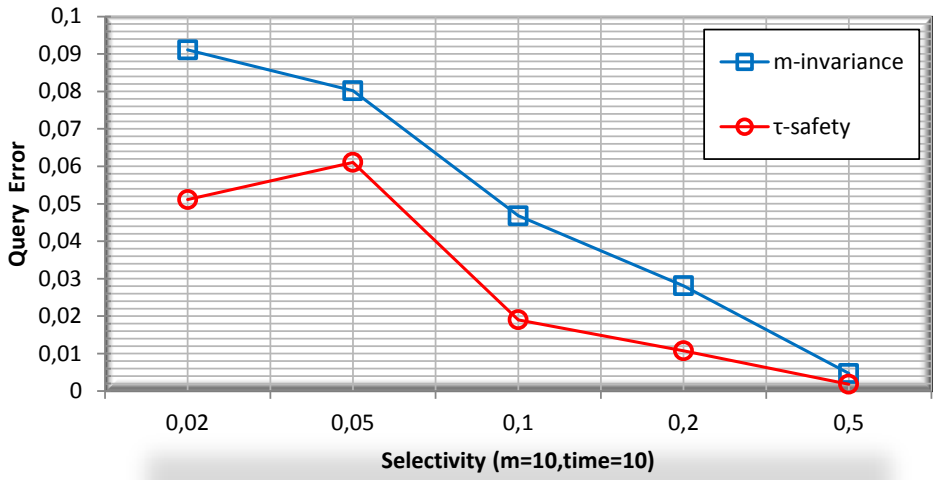


Figure 4.15: Counterfeits with varying update rate

demonstrates the cost with varying m. The cost decreases when m increases which is due to the fact that there are less number of records in buckets due to large signatures and splitting and generating QI-groups cost smaller. τ-safety performs better than m-invariance due to the fact that it partitions the buckets faster than how m-invariance does.

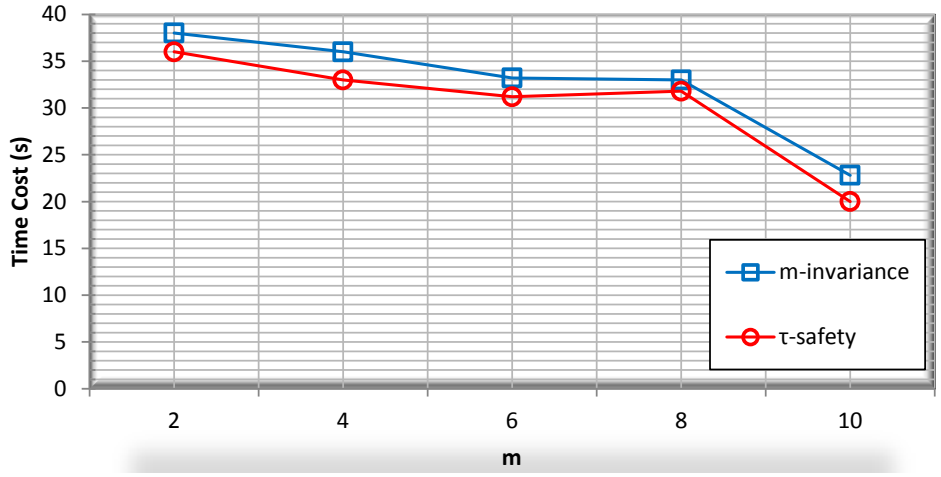


Figure 4.16: Cost by m

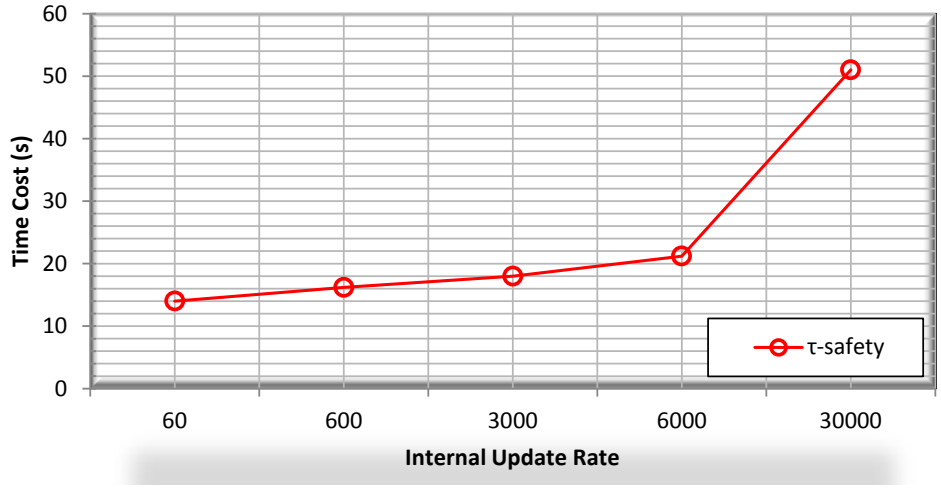


Figure 4.17: Cost by update rate

4.7 Synthesis

Chapter 3 has focused on a single release of data. In more complex scenarios, data is not released statically, but is published continuously and dynamically to serve numerous information needs. Thus sequential data publication remains a complex problem because it offers several leakage channels for the adversary. Among few works in the literature concerning sequential data publication, none of them caters the problem of arbitrary updates in the presence of chainability-oriented knowledge i.e., the event list, that tracks an individual through all the previous public release. In this Chapter, we highlighted that if an adversary has access to the event list of the individual(s), the privacy breach is imminent. We proposed an extension of m-invariance privacy model

which provides an effective solution to sequential data publication but to a limited capacity. We show that m-invariance is not achievable in the presence of arbitrary updates. In addition, it is vulnerable to privacy breaches in the presence of event list attacks (τ-attacks). We propose an extension to m-invariance, termed as τ-safety, which not only preserves the privacy of individuals but also helps generating better quality public release with minimum possible counterfeits.

5

Conclusion and perspectives

***Summary**: The time has come to draw a finishing line for this dissertation. Recent advancement in information storage and processing has induced an explosion of data promulgation. In this dissertation, we proposed state of the art algorithms for minimizing the risks pertaining to data dissemination. This Chapter provides an summary of this dissertation and also highlights possible future perspectives along with research directions.*

Contents

5.1 Introduction

Many public and private organizations collect and disseminate personal information for a variety of different purposes, including research and funding purposes. Disseminating such information without the privacy scare is an important problem. In such situations, the data publishers often face uncertainty - They need to protect the privacy of individuals on one hand and on the other hand, it is also extremely important to preserve the usefulness of the data for the researchers. In this dissertation, we mainly focus on crafting the notions of anonymity in various settings. We show that spatial indexes are extremely efficient for data publication tasks due to their ability to scale. An extensive empirical evaluation reveals that it is possible to disseminate high-quality data that follows meaningful notions of privacy. Furthermore, it is possible to do this efficiently for high dimensional very large data sets.
Nowadays, sequential data is being increasingly employed in a wide variety of applications and the publication of sequential data is of utmost importance for the betterment of these applications.

5.2 Synthesis

This thesis highlights the conceptual and practical issues to culminate the privacy risk originating from the promulgation of personal data. Privacy can be defined in many ways and the risk of privacy leak or information disclosure needs different modeling in different settings. Also there is a dire need of practical mechanisms/algorithms for the enforcement of these varied trends of privacy. This manuscript puts forward state-of-the-art algorithms that can facilitate data publishers to collect and promulgate personal information while alleviating the privacy risk along with improving data utility in different settings.
In this thesis, we examined various kinds of linking attacks in the different publishing scenarios of single release (Chapter 3) and sequential release (Chapter 4). Our contributions can be summed up as follows:

1. *Spatial indexes for data anonymization:* In first part of our work, we advocated the use of Bucketed Point Access Methods for Privacy-Preserving Data Publishing (PPDP) tasks. We reviewed the existing approaches based on multidimensional point partitioning and presented an almost comprehensive list of point access methods eligible for PPDP tasks. We argued for Nested Hyper-Rectangle-based BPAMs as the most promising structures to support PPDP. We then considered decomposable point and range queries against tabular representation of anonymous public releases, and we proposed a first attempt to answer such queries.
2. *Combining spatial index with clustering for anonymization:* Taking advantage of the above mentioned in-depth study, we chose BANG-clustering approach

for data anonymization since it combines clustering with point access method namely BANG-file. Specifically, we proposed BangA, which provides non hyper-rectangular blocks assigned to the equivalence classes of the public release. Hence, it achieves fast computation and scalability and very high quality thanks to its density-based clustering step. Moreover, BangA could incorporate background knowledge in the generalization process and the resulting public releases natively support orthogonal range queries. By virtue of its ability to scale and splitting strategy, BangA produces optimized equivalence classes as measured by popular quality metrics. Extensive experimentation confirms the supremacy of BangA over its counterparts specially R^+-tree based anonymization algorithm. Along with usual advantages, BangA could easily be molded for other more popular generalization models. Last but not the least, BangA could be used as-is for sequential data anonymization with continuous data releasing.

3. *Privacy model for dynamic data publishing :* The publication of micro-data for research purposes without the privacy scare is an important problem. Most of the work in this scenario caters only static data publication. In dynamic setting however, the data is modified and published multiple times. Dynamic data republication is naturally more complicated than static data publication as it allows certain attacks that are not applicable w.r.t single static publication. Such mechanisms are insufficient because they only guarantee privacy up to any single release. In Chapter 4, we present an anonymization framework for dynamic data publishing where data is published multiple times with a series of inserts/updates/deletes. We study the problem of anonymizing fully dynamic datasets in the presence of arbitrary updates. We show that *m*-invariance is not achievable in the presence of arbitrary updates specially in the presence of auxiliary knowledge and event list concerning the vital information about individuals. We propose an extension to *m*-invariance, termed as τ-safety, which not only preserves the privacy of individuals but also helps generating better quality public release with negligible counterfeits.
4. *Sequential data anonymization :* Sequential data is being increasingly employed in a wide variety of applications. Based on τ-safety privacy model, we proposed a bucketization-based algorithm for sequential data publication that not only guarantees the privacy offered by τ-safety privacy model but also provide substantial improvement in the utility along with better query accuracy than m-invariance bucketization based algorithm.

5.3 Perspectives

The research conducted in this dissertation can be extended in various directions. Below we analyze few interesting and challenging extensions to our work and outline possible directions towards them.

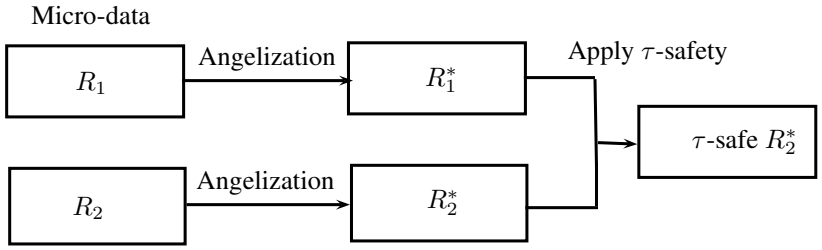


Figure 5.1: Proposition for τ-safe Angelization

- **τ-safety with Angelization**: Generalization is a popular technique for data sanitization. Tao et al. [97] presented a study indicating that generalization is subject to several drawbacks including information loss. The authors of [97] proposed a new anonymization technique, coined *Angelization*, which can be applied to any monotonic privacy model e.g., k-anonymity, ℓ-diversity etc.
  Angelization focuses on blurring the association between quasi-identifiers and sensitive attribute by releasing two separate tables, one for generalized QIs and other for sensitive values. Major steps of angelization are:
  1. Divide the input relation into batches such that each batch satisfies some privacy requirement e.g., for 2-diverse angelization, input relation is divided such that each batch is 2-diverse. This step results in so-called *Batch Table* (BT);
  2. Create buckets of size atleast k from the input relation where k is the parameter controlling the degree of protection;
  3. Generalize the records in each bucket and create Generalized Table (GT) from the generalized buckets;

  GT does not contain the sensitive attribute and the association between BT and GT is made by a column "Batch-ID" which serves as a foreign key in GT.
  Angelization provides same privacy guarantees as generalization (angelization actually subsumes generalization as a special case) but with much less data reconstruction error than generalization.
  In order to achieve even better utility with τ-safety, it may be interesting if generalization is replaced with angelization. Furthermore, angelization has not been extended for sequential data publication. Figure 5.1 depicts a proposition for achieving τ-safety via angelization. Though it presents a naive approach to achieve τ-safety using angelization, it seems to be applicable and if achieved, may substantially improve the utility of a τ-safe anonymized release.
- **τ-Safe BangA**: By virtue of its efficiency and effectiveness, BangA generalization mechanism is a first hand candidate for dynamic data publication. As explained before, BangA can be employed as it stands for sequential data publication but in

insert-only scenario in which there are only new records to manage. Since BangA can be extended to any other generalization model, thanks to its flexible splitting strategy, it can be extended to achieve τ-safety. The fundamental requirement for any algorithm to achieve τ-safety is to maintain δ-privacy by enforcing persistent invariance in each equivalence class on any publication timestamp. The naive way to achieve τ-safety through BangA is to put constraint on its splitting strategy. By following the same assumptions as for τ-safety, proposed steps for τ-safe BangA are as follows:

1. reconstruct the bang grid using previous microdata;
2. the most important step is to manage the deleted and re-inserted records. In worst case, this step may require extensive re-partitioning and may effect the overall efficiency as well. Deleted records can be managed by using either new records or by assigning the counterfeit records;
3. the records having internal update on *QI* values require meticulous partitioning since it may effect the overall utility at the end;

Finally, there exist several suitable partitioning schemes for τ-safe BangA such that the best one depends on how several kinds of updates are managed.

Bibliography

[1] N.R. Adam and J.C. Worthmann. Security-control methods for statistical databases: a comparative study. *ACM Computing Surveys (CSUR)*, 21(4):515–556, 1989. 22

[2] P.K. Agarwal. Range searching. *Handbook of discrete and computational geometry*, 2:809–838, 1997. 56

[3] C.C. Aggarwal. On k-anonymity and the curse of dimensionality. In *Proceedings of the 31st international conference on Very large data bases*, pages 901–909. VLDB Endowment, 2005. 51, 54

[4] C.C. Aggarwal. On randomization, public information and the curse of dimensionality. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 136–145. IEEE, 2007. 38, 41

[5] C.C. Aggarwal and P.S. Yu. A general survey of privacy-preserving data mining models and algorithms. *Privacy-preserving data mining*, pages 11–52, 2008. 22

[6] G. Aggarwal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu. Achieving anonymity via clustering. In *Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 153–162. ACM, 2006. 37, 38, 40

[7] Gagan Aggarwal, Tomas Feder, Krishnaram Kenthapadi, Rajeev Motwani, Rina Panigrahy, Dilys Thomas, and An Zhu. Approximation algorithms for k-anonymity. *Journal of Privacy Technology (JOPT)*, 2005. 38

[8] R.J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, pages 217–228. IEEE, 2005. 37, 41

[9] RJ Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, pages 217–228, 2005. 107

[10] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: the sulq framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 128–138. ACM, 2005. 48

[11] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 609–618, 2008. 23

[12] Y. Bu, A.W.C. Fu, R.C.W. Wong, L. Chen, and J. Li. Privacy preserving serial data publishing by role composition. *Proceedings of the VLDB Endowment*, 1(1):845–856, 2008. 33, 35, 85

[13] J.W. Byun, A. Kamra, E. Bertino, and N. Li. Efficient k-anonymization using clustering techniques. *Advances in Databases: Concepts, Systems and Applications*, pages 188–200, 2007. 38, 39, 40

[14] J.W. Byun, T. Li, E. Bertino, N. Li, and Y. Sohn. Privacy-preserving incremental data dissemination. *Journal of Computer Security*, 17(1):43–68, 2009. 32, 35

[15] J.W. Byun, Y. Sohn, E. Bertino, and N. Li. Secure anonymization for incremental datasets. *Secure Data Management*, pages 48–63, 2006. 31, 32, 35, 88

[16] Jianneng Cao, Panagiotis Karras, Panos Kalnis, and Kian-Lee Tan. Sabre: a sensitive attribute bucketization and redistribution framework for t-closeness. *The VLDB Journal*, 20(1):59–81, February 2011. 38

[17] D.L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981. 10

[18] S. Chawla, C. Dwork, F. McSherry, A. Smith, and H. Wee. Toward privacy in public databases. *Theory of Cryptography*, pages 363–385, 2005. 23

[19] B.C. Chen, D. Kifer, K. LeFevre, and A. Machanavajjhala. Privacy preserving data publishing. *Foundations and Trends in Databases*, 2(1-2):1–167, 2009. 42

[20] B.C. Chen, K. LeFevre, and R. Ramakrishnan. Privacy skyline: Privacy with multidimensional adversarial knowledge. *Proceedings of the 33rd international conference on Very large data bases*, pages 770–781, 2007. 26, 28, 29

[21] B.C. Chen, K. LeFevre, and R. Ramakrishnan. Adversarial-knowledge dimensions in data privacy. *The VLDB Journal*, 18(2):429–467, 2009. 28, 29

[22] Bee-Chung Chen, Daniel Kifer, Kristen LeFevre, and Ashwin Machanavajjhala. Privacy-preserving data publishing. *Found. Trends databases*, 2(1–2):1–167, January 2009. 24

[23] C.C. Chiu and C.Y. Tsai. A k-anonymity clustering method for effective data privacy preservation. *Advanced Data Mining and Applications*, pages 89–99, 2007. 38, 40

[24] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 41–50. IEEE, 1995. 40

[25] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M.Y. Zhu. Tools for privacy preserving distributed data mining. *ACM SIGKDD Explorations Newsletter*, 4(2):28–34, 2002. 21

[26] T. Dalenius. Towards a methodology for statistical disclosure control. *Statistik Tidskrift*, 15(429-444):2–1, 1977. 26

[27] M.L. Damiani, E. Bertino, and C. Silvestri. The probe framework for the personalized cloaking of private locations. *Transactions on Data Privacy*, 3(2):123–148, 2010. 40

[28] S. DE CAPITANI DI VIMERCATI, S. FORESTI, G. LIVRAGA, and P. SAMARATI. Data privacy: Definitions and techniques. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 20(06):793–817, 2012. 22, 44

[29] J. Domingo-Ferrer and J.M. Mateo-Sanz. Practical data-oriented microaggregation for statistical disclosure control. *Knowledge and Data Engineering, IEEE Transactions on*, 14(1):189–201, 2002. 38

[30] W. Du, Z. Teng, and Z. Zhu. Privacy-maxent: integrating background knowledge in privacy quantification. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 459–472. ACM, 2008. 29

[31] C. Dwork. Differential privacy. *Automata, languages and programming*, pages 1–12, 2006. 17, 23, 45

[32] C. Dwork. Differential privacy: A survey of results. *Theory and Applications of Models of Computation*, pages 1–19, 2008. 44

[33] C. Dwork. A firm foundation for private data analysis. *Communications of the ACM*, 54(1):86–95, 2011. 45

[34] C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 371–380. ACM, 2009. 45

[35] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. *Theory of Cryptography*, pages 265–284, 2006. 45, 46

[36] C. Dwork, F. McSherry, and K. Talwar. The price of privacy and the limits of lp decoding. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 85–94. ACM, 2007. 48

[37] C. Dwork and A. Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2):2, 2010. 46

[38] Cynthia Dwork, Moni Naor, Toniann Pitassi, Guy N Rothblum, and Sergey Yekhanin. Pan-private streaming algorithms. In *In Proceedings of ICS*, 2010. 47

[39] I.P. Fellegi. On the question of statistical confidentiality. *Journal of the American Statistical Association*, 67(337):7–18, 1972. 17

[40] M. Freeston. The BANG file: A new kind of grid file. *ACM SIGMOD Record*, 16(3):269, 1987. 51, 61, 64, 65, 66

[41] B. Fung, K. Wang, A.W.C. Fu, and J. Pei. Anonymity for continuous data publishing. In *Proceedings of the 11th international conference on Extending database technology: Advances in database technology*, pages 264–275. ACM, 2008. 33, 35

[42] BCM Fung, R. Chen, and PS Yu. Privacy-preserving data publishing: A survey on recent developments. *Computing*, 5(4):1–53, 2010. 17, 23, 30, 41, 133

[43] S.R. Ganta, S.P. Kasiviswanathan, and A. Smith. Composition attacks and auxiliary information in data privacy. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 265–273. ACM, 2008. 92, 93

[44] S. Garfinkel. *Database nation: the death of privacy in the 21st century*. O'Reilly Media, Incorporated, 2000. 9

[45] J. Gehrke. Models and methods for privacy-preserving data analysis and publishing. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 105–105. IEEE, 2006. 10

[46] J. Gehrke, M. Hay, E. Lui, and R. Pass. Crowd-blending privacy. *Advances in Cryptology–CRYPTO 2012*, pages 479–496, 2012. 18, 48, 49, 55, 78

[47] J. Gehrke, E. Lui, and R. Pass. Towards privacy for social networks: A zero-knowledge based definition of privacy. *Theory of Cryptography*, pages 432–449, 2011. 17

[48] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.L. Tan. Private queries in location based services: anonymizers are not necessary. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 121–132. ACM, 2008. 40

[49] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis. Fast data anonymization with low information loss. In *Proceedings of the 33rd international conference on Very large data bases*, pages 758–769. VLDB Endowment, 2007. 38, 39

[50] Samuel Greengard. Privacy matters. *Commun. ACM*, 51(9):17–18, 2008. 45

[51] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. *Proceedings of the 1st international conference on Mobile systems, applications and services*, pages 31–42, 2003. 40

[52] Y. He, S. Barman, and J.F. Naughton. Preventing equivalence attacks in updated, anonymized data. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, pages 529–540. IEEE, 2011. 35

[53] A.J.G. Hey, S. Tansley, and K.M. Tolle. *The fourth paradigm: data-intensive scientific discovery*. Microsoft Research Redmond, WA, 2009. 9

[54] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino. Private record matching using differential privacy. In *Proceedings of the 13th International Conference on Extending Database Technology*, pages 123–134. ACM, 2010. 45

[55] T. Iwuchukwu and J.F. Naughton. K-anonymization as spatial indexing: Toward scalable and incremental anonymization. In *Proceedings of the 33rd international conference on Very large data bases*, pages 746–757. VLDB Endowment, 2007. 15, 38, 39, 51, 54, 58, 69, 71, 72, 73, 74, 77, 79

[56] V.S. Iyengar. Transforming data to satisfy privacy constraints. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 279–288. ACM, 2002. 37

[57] M. Jakobsson, A. Juels, and R.L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *Proceedings of the 11th USENIX Security Symposium*, pages 339–353, 2002. 10

[58] Gunter Karjoth. E-privacy lectures http://www.zurich.ibm.com/ gka/eprivacy/. 21, 131

[59] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, page 228. ACM, 2006. 25, 41, 42, 107

[60] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 international conference on Management of data*, pages 193–204. ACM, 2011. 45, 48

[61] D. Kifer and A. Machanavajjhala. A rigorous and customizable framework for privacy. In *Proceedings of the 31st symposium on Principles of Database Systems*, pages 77–88. ACM, 2012. 45, 47, 48

[62] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In *Proceedings of the 18th international conference on World wide web*, pages 171–180. ACM, 2009. 45

[63] B. Krishnamachari, G. Ghinita, and P. Kalnis. Privacy-preserving publication of user locations in the proximity of sensitive sites. In *Scientific and Statistical Database Management*, pages 95–113. Springer, 2008. 40

[64] Jaewoo Lee and Chris Clifton. Differential identifiability. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, pages 1041–1049, New York, NY, USA, 2012. ACM. 47

[65] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, page 60. ACM, 2005. 37

[66] K. LeFevre, DJ DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 25–25, 2006. 37, 39, 73

[67] K.R. LeFevre. *Anonymity in data publishing and distribution*. PhD thesis, UNIVERSITY OF WISCONSIN, 2007. 20

[68] Feng Li and Shuigeng Zhou. Challenging more updates: Towards anonymous re-publication of fully dynamic datasets. *CoRR*, abs/0806.4703, 2008. 34, 35, 83, 85, 109

[69] Jiexing Li, Yufei Tao, and Xiaokui Xiao. Preservation of proximity privacy in publishing numerical sensitive data. *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 473–486, 2008. 23, 41

[70] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 106–115. IEEE, 2007. 14, 17, 23, 27, 28, 70

[71] N. Li, T. Li, and S. Venkatasubramanian. Closeness: A new privacy measure for data publishing. *Knowledge and Data Engineering, IEEE Transactions on*, 22(7):943–956, 2010. 26, 27

[72] N. Li, W.H. Qardaji, and D. Su. Provably private data anonymization: Or, k-anonymity meets differential privacy. *CoRR, abs/1101.2604*, 2011. 49, 55

[73] T. Li and N. Li. Injector: Mining background knowledge for data anonymization. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 446–455. IEEE, 2008. 38

[74] T. Li, N. Li, and J. Zhang. Modeling and integrating background knowledge in data anonymization. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*, pages 6–17. IEEE, 2009. 29

[75] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007. 14, 17, 23, 24, 25, 26, 27, 70

[76] D.J. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J.Y. Halpern. Worst-case background knowledge for privacy-preserving data publishing. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 126–135. IEEE, 2007. 28, 29

[77] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 223–228. ACM, 2004. 37, 39

[78] I. Mironov, O. Pandey, O. Reingold, and S. Vadhan. Computational differential privacy. *Advances in Cryptology-CRYPTO 2009*, pages 126–142, 2009. 46

[79] Noman Mohammed, Rui Chen, Benjamin C.M. Fung, and Philip S. Yu. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 493–501, New York, NY, USA, 2011. ACM. 46

[80] M.F. Mokbel, C.Y. Chow, and W.G. Aref. The new casper: query processing for location services without compromising privacy. *Proceedings of the 32nd international conference on Very large data bases*, pages 763–774, 2006. 40

[81] A. Narayanan and V. Shmatikov. Myths and fallacies of personally identifiable information. *Communications of the ACM*, 53(6):24–26, 2010. 45

[82] 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*, pages 665–676. ACM, 2007. 23, 37

[83] M.E. Nergiz, C. Clifton, and A.E. Nergiz. Multirelational k-anonymity. *Knowledge and Data Engineering, IEEE Transactions on*, 21(8):1104–1117, 2009. 23

[84] J. Nievergelt, H. Hinterberger, and K.C. Sevcik. The grid file: An adaptable, symmetric multikey file structure. *ACM Transactions on Database Systems (TODS)*, 9(1):38–71, 1984. 51

[85] AH Pilevar and M. Sukumar. Gchl: A grid-clustering algorithm for high-dimensional very large spatial data bases. *Pattern recognition letters*, 26(7):999–1010, 2005. 40

[86] W. Qardaji and N. Li. Recursive partitioning and summarization: a practical framework for differentially private data publishing. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pages 38–39. ACM, 2012. 79

[87] R. Ramakrishnan, J. Gehrke, and J. Gehrke. *Database management systems*, volume 3. McGraw-Hill, 2003. 11

[88] V. Rastogi, M. Hay, G. Miklau, and D. Suciu. Relationship privacy: output perturbation for queries with joins. In *Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 107–116. ACM, 2009. 45

[89] V. Rastogi, D. Suciu, and S. Hong. The boundary between privacy and utility in data publishing. In *Proceedings of the 33rd international conference on Very large data bases*, pages 531–542. VLDB Endowment, 2007. 10, 23, 26, 38, 41

[90] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *PROCEEDINGS OF THE ACM SIGACT SIGMOD SIGART SYMPOSIUM ON PRINCIPLES OF DATABASE SYSTEMS*, volume 17, pages 188–188. ASSOCIATION FOR COMPUTING MACHINERY, 1998. 37, 54

[91] H. Samet. *Foundations of multidimensional and metric data structures*. Morgan Kaufmann, 2006. 57, 63, 66

[92] E. Schikuta. Grid-clustering: An efficient hierarchical clustering method for very large data sets. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 2, pages 101–105. IEEE, 1996. 51, 68

[93] E. Schikuta and M. Erhart. The BANG-clustering system: grid-based data analysis. *Advances in Intelligent Data Analysis Reasoning about Data*, pages 513–524, 1997. 51, 68

[94] Xiaoxun Sun, Min Li, and Hua Wang. A family of enhanced (l,α)-diversity models for privacy preserving data publishing. *Future Gener. Comput. Syst.*, 27(3):348–356, March 2011. 27

[95] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems*, 10(5):571–588, 2002. 17, 23, 24, 28, 35, 37, 39, 54, 71

[96] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty Fuzziness and Knowledge Based Systems*, 10(5):557–570, 2002. 12, 14, 17, 23, 24, 28, 31, 89

[97] Y. Tao, H. Chen, X. Xiao, S. Zhou, and D. Zhang. Angel: Enhancing the utility of generalization for privacy preserving publication. *Knowledge and Data Engineering, IEEE Transactions on*, 21(7):1073–1087, 2009. 119

[98] Y. Tao, X. Xiao, J. Li, and D. Zhang. On anti-corruption privacy preserving publication. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 725–734. IEEE, 2008. 38, 41

[99] TechCrunch. Aol proudly releases massive amounts of private data, 2006. 10

[100] Hongwei Tian and Weining Zhang. Extending l-diversity to generalize sensitive data. *Data Knowl. Eng.*, 70(1):101–126, January 2011. 36

[101] T.M. Truta and B. Vinay. Privacy protection: p-sensitive k-anonymity property. *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on*, pages 94–94, 2006. 27

[102] 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*, 33(1):50–57, 2004. 22

[103] K. Wang, B.C.M. Fung, and P.S. Yu. Handicapping attacker's confidence: an alternative to k-anonymization. *Knowledge and Information Systems*, 11(3):345–368, 2007. 23

[104] Ke Wang and Benjamin C. M. Fung. Anonymizing sequential releases. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 414–423, 2006. 23

[105] W. Wang, J. Yang, R. Muntz, et al. Sting: A statistical information grid approach to spatial data mining. In *Proceedings of the International Conference on Very Large Data Bases*, pages 186–195. INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS (IEEE), 1997. 40

[106] S.L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, pages 63–69, 1965. 10

[107] L. Willenborg and T. De Waal. *Statistical disclosure control in practice*, volume 111. Springer, 1996. 17

[108] R.C.W. Wong, A.W.C. Fu, K. Wang, and J. Pei. Minimality attack in privacy preserving data publishing. In *Proceedings of the 33rd international conference on Very large data bases*, pages 543–554. VLDB Endowment, 2007. 30

[109] R.C.W. Wong, J. Li, A.W.C. Fu, and K. Wang. (α, k)-anonymity: an enhanced k-anonymity model for privacy preserving data publishing. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 754–759, 2006. 23, 27

[110] X. Xiao and Y. Tao. Anatomy: Simple and effective privacy preservation. In *Proceedings of the 32nd international conference on Very large data bases*, pages 139–150. VLDB Endowment, 2006. 38

[111] X. Xiao and Y. Tao. Personalized privacy preservation. *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 229–240, 2006. 23

[112] X. Xiao and Y. Tao. M-invariance: towards privacy preserving re-publication of dynamic datasets. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, page 700. ACM, 2007. 16, 17, 32, 33, 35, 85, 91, 92, 94, 95, 97, 98, 99, 103, 106, 107, 109

[113] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. *Knowledge and Data Engineering, IEEE Transactions on*, 23(8):1200–1214, 2011. 46

[114] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A.W.C. Fu. Utility-based anonymization using local recoding. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 790. ACM, 2006. 41, 107

[115] Yin Yang, Zhenjie Zhang, Gerome Miklau, Marianne Winslett, and Xiaokui Xiao. Differential privacy in data publication and analysis. *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 601–606, 2012. 45, 48

[116] Zhiqiang Yang, Sheng Zhong, and Rebecca N. Wright. Anonymity-preserving data collection. In *In KDD âĂŹ05: Proc. of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 334–343, 2005. 10

[117] Q. Zhang, N. Koudas, D. Srivastava, and T. Yu. Aggregate query answering on anonymized tables. *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 116–125, 2007. 23, 38

List of Figures

List of Tables

List of Algorithms

l'université nantes angers le mans
PÔLE DE RECHERCHE ET D'ENSEIGNEMENT SUPÉRIEUR

UNIVERSITÉ DE NANTES

Thèse de Doctorat

Adeel Anjum

Towards Privacy-Preserving Publication of Continuous and Dynamic Data

Spatial Indexing and Bucketization Approaches

Résumé

La publication de données soucieuse du respect de la vie privée est au cœur des préoccupations des organisations qui souhaitent publier leurs données. Un nombre croissant d'entreprises et d'organismes collectent et publient des données à caractère personnel pour diverses raisons (études démographiques, recherche médicale,...). Selon ces cas, celui qui publie les données fait face au dilemme suivant : *comment permettre à un tiers l'analyse de ces données tout en évitant de divulguer des informations trop sensibles, relatives aux individus concernés?* L'enjeu est donc la capacité à publier des jeux de données en maîtrisant ce risque de divulgation, c.a.d. de traiter l'opposition entre deux critères : d'un côté, on souhaite garantir la préservation de la confidentialité sur des données personnelles et, d'autre part, on souhaite préserver au maximum l'utilité du jeu de données pour ceux qui l'exploiteraient (notamment, des chercheurs). Dans ce travail, nous cherchons d'abord à élaborer plusieurs notions d'anonymisation des données selon plusieurs contextes. Nous montrons que les index spatiaux sont extrêmement efficaces dans le cadre de la publication de données, en raison de leur capacité à passer à l'échelle. Une évaluation empirique approfondie révèle qu'il est possible de diffuser des données de grande qualité et préservant un certain niveau de confidentialité dans les données. Il est de plus possible de traiter efficacement de très grands jeux de données en grandes dimensions et cette méthode peut être étendue à un niveau de confidentialité plus fort (differential privacy). Par ailleurs, la publication séquentielle de données (mise à jour du jeu de données) est cruciale dans un grand nombre d'applications. Nous proposons une technique menant à bien cette tâche, garantissant à la fois une forte confidentialité des données et une très bonne préservation de leur utilité.

Mots clés

Publication de données qui préserve la vie privée, indexation spatials, bucketization, k-anonymat, differential privacy

Abstract

Privacy-Preserving Data Publishing (PPDP) has become a critical issue for companies and organizations that would release their data. Many organizations collect and distribute personal data for a variety of different purposes, including demographic and public health research. In these situations, the data distributor is often faced with a dilemma: *how to publish this personal data for analysis purposes without endangering the privacy of the concerned individuals*? Disseminating such information without the privacy scare is an important problem. On one hand, the data publishers need to protect the privacy of individuals and on the other hand, it is also extremely important to preserve the usefulness of the data for the researchers. In this dissertation, we mainly focus on crafting the notions of privacy in various settings. We show that spatial indexes are extremely efficient for data publication tasks due to their ability to scale up. An extensive empirical evaluation reveals that it is possible to disseminate high-quality data that follows meaningful notions of privacy. Furthermore, it is possible to do this efficiently for high dimensional very large data sets and this approach can be extended to stronger notions of privacy e.g., differential privacy. Also, sequential data is being increasingly employed in a wide variety of applications and the publication of sequential data is of utmost importance for the betterment of these applications. We provide a bucketization-based approach to achieve a stronger privacy guarantee along with higher utility of final release.

Key Words

Data Privacy, Spatial Indexing ,k-anonymity, Bucketization, Differential Privacy