

Butterfly: Privacy Preserving Publishing on Multiple Quasi-Identifiers*

Technical Report TR 2008-18

School of Computing Science, Simon Fraser University

Jian Pei
Simon Fraser University
jpei@cs.sfu.ca

Yufei Tao Jiexing Li Xiaokui Xiao
The Chinese University of Hong Kong
{taoyf, jxli, xkxiao}@cse.cuhk.edu.hk

Abstract

Recently, privacy preserving data publishing has attracted significant interest in research. Most of the existing studies focus on only the situations where the data in question is published using one quasi-identifier. However, in a few important applications, a practical demand is to publish a data set on multiple quasi-identifiers for multiple users simultaneously, which poses several challenges. How can we generate one anonymized version of the data so that the privacy preservation requirement like k -anonymity is satisfied for all users? Moreover, how can we reduce the information loss as much as possible while the privacy preservation requirements are met?

In this paper, we identify and tackle the novel problem of privacy preserving publishing on multiple quasi-identifiers. A naïve solution of respectively publishing multiple versions for different quasi-identifiers unfortunately suffers from the possibility that those releases can be joined to intrude the privacy. Interestingly, we show that it is possible to generate only one anonymized table to satisfy the k -anonymity on all quasi-identifiers for all users without significant information loss. We systematically develop an effective method for privacy preserving publishing for multiple users, and report an empirical study using real data to verify the feasibility and the effectiveness of our method.

1 Introduction

Micro-data, which is actual data collected instead of statistical summaries, is particularly useful in many data analysis applications. However, privacy is a serious concern in sharing micro-data. To address the concern, a large amount of research has been focused on privacy-preserving data publishing. Several models such as k -anonymity [35, 37, 38] and l -diversity [27] are proposed, and a few effective and efficient methods such as [23, 24, 46, 45, 8] are designed.

Most of the existing studies focus on only the situations where a micro-data set in question is published using one quasi-identifier. However, in a few important applications, a practical demand

*This work was mainly done when the first author visited the Chinese University of Hong Kong in the summer of 2007. An extended abstract appears in *Proceedings of the Twenty-fifth IEEE International Conference on Data Engineering (ICDE'09)*, March 29 - April 4, 2009, Shanghai, China.

Table 1: A set of traffic accident records.

Occupation	age	vehicle	postcode	faulty
Dentist	30	Red Truck	31043	No
Family doctor	30	White Sedan	31043	Yes
Banker	30	Green Sedan	31043	No
Mortgage broker	30	Black Truck	31043	No

is to publish a data set simultaneously on multiple quasi-identifiers for users carrying different background knowledge.

For example, the traffic management board of a region collects records of road accidents for research and analysis. Such records are interesting to auto insurance companies which can use such information to analyze the risk of their business and define their policies accordingly. Simultaneously, such traffic accident records are also interesting to the human resource department in the government since they can be used to analyze the impact of accidents on working groups. Therefore, the traffic management board may want to release the data to multiple users. Importantly, different users may carry different background knowledge. For example, an auto insurance company has the vehicle registration records of its customers, and the human resource department has the resident records in the region. Thus, the traffic management board needs to protect the privacy against attacks using different background knowledge.

Releasing a data set to multiple users leads to serious concerns on privacy preservation. Even though we ensure that the release to each user satisfies the corresponding privacy-preservation requirement such as k -anonymity, privacy still can be disclosed if collusion happens.

Example 1 (Multiple releases are unsafe). *In the traffic accident database, suppose each record has five attributes, namely occupation, age, vehicle-type, postcode, and faulty. Consider the tuples in Table 1.*

The data set is to be released respectively to an auto insurance company and the human resource department of the government. The auto insurance company may join the traffic accident records with the vehicle registration records on attributes age, vehicle-type, and postcode to find out whether its customers were at fault in some accidents. Typically, the company does not have the occupation information of its customers, as such information is not required in applying for auto insurance. Therefore, to protect privacy, the traffic management board has to anonymize the traffic accident records on attributes age, vehicle-type, and postcode before the data can be released to the auto insurance company. Suppose 2-anonymity is required. Table 2 shows a 2-anonymous release of the records with respect to quasi-identifier (age, vehicle-type, postcode).

Simultaneously, the human resource department may join the traffic accident records with the resident records on attributes occupation, age, and postcode to find out which residents were faulty in some accidents. Therefore, to protect privacy, the traffic management board needs to anonymize the traffic accident records on attributes occupation, age, and postcode. Note that vehicle-type is not part of the anonymization, because the human resource department typically does not have information about residents' vehicle types. Again, suppose 2-anonymity is required.

Table 2: A 2-anonymous release of the traffic accident records in Table 1 with respect to quasi-identifier (**age**, **vehicle-type**, and **postcode**).

Occupation	age	vehicle	postcode	faulty
Dentist	30	Truck	31043	No
Family doctor	30	Sedan	31043	Yes
Banker	30	Sedan	31043	No
Mortgage broker	30	Truck	31043	No

Table 3: A 2-anonymous release of the traffic accident records in Table 1 with respect to quasi-identifier (**occupation**, **age**, and **postcode**).

Occupation	age	vehicle	postcode	faulty
Medical	30	Red Truck	31043	No
Medical	30	White Sedan	31043	Yes
Finance	30	Green Sedan	31043	No
Finance	30	Black Truck	31043	No

Table 3 shows a 2-anonymous release of the records with respect to quasi-identifier (**occupation**, **age**, **postcode**).

Nevertheless, the two 2-anonymous tables are insufficient to protect privacy when collusion happens. Suppose an adversary obtains both releases in Tables 2 and 3. By comparing the two tables, the adversary immediately knows that a family doctor of age 30 driving a white Sedan living in area 31043 was faulty in an accident. The victim may be easily re-identified by both the vehicle registration record and the human resource resident record. ■

As shown in Example 1, generating a k -anonymous release for each user does not automatically guarantee the k -anonymity for all data entries when collusion among users happens. In Example 1, the loophole is serious since the attack can be made even without sharing any background knowledge (i.e., vehicle registration records and resident records) from the two users. Instead, any adversary obtaining both releases can intrude the privacy.

A naïve method is to anonymize the table on the union of the quasi-identifiers of all users. In the context of Example 1, we can anonymize tuples on the union quasi-identifier (**occupation**, **age**, **vehicle-type**, **postcode**). However, anonymization on the union quasi-identifier, which may contain much more attributes than any individual quasi-identifier, may lead to heavy information loss.

Generally, privacy preserving publishing for multiple users poses several technical challenges in both data publishing models and anonymization methods. How can we generate one release of the data so that the privacy preservation requirement like k -anonymity is satisfied for all users? Moreover, how can we reduce information loss as much as possible while the privacy preservation requirements are met?

In this paper, we tackle the problem and make the following contributions. First, we identify the novel problem of privacy preserving publishing on multiple quasi-identifiers. Second, we indicate

that it is possible to generate only one anonymized table to satisfy the k -anonymity on all quasi-identifiers for all users without significant information loss. Our method is substantially better than the naïve method which conducts anonymization using the union quasi-identifier. Third, we systematically develop an effective method to generate such an anonymized table for multiple users. Last, we report an empirical study using real data sets to verify the feasibility and the effectiveness of our method.

The rest of the paper is organized as follows. In Section 2, we define the problem of privacy preserving publishing on multiple quasi-identifiers, and present an important observation. We review the related work and compare with our study in Section 3. An effective solution for the basic case of 2 quasi-identifiers is developed in Section 4. In Section 5, we discuss how the basic case can be extended to the general case where more than 2 quasi-identifiers exist and how to protect privacy when collusion among users may happen. An empirical study is reported in Section 6. We discuss the possible extensions and conclude the paper in Section 7.

2 Problem Definition

In this section, we first recall the essential notions of privacy preserving publishing and the k -anonymity mechanism. Then, we define the problem of privacy preserving publishing on multiple quasi-identifiers, and present an important observation.

2.1 K-Anonymity

Consider a micro-data table $T = (A_1, \dots, A_n)$, where a record in the table represents the data for one individual. An *external table* $E = (B_1, \dots, B_m)$ which also contains records of individuals can be used to model the background knowledge of a user. A *re-identification attack* to the privacy of individuals in table T is that the user can join tables T and E on the common attributes of the two tables so that individuals in T may be re-identified. The set of common attributes between tables T and E , i.e., $S = T \cap E$, is called the *quasi-identifier* (or *QID* for short) with respect to the re-identification attack using E .

To protect privacy against re-identification attacks, the owner of T may change the values of tuples in T on attributes in QID S so that at least k tuples look the same on QID S . Then, each individual cannot be re-identified with a probability over $\frac{1}{k}$. Technically, an *anonymization* is a function f on T such that for each tuple $t \in T$, $f(t)$ is a tuple where some values of t may be changed.

Suppose a table T is anonymized as T' , i.e., $T' = \{f(t) | t \in T\}$. For a tuple $t \in T'$, the set of tuples $t' \in T'$ which have the same values as t on all attributes in S form an *equivalence class* (*EC* for short) on S , i.e., $E(t) = \{t' \in T' | \forall A \in S, t'[A] = t[A]\}$. Clearly, $t \in E(t)$. T' is *k -anonymous* ($k > 0$) on QID S if for each tuple $t \in T'$, $\|E(t)\| \geq k$.

A general representation of anonymized tuples [48, 18] is to generalize an attribute value to a range. For example, if we want to make k tuples into an EC and the values of those tuples on attribute age range from 20 to 30, we can generalize the values to a range [20, 30]. Apparently, the larger the range, the more information loss is introduced by the anonymization.

Some methods have been developed to measure the information loss in anonymization. For

example, the *discernability measure* [8] assigns to each tuple t a penalty based on the size of the equivalence class that t is generalized, i.e., the number of tuples equivalent to t on the quasi-identifier. That is, $C_{DM} = \sum_{E \in \text{group-by } S \text{ on quasi-identifier } E} \|E\|^2$.

In this paper, we adopt the uncertainty penalty measure of information loss which is also used in [48, 18].

Definition 1 (Uncertainty penalty). *Suppose table T is anonymized to T' . In the domain of each attribute in T , suppose there exists a global order on all possible values in the domain. If a tuple t in T' has range $[x, y]$ on attribute A , then the **uncertainty penalty** in t on A is*

$$\text{loss}_A(t) = \frac{\|y - x\|}{\|A\|},$$

where

$$\|A\| = \max_{t' \in T} \{t'[A]\} - \min_{t' \in T} \{t'[A]\}$$

is the range of attribute A in T . For tuple t , the **uncertainty penalty** in t is $\text{loss}(t) = \sum_{A \in S} \text{loss}_A(t)$, where S is the QID.

The **uncertainty penalty** in T' is $\sum_{t \in T'} \text{loss}(t)$. ■

Given a table T and a QID S , the *problem of k -anonymization* is to generalize T to a k -anonymous table T' such that the information loss is minimized. The k -anonymization problem has been shown NP-hard [29] in general. Several heuristic methods have been proposed. We will review some representative ones in Section 3.

2.2 Privacy Preserving Publishing for Multiple Users

In this paper, we consider the situation where a micro-data table $T = (A_1, \dots, A_n)$ needs to be anonymized and released for a group of users U_1, \dots, U_m . For each user U_i ($1 \leq i \leq m$), we assume that U_i can issue re-identification attacks using some background knowledge. Technically, we assume that the background knowledge that U_i has can be used to attack the privacy of individuals in T on a quasi-identifier $S_i \subseteq T$. Thus, we need to make sure that the release for U_i is k -anonymous with respect to S_i .

Due to the problem in generating different releases for different users as analyzed in Example 1, we are interested in generating only one anonymized version T' such that T' is k -anonymous with respect to all S_i ($1 \leq i \leq m$). The problem of *privacy preserving publishing for multiple users* is to generate the k -anonymous table T' so that the k -anonymity requirement for each user is satisfied, and the information loss is as small as possible.

A naïve approach is to generate a table T' such that T' is k -anonymous with respect to the *union QID* $S = \cup_{i=1}^m S_i$. Apparently, if T' is k -anonymous with respect to S , T' is also k -anonymous with respect to any individual S_i . We call this method the *union QID method*.

As analyzed in Section 1, the union QID may contain many more attributes than any individual QID, and thus the union QID method may introduce substantial information loss. Interestingly, we can show that the union QID may not be necessary to ensure k -anonymity with respect to all individual QIDs.

Table 4: An example of constructing R_1 in the proof of Theorem 1.

A	B	C
a_1	b_1	c_1
a_1	b_1	c_2
a_1	b_2	c_1
a_1	b_2	c_2
a_2	b_1	c_1
a_2	b_1	c_2
a_2	b_2	c_1
a_2	b_2	c_2

Theorem 1 (Union QID). *Let $\mathcal{R} = (A_1, \dots, A_n)$ be a schema of micro-data where the domain of each attribute has the cardinality of at least 2. Let S_1, \dots, S_m be m QIDs and $S = \cup_{i=1}^m S_i$ be the union QID. If there does not exist S_{i_0} ($1 \leq i_0 \leq m$) such that $S_{i_0} = S$, then there exists a table R on \mathcal{R} such that R is k -anonymous with respect to every S_i ($1 \leq i \leq m$) but R is not k -anonymous with respect to S .*

Proof. Since the domain of each attribute has the cardinality of at least 2, for each attribute A_j , we can choose two distinct values $v_{j,1}, v_{j,2} \in A_j$. Since the attributes not in S do not matter for anonymization, without loss of generality, let us assume $S = \mathcal{R}$.

Consider table $R_1 = \{v_{1,1}, v_{1,2}\} \times \dots \times \{v_{m,1}, v_{m,2}\}$ which contains 2^m tuples. Since there does not exist S_{i_0} ($1 \leq i_0 \leq m$) such that $S_{i_0} = S$, there does not exist S_{i_0} such that $S_{i_0} = \mathcal{R}$. Thus, table R_1 is 2-anonymous with respect to every S_i , but is not 2-anonymous with respect to S .

For example, Table 4 shows a table constructed as such which is 2-anonymous with respect to any proper subset of ABC , but is not 2-anonymous with respect to ABC .

To construct a table R in the theorem, we copy $\lceil \frac{k}{2} \rceil$ times every tuple $t \in R_1$ into R . Then, R is k -anonymous with respect to every S_i , but is not with respect to S . \square

Theorem 1 presents an interesting observation: when the union QID is not a QID for any user, k -anonymity with respect to the union QID is not necessary to achieve k -anonymity with respect to the QID of every user. Clearly, avoiding anonymization using the union QID can reduce information loss substantially. Then, the question is how we can avoid anonymization with respect to the union QID, which is the topic of Sections 4 and 5.

3 Related Work

In this section, we first review the previous algorithms for generalizing microdata. Then, we discuss the alternative privacy models. Finally, we briefly survey research on other topics of privacy protection.

3.1 Generalization Algorithms

Since the introduction of k -anonymity by Samarati and Sweeney [35, 37], numerous algorithms have been developed to compute k -anonymous tables with small information loss. An early attempt is

by Iyengar [20], who suggests a genetic paradigm to explore the vast search space, and return the best solution among those examined within a time limit. Bayardo and Agrawal [8] propose a method that guarantees the optimal solution. The method utilizes a *set-enumeration tree*, which organizes all the possible solutions into a hierarchy, and effectively prunes the branches of the tree that cannot contain a better solution (than the best one currently found). LeFevre *et al.* [23] devise a faster approach that finds the optimal generalization satisfying certain constraints (called *full-domain generalization*), by leveraging techniques reminiscent of mining frequent item sets.

Apart from the above algorithms, considerable efforts have been paid to computing reasonably good (although not optimal) generalization by fast heuristics. There are two primary directions for designing effective heuristics. The first one treats the generalization problem as an instance of clustering, where each equivalence class (EC) is a cluster, and the quality of clustering is gauged by the information loss. Fung *et al.* [17] advocate a top-down approach, where initially a single gigantic cluster covers the entire dataset, and it is gradually broken into smaller clusters to achieve better clustering. Xu *et al.* [48] propose a bottom-up method, where each micro-cluster includes tuples with exactly the same QID values, and micro-clusters are recursively grouped together into larger clusters to satisfy k -anonymity. Ghinita *et al.* [18] present an interesting solution that forms clusters according to a space filling curve.

The second direction analogizes generalization to building a multi-dimensional index, where each leaf node is adopted as an EC directly. The research issue here is to adapt the index construction routines to ensure that each leaf node should have at least k entries. Iwuchukwu and Naughton [19] present a method based on R^+ -trees [36], while LeFevre *et al.* [24] give an algorithm, *Mondrian*, based on kd-trees, which is later extended in [25] to *workload-aware anonymization*. Next, we elaborate the details of *Mondrian*, because they are useful to our discussion in Section 4.2.2.

Let us consider a simple example, where a microdata set T has nine tuples, and the QID of T has two attributes x and y . Thus, each tuple of T can be regarded as a two-dimensional point in the x - y plane. Figure 1(a) shows the point representations p_1, p_2, \dots, p_9 of the tuples in T . Suppose that the goal is to achieve 2-anonymity.

Mondrian repetitively carries out the following split operation: given a set E of points and a split axis, divide E into E' and E'' , where E' (E'') includes the points of E whose coordinates on the split axis is at most (larger than) the median coordinate in E . To illustrate, let the first split be performed on the x -dimension. The median coordinate of T is 4 (i.e., the $\lfloor \|T\|/2 \rfloor = 4$ -th smallest¹ x -coordinate in T); hence, the split divides T into two sets E_1 and E_2 . Figure 1(b) represents E_1 and E_2 using the minimum bounding rectangles (MBR) of their points, respectively. Next, E_1 (E_2) is split on the y -dimension, producing E_3, E_4 (E_5, E_6), as shown in Figure 1(c).

In general, the split axis is chosen in a round-robin fashion: if an MBR E results from a split on x , then the next split happens on y , and vice versa. The split continues until E contains between k and $2k - 1$ points (in this case, no more split is possible, because another split necessarily spawns an MBR covering less than k points). The final E becomes an EC, and its MBR is taken as its generalized form. In Figure 1(c), since no more splits are possible on E_1, E_2, E_3 and E_4 , they constitute the final ECs of the 2-anonymous generalization of T . Figure 1(d) demonstrates a binary tree recording the *split history*. For example, E_1 is the parent of E_3 and E_4 , because they result from the split of E_1 .

¹Likewise, we could also decide the median as the $\lceil \|T\|/2 \rceil$ -th smallest.

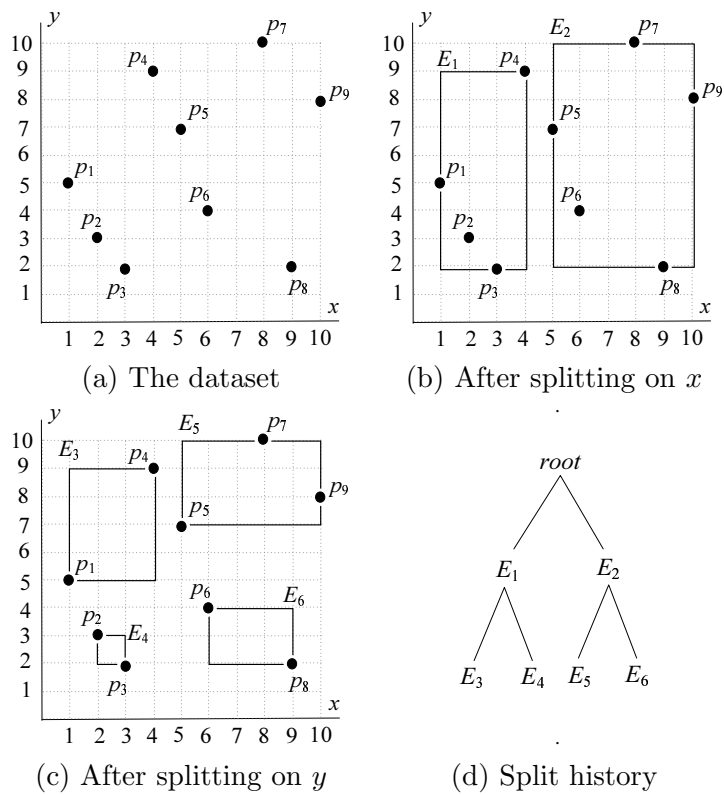


Figure 1: Illustration of *Mondrian*

The algorithms mentioned earlier work well on practical datasets, but do not have attractive asymptotical performance in the worst case. This motivates studies on the theoretical aspects of k -anonymity. Meyerson and Williams [29] are the first to prove the NP-hardness of optimal k -anonymous generalization, and give an $O(k \log k)$ -approximation algorithm. Aggarwal *et al.* [4] reduce the approximation ratio to $O(k)$, which is further improved to $O(\log k)$ by Park and Shim [32]. Unlike these solutions whose approximation ratios are functions of k , Du *et al.* [14] present a method having a ratio $O(d)$, where d is the number of attributes in the QID. Aggarwal *et al.* [5] develop constant approximation algorithms. Wong *et al.* [43] observe that, in general, a deterministic algorithm may suffer from *minimality attacks*, which can be avoided with certain randomization.

3.2 Privacy Models

Besides k -anonymity, several other privacy models have been developed, explicitly assuming the existence of a *sensitive attribute* (SA), which is not part of the QID. In particular, l -diversity [27] requires each EC to include at least l “well-represented” SA-values. (α, k) -anonymity [44] is a combination of l -diversity and k -anonymity. t -closeness [26] demands that the SA distribution of each EC should deviate from that of the entire table by at most a certain threshold t . *Personalization* [46] allows each individual to specify the level of protection for her/his own data. Focusing on a numeric SA, (k, e) -anonymity [49] demands that, in each EC, there are at least k tuples, and the maximum SA value exceeds the minimum by at least e . (c, k) -safety [28] deals with complex forms of background knowledge, and *privacy-skyline* [12] balances the protection against different types of background knowledge. Unlike the above principles, δ -presence [31] does not deal with any SA; it aims at preventing an adversary from knowing that an individual has records in the microdata.

In this work, we focus on k -anonymity for several reasons. First, k -anonymous publishing is compliant with governmental privacy regulations [1] such as HIPAA (Health Insurance Portability and Accountability Act) and European Union Data Directive. Second, k -anonymity can be applied in circumstances where it is difficult to determine which attributes are “sensitive”, and/or sensitive attributes are also a part of the QID. Third, the simplicity of k -anonymity permits us to explain, in an approachable context, the complex concepts proposed in this paper. Nevertheless, extending our techniques to other privacy models is an interesting direction for future work.

3.3 Other Research on Privacy Preserving Publishing

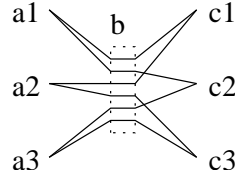
The literature of data anonymity has expanded considerably in the past few years, and a thorough survey falls out of the scope of this paper. Instead, we indicate here several major topics in this area, and provide references that serve as entry points to further reading.

Kifer and Gehrke [22] study *marginal publication*, where the objective is to publish the anonymized versions of various projections of the microdata onto different subsets of attributes. Wang and Fung [41] tackle *sequential publication* where new projections are released after the old ones have gone public. Xiao and Tao [47] address *re-publication*, which aims to release new versions of the microdata after it has been updated with insertions and deletions and secure privacy against a persistent adversary that may have audited all the versions released in the past.

In addition to generalization, privacy protection in data publication can also be achieved with other methodologies such as *anatomy* [18, 45, 49], *condensation* [3] and *perturbation* [7, 16, 33].

A	B	C
a_1	b	c_1
a_1	b	c_2
a_2	b	c_1
a_2	b	c_3
a_3	b	c_2
a_3	b	c_3

(a) A 2-anonymous table



(b) Visualization of a butterfly.

Figure 2: A butterfly in a 2-anonymous table.

Finally, it is worth mentioning that data anonymity is a general concern in several other applications as well, including association rule hiding [2, 13, 40, 42], multi-party computation [21, 34, 39], privacy-aware query processing [11, 15, 30], and access control [6, 9, 10].

Summary. Despite the bulk of literature on privacy preservation, we are not aware of any work on generalization in the presence of multiple QIDs. The next section settles this problem with a novel technique called *butterfly*.

4 The Butterfly Method

In this section, we discuss the basic case where there are 2 users, U_1 and U_2 , using QIDs S_1 and S_2 , respectively. We need to anonymize a table $T = (A_1, \dots, A_n)$ to a table T' such that T' is k -anonymous with respect to QIDs S_1 and S_2 . Let $S = S_1 \cup S_2$. We consider the general case where $S \neq S_1$ and $S \neq S_2$.

We first identify an essential structure called butterfly that is essential in every anonymization for two QIDs. Then, we develop an efficient algorithm for anonymization.

4.1 The Butterfly Structure

Example 2 (Butterfly). *Table $T = (A, B, C)$ in Figure 2(a) is 2-anonymous with respect to $S_1 = AB$ and $S_2 = BC$, but not 2-anonymous with respect to $S = ABC$.*

On QIDs $S_1 = AB$ and $S_2 = BC$, respectively, the tuples form ECs such that each EC is of size 2. Interestingly, the tuples share the same values on B , the common attribute between S_1 and S_2 . This sharing is critical to achieve tuples that do not need to form ECs on ABC but still can satisfy the k -anonymity requirements on S_1 and S_2 .

Figure 2(b) visualizes the tuples. A tuple is a line connecting the values on attributes A , B , and C . It looks like a butterfly: the tuples sharing the same value on B which is the body of the butterfly. Different values on A and C form “wings” of the butterfly. Generally, a butterfly structure in our study may have multiple “wings”, but a biological butterfly has only 4 wings. ■

Based on the observation in Example 2, we identify butterfly, the essential structure in anonymizing tables for multiple QIDs.

Definition 2 (Butterfly). *Given a table T , and two QIDs S_1 and S_2 on T such that $S_1 \cup S_2 \neq S_1$ and $S_1 \cup S_2 \neq S_2$. A set of tuples $P \subseteq T$ is a k -butterfly with respect to S_1 and S_2 if*

1. P can be partitioned into ECs on $S_1 - S_2$ such that each EC is of size at least k ;
2. P can be partitioned into ECs on $S_2 - S_1$ such that each EC is of size at least k ;
3. All tuples in P have the same values on attributes in $S_1 \cap S_2$. ■

According to Definition 2, all tuples in Figure 2(a) form a 2-butterfly. A k -butterfly has several interesting and desirable properties.

Proposition 1 (k -anonymity of butterfly). *Let P be a k -butterfly with respect to QIDs S_1 and S_2 . Then, P is k -anonymous with respect to S_1 and S_2 .*

Proof. The k -anonymity of P with respect to S_1 follows with the first and the third conditions of Definition 2. Similarly, the k -anonymity of P with respect to S_2 follows with the second and the third conditions of Definition 2. □

Proposition 1 indicates that k -butterflies can be used to anonymize a table for two QIDs since the k -anonymity requirement on each QID can be satisfied.

Proposition 2 (EC and butterfly). *In table T , an equivalence class of size k with respect to union QID $S_1 \cup S_2$ is a k -butterfly with respect to S_1 and S_2 , where S_1 and S_2 are two QIDs on T .*

Proof. An EC of size k clearly satisfies the three conditions in Definition 2. □

Proposition 2 indicates that, in anonymization for multiple QIDs, ECs with respect to the union QID is a special case of butterflies. Importantly, a butterfly provides more flexibility that it does not require all values be the same on the union QID. The flexibility brings in the opportunity for reducing information loss in anonymization, as will be explored by our anonymization algorithm in Section 4.2.

Are butterflies sufficient to anonymize a table for multiple QIDs?

Theorem 2 (Butterfly). *A table T is k -anonymous with respect to QIDs S_1 and S_2 if and only if the tuples in T can be partitioned into exclusive subsets P_1, \dots, P_l such that each P_i ($1 \leq i \leq l$) is a k -butterfly with respect to S_1 and S_2 .*

Proof. The sufficiency follows with Proposition 1 immediately. We only need to show the necessity.

Assume that T is k -anonymous with respect to QIDs S_1 and S_2 . If $S_1 \cap S_2 = \emptyset$, then trivially all tuples in T form a k -butterfly. Moreover, if $S_1 \cup S_2 = S_1$ or $S_1 \cup S_2 = S_2$, then T is k -anonymous with respect to $S_1 \cup S_2$. In other words, table T can be divided into ECs on $S_1 \cup S_2$ such that each EC is of size at least k . According to Proposition 1, each EC on $S_1 \cup S_2$ is a k -butterfly.

Now, let us consider the situation where $S_1 \cap S_2 \neq \emptyset$, $S_1 \cup S_2 \neq S_1$, and $S_1 \cup S_2 \neq S_2$. Then, the tuples can be divided into ECs on $S_1 \cap S_2$. Since T is k anonymous with respect to S_1 and $S_1 \supset S_1 \cap S_2$, each EC on $S_1 \cap S_2$ has at least k tuples. We show that each EC on $S_1 \cap S_2$ is a k -butterfly with respect to S_1 and S_2 .

Consider an EC P on $S_1 \cap S_2$, i.e., all tuples in P have the same values on the attributes in $S_1 \cap S_2$. Immediately, the third condition in Definition 2 is satisfied. Since T is k -anonymous with respect to S_1 , the tuples in P can be divided into ECs on S_1 such that each EC has at least k tuples. Each such an EC is also an EC on $S_1 - (S_1 \cap S_2) = S_1 - S_2$. Similarly, all tuples in P can be divided into ECs on S_2 such that each such an EC is also an EC in $S_2 - S_1$ and has at least k tuples. Thus, the first two conditions in Definition 2 are satisfied, too. That is, P is a k -butterfly. \square

According to Theorem 2, the problem of anonymizing a table for QIDs S_1 and S_2 can be reduced to transforming the tuples in T into a set of k -butterflies. We define the k -butterfly anonymization problem as follows.

Given a table T and QIDs S_1 and S_2 , the problem of *k -anonymization using butterflies* is to transform T into table T' consisting of a set of k -butterflies with respect to S_1 and S_2 , and the information loss from T to T' is minimized.

Theorem 3 (Complexity). *The problem of k -anonymization using butterflies is NP-hard.*

Proof. We conduct a reduction from the k -anonymization problem defined in Section 2.1, which has been shown NP-hard [29].

Suppose we want to anonymize table T on QID S . Without loss of generality, let us assume $\|S\| \geq 3$ and let $S = \{A_1, \dots, A_l\}$. We expand table T to table \bar{T} by adding $(l - 1)$ attributes A'_2, \dots, A'_l . On attribute A'_i ($2 \leq i \leq l$), each tuple t in T takes the same value as it has on A_i .

Consider anonymizing \bar{T} using k -butterflies to satisfy k -anonymity with respect to QIDs S and $S' = \{A_1, A'_2, \dots, A'_l\}$. We claim that we can always obtain an anonymization using k -butterflies with the minimum information loss such that for each tuple t in the anonymization, t has the same value on A_i and A'_i ($2 \leq i \leq l$).

Suppose \bar{T}' is an anonymized table with the minimum information loss. We consider C , the information loss of all tuples on attributes A_1, A_2, \dots, A_l (i.e., the information loss in projection $\Pi_{A_1, A_2, \dots, A_l} \bar{T}'$), and C' , the information loss of all tuples on attributes A_1, A'_2, \dots, A'_l (i.e., the information loss in projection $\Pi_{A_1, A'_2, \dots, A'_l} \bar{T}'$).

We show by contradiction $C = C'$. Assume $C < C'$. We change the value of t on A'_i ($2 \leq i \leq l$) to that of t on A_i . The information loss decreases to C which contradicts the assumption that \bar{T}' has the minimum information loss. Similarly, it is impossible that $C > C'$.

Since $C = C'$, we can change the value of t on A'_i ($2 \leq i \leq l$) to that of t on A_i . The information loss does not change.

Now, we prove by contradiction that $\Pi_{A_1, A_2, \dots, A_l} \bar{T}'$ is the k -anonymization with respect to S with the minimum information loss. Assume T' is a k -anonymization of T with respect to S and T' has less information loss than $\Pi_{A_1, A_2, \dots, A_l} \bar{T}'$. Then, we can expand T' to table \bar{T}'' such that attributes A'_2, \dots, A'_l are added, and a tuple t in \bar{T}'' takes the same value on A'_i as it has on A_i ($2 \leq i \leq l$). Table \bar{T}'' is k -anonymous with respect to $S \cup S'$ and has less information loss than \bar{T}' . A contradiction.

The above reduction is of time complexity linear to the number of tuples and the number of attributes in T . \square

4.2 Anonymization Algorithm

Due to Theorem 3, in this section, we develop a heuristic algorithm to anonymize a table using butterflies on two QIDs in order to reduce information loss as much as possible.

4.2.1 General Idea

An EC of at least k tuples on $S_1 \cup S_2$ is a special type of k -butterfly. In the naïve union QID method, we can anonymize table T using ECs of size at least k on $S_1 \cup S_2$.

Alternatively, we can construct large butterflies which are not k -anonymous with respect to $S_1 \cup S_2$. An extreme is that we generalize all tuples to the same on attributes $S_1 \cap S_2$, and then we can conduct k -anonymization on $S_1 - S_2$ and $S_2 - S_1$ independently.

Essentially, there is a tradeoff between using small butterflies and using large butterflies in information loss on different attributes. The advantage of a large butterfly is that it allows less information loss on attributes in $S_1 - S_2$ and $S_2 - S_1$ since tuples do not need to take the same values on those attributes. The disadvantage is that, since all tuples in a butterfly have the same values on attributes in $S_1 \cap S_2$, a large butterfly may lead to heavy information loss on those attributes.

Therefore, to reduce the information loss using butterflies, we need to balance the gain on the attributes in $S_1 \cup S_2 - S_1 \cap S_2$ and the loss on the attributes in $S_1 \cap S_2$.

The general idea of our method to anonymize a table T is in two steps.

First, we anonymize T on the union QID $S_1 \cup S_2$, and form a binary hierarchy (i.e., a binary tree) of ECs. Each internal node in the hierarchy is a set of ECs.

Second, we examine the nodes in the hierarchy of ECs bottom-up to check whether reorganizing the tuples at a node into a butterfly may potentially reduce the information loss. If so, we apply a butterfly construction algorithm on the set of tuples. If the butterfly constructed as such reduces the information loss, it is used to anonymize the tuples.

4.2.2 Step 1: Building a Binary Hierarchy of ECs

First, we build a binary hierarchy of ECs on $S_1 \cup S_2$. This hierarchy is a natural product of some generalization algorithms such as *Mondrian* [24] (reviewed in Section 3), which is employed in our experiments. In case the ECs are computed by other algorithms, the hierarchy can be easily constructed through a “binary clustering” of all the ECs. To illustrate, assume that the generalized form of each EC is a rectangle (c.f. Figure 1) in the space formed by the attributes of $S_1 \cup S_2$. Then, we only need to create a binary R-tree on the ECs, which is already a good hierarchy.

4.2.3 Estimating Reduction of Information Loss

In a binary hierarchy of ECs, *the set of tuples at a node N* , denoted by $T(N)$, are the tuples in the ECs that are descendants of N . Now, our task is to try to organize the tuples in $T(N)$ into butterflies to reduce information loss.

For a node N in the binary hierarchy of ECs, in order to efficiently check whether reorganizing the tuples in $T(N)$ into a butterfly may reduce information loss, we derive a lower bound of the information loss in such a butterfly. The computation of the lower bound does not require to

construct the butterfly. Thus, we can first check whether the lower bound indicates a potential reduction of information loss before we construct the butterfly.

Recall that we adopt the uncertainty penalty (Definition 1) to measure information loss. We use the i NN distance to establish a lower bound.

Definition 3 (i NN distance). *Let E be an equivalence class, $t \in E$ be a tuple in E , and A be a set of attributes. For i ($0 \leq i \leq \|E\|$), the i -th nearest neighbor distance (i NN distance for short) of t on A is $NNDist_A(t, i, E) = dist(t, NN(t, i, E))$, where $NN(t, i, E)$ is the i -th nearest neighbor of t in E , and $dist(t_1, t_2) = \sum_{A \in S} \frac{\|t_1[A] - t_2[A]\|}{\|A\|}$ is the minimum uncertainty penalty needed to generalize t_1 and t_2 into the same EC with respect to QID S . ■*

We have the following lower bound of information loss.

Theorem 4 (Information loss). *Let E_1, \dots, E_m ($m \geq 1$) be ECs on $S_1 \cup S_2$ in a table T , and $G = \cup_{i=1}^m E_i$ be the set of tuples in those ECs. If a k -butterfly with respect to QIDs S_1 and S_2 is constructed using all tuples in G , then the information loss in the k -butterfly is at least*

$$L(G) = \|G\|(\lambda_{S_1-S_2} + \lambda_{S_2-S_1}) + Loss(S_1 \cap S_2)$$

where $Loss(S_1 \cap S_2)$ is the information loss due to the generalization of all tuples in G to the same on $S_1 \cap S_2$, and for $A = S_1 - S_2$ or $S_2 - S_1$,

$$\lambda_A = \max\{\min_{t \in E \subset G} \{NNDist_A(t, \lceil \frac{k}{m} \rceil - 1, E)\}, \min_{t \in E \subset G} \{NNDist_A(t, k - 1, T)\}\}.$$

Proof. Suppose we build a k -butterfly on m ECs E_1, \dots, E_m . In the resulting butterfly, all tuples are generalized to the same on $S_1 \cap S_2$, which leads to information loss $Loss(S_1 \cap S_2)$. Let us consider the information loss on attributes sets $S_1 - S_2$ and $S_2 - S_1$.

On $S_1 - S_2$, the tuples are partitioned into ECs E'_1, \dots, E'_l . Clearly, due to the pigeonhole principle, when $k > m$, for each j ($1 \leq j \leq l$), at least $\lceil \frac{k}{m} \rceil$ tuples in E'_j must be from one of E_1, \dots, E_m . Thus, the information loss for each tuple in each E'_j , denoted by $\lambda_{S_1-S_2}$, satisfies

$$\lambda_{S_1-S_2} \leq \min_{t \in E_i, 1 \leq i \leq m} \{NNDist_{S_1-S_2}(t, \lceil \frac{k}{m} \rceil - 1, E_i)\} \quad (1)$$

Moreover, each EC contains at least k tuples in T . Thus, trivially we have

$$\lambda_{S_1-S_2} \leq \min_{t \in E_i, 1 \leq i \leq m} \{NNDist_{S_1-S_2}(t, k - 1, T)\} \quad (2)$$

Combining Equations 1 and 2, we have

$$\lambda_{S_1-S_2} = \max\{\min_{t \in E \subset G} \{NNDist_{S_1-S_2}(t, \lceil \frac{k}{m} \rceil - 1, E)\}, \min_{t \in E \subset G} \{NNDist_{S_1-S_2}(t, k - 1, T)\}\}.$$

There are $\|\cup_{i=1}^m E_i\|$ tuples in total. Thus, the information loss of the butterfly on $S_1 - S_2$ is at least $\|\cup_{i=1}^m E_i\| \cdot \lambda_{S_1-S_2}$. A similar analysis holds for the information loss on $S_2 - S_1$. Summing up the information loss items on $S_1 - S_2$, $S_1 \cap S_2$, and $S_2 - S_1$, we have $L(\cup_{i=1}^m E_i)$ in the theorem as the lower bound of the information loss. □

Computing the lower bound in Theorem 4 is efficient. By scanning all tuples in G once, we can get the range of the values on each attribute in $S_1 \cap S_2$. Thus, $Loss(S_1 \cap S_2)$ can be calculated immediately as the sum of the width of ranges on all attributes in $S_1 \cap S_2$. We will explain how to compute $\lambda_{S_1-S_2}$ and $\lambda_{S_2-S_1}$ efficiently in Section 4.2.4. Furthermore, we preprocess the data set T to find $NNDist_{S_1-S_2}(t, k-1, T)$ and $NNDist_{S_2-S_1}(t, k-1, T)$ for each tuple t in the whole table T .

4.2.4 Step 2: Applying Butterflies

In the second step, we scan in the bottom-up manner the binary hierarchy of ECs built in the first step. For each node N , the tuples in $T(N)$ are anonymized by the children of N using either ECs or butterflies. Thus, the total information loss can be calculated by summing up the loss of all children of N . We compare this loss with the lower bound of information loss using one butterfly on all tuples in $T(N)$ given by Theorem 4. If the lower bound given by Theorem 4 is less, then we construct a butterfly on N .

As a special case, if all tuples in $T(N)$ are identical on attributes in $S_1 \cap S_2$, then we construct a butterfly on the node. In such a case, the information loss $Loss(S_1 \cap S_2) = 0$ in the butterfly. Moreover, the butterfly allows that the tuples do not need to agree with each other on $S_1 - S_2$ and $S_2 - S_1$, and thus is expected to have lower information loss.

The detailed algorithm is presented in Figure 3. In order to compute the lower bounds in Theorem 4 efficiently, we pre-compute the i NN distance for each tuple in an EC for $i = 1, \dots, k$. In implementation, for each EC, we only need to keep the shortest i NN distance for $i = 1, \dots, k$ among all tuples in the EC. Therefore, the space cost for each EC is $O(k)$. In the binary hierarchy of ECs, there are totally $O(n/k)$ leaf nodes (each corresponding to an EC), where n is the total number of tuples in the table. The hierarchy has $O(n/k)$ internal nodes, each of which requires $O(1)$ space. Hence, the overall space cost of the binary hierarchy is $O(\frac{n}{k} \cdot k) = O(n)$.

Importantly, $Loss(S_1 \cap S_2)$ in Theorem 4 monotonically increases as we traverse the binary hierarchy bottom-up. That is, if N_1 is the parent of N_2 in the hierarchy, then $Loss(S_1 \cap S_2)$ at N_1 is larger than or equal to that at N_2 . Thus, when the information loss on $Loss(S_1 \cap S_2)$ is large, it offsets the gain on $S_1 - S_2$ and $S_2 - S_1$ of using butterflies. This control avoids constructing large butterflies at nodes close to the root of the binary hierarchy. In the algorithm, once $Loss(S_1 \cap S_2)$ at a node N is greater than or equal to the total information loss on the whole data set, we do not need to check any ancestors of N further based on the monotonicity.

Without any index on points in an EC, it takes $O(k^2)$ time to calculate the i NN distances within each EC; thus the total overhead of i NN-distance computation is $(k^2 \cdot \frac{n}{k})$. In butterfly computation, the time cost to check a node is linear to the number of leaves in its subtree. Hence, the total check-time for all nodes at the same level of the hierarchy equals the number $O(n/k)$ of leaves. Considering all levels of the hierarchy, the overall checking time is $O(\frac{n}{k} \log_2 \frac{n}{k})$. Hence, the overall time complexity of the second step is $O(k^2 \cdot \frac{n}{k} + \frac{n}{k} \log_2 \frac{n}{k} \cdot \alpha) = O(k \cdot n + \frac{n}{k} \log_2 \frac{n}{k} \cdot \alpha)$, where α is the cost to build a butterfly.

Input: a binary hierarchy H of ECs, QIDs S_1 and S_2 ;
Output: a binary hierarchy H using ECs and butterflies
to reduce information loss;

Method:

- 1: $l = 0$; // l is the total information loss in the current H
- 2: check all nodes in H in the bottom-up manner,
for each node N do
- 3: if N is a leaf node then
- 4: for each tuple $t \in E$ where E is the EC in N do {
- 5: compute $NNDist_{S_1-S_2}(t, i, E)$ and
 $NNDist_{S_2-S_1}(t, i, E)$ for $i = 1, \dots, k$;
- 6: calculate the information loss of N ;
- 7: $l = l +$ the information loss of N ;
- }
- 8: else { // N is not a leaf node
- 9: $G =$ the set of tuples in the ECs that are
descendants of N ;
- 10: if all tuples in G take the same value on $S_1 \cap S_2$
- 11: then call *butterfly*(N);
- 12: else {
- 13: let $Loss(G)$ be the sum of information loss in
children of N ;
- 14: if $Loss(G) > L(G)$ in Theorem 4
- 15: then call *butterfly*(N);
- 16: if $Loss(S_1 \cap S_2) \geq l$ at N
- 17: then prune all ancestors of N ;
- }
- }
- }

Figure 3: Applying butterflies to reduce information loss.

4.2.5 Building Butterflies

Now, the only remaining issue is how to construct a butterfly on a set of tuples. This is straightforward according to Definition 2.

Let G be the set of tuples on which we want to construct a k -butterfly. We conduct the following two steps.

First, we generalize all tuples in G to the same on attributes in $S_1 \cap S_2$ and calculate the information loss $Loss(S_1 \cap S_2)$. As explained before, this can be done by one scan of all tuples in G .

Second, we can use a k -anonymization algorithm to anonymize tuples in G on QID $S_1 - S_2$. This step constructs the “left wings” of the butterfly. Similarly, we apply the anonymization algorithm to anonymize tuples in G on QID $S_2 - S_1$ for the “right wings” of the butterfly.

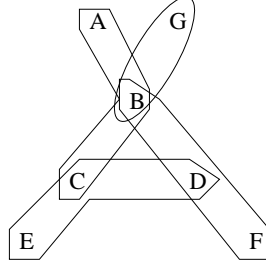


Figure 4: A super butterfly for k -anonymization with respect to 4 QIDs.

Once the butterfly is computed at node N , we can calculate the information loss in the butterfly. If it is smaller than the sum of information loss of the children of N , then the butterfly replaces the children of N , and the information of N is updated. Consequently, we also update the total information loss of the whole data set (i.e., variable l in Figure 3) accordingly.

5 Extensions

In Section 4, we discussed the basic case where there are two QIDs. In this section, we first extend our solution to the general case where there are more than two QIDs. In addition, we discuss how to provide privacy preservation when users may collude.

5.1 Handling More Than Two QIDs

In the case of having two QIDs, we generalize the tuples in a butterfly to be the same on the common attributes between the two QIDs, and thus the tuples can be anonymized independently in the remaining attributes in the two QIDs, respectively. We can generalize the idea to handle more than two QIDs as elaborated in the following example.

Example 3 (Super butterfly). *Consider anonymizing a table $T = (A, B, C, D, E, F, G)$ with respect to 4 QIDs, $S_1 = ABC$, $S_2 = CDE$, $S_3 = BDF$ and $S_4 = BG$. The union QID method can anonymize the table on the union QID $S = \cup_{i=1}^4 S_i = ABCDEFG$.*

Noticing that attributes A, E, F and G appear in only one QID, while attributes B, C and D appear in multiple QIDs, we can construct a super butterfly on a set of tuples as follows. First, all tuples are generalized to the same on attributes BCD . Then, the tuples can be independently generalized into ECs on A, E, F and G , respectively. This super butterfly has wings in 4 “directions”, as visualized in Figure 4. ■

Let us generalize the observation in Example 3.

Definition 4 (Super butterfly). *Given a table T and m QIDs S_1, \dots, S_m in T such that $S = \cup_{i=1}^m S_i$ is a proper superset of every S_i ($1 \leq i \leq m$), i.e., $S_i \subset S$. A set of tuples $P \subseteq T$ in T is a **k -super butterfly** with respect to S_1, \dots, S_m if (1) for each S_i ($1 \leq i \leq m$), P can be partitioned into ECs on $S_i - \cup_{j \neq i} S_j$ such that each EC is of size at least k ; and (2) all tuples in P have the same values on $\cup_{i \neq j} S_i \cap S_j$. ■*

Similar to the discussion in Section 4.1, we have the following properties of super butterflies.

Proposition 3 (*k*-anonymity of super butterfly). *Let P be a k -super butterfly with respect to QIDs S_1, \dots, S_m . Then, P is k -anonymous with respect to S_i ($1 \leq i \leq m$).* ■

Proposition 4 (EC and super butterfly). *In table T , an EC of size k with respect to union QID $\cup_{i=1}^m S_i$ is a k -super butterfly with respect to S_i ($1 \leq i \leq m$), where S_i is a QID on T .* ■

Importantly and interestingly, Theorem 2 can also be generalized for super-butterflies.

Theorem 5 (Super butterfly). *A table T is k -anonymous with respect to QIDs S_1, \dots, S_m if and only if the tuples in T can be partitioned into exclusive subsets P_1, \dots, P_l such that each P_j ($1 \leq j \leq l$) is a k -super butterfly with respect to S_1, \dots, S_m .*

Proof. The sufficiency follows with Proposition 3 immediately. We only need to show the necessity. The proof of the necessity is a straightforward generalization of the proof of Theorem 2. □

Consequently, our anonymization algorithm in Section 4.2 can also be generalized for multiple QIDs. We first anonymize a table into ECs of size at least k on the union QID, and construct a binary hierarchy of those ECs. Then, we scan the nodes in the binary hierarchy in a bottom-up manner and check for each node whether using all tuples at the node to build a super butterfly may lead to less information loss using a lower bound similar to Theorem 4. If so, we generalize all tuples at the node to the same on $\cup_{i \neq j} S_i \cap S_j$, the “body” of the super butterfly, and anonymize the tuples into ECs of size at least k on each $S_i - \cup_{i \neq j} S_j$ independently and compute the reduction of information loss. If the super butterfly leads to less information loss, it is taken. Limited by space, we omit the details here.

5.2 Handling Collusion

In the discussion so far, we assume that different users would not collude, that is, they would not share or exchange their background knowledge. This assumption is often reliable in practice when the data receivers are reputable organizations. In the example in Section 1, the anonymized data is sent to an auto-insurance company and a government department, respectively. The likelihood that these two organizations collude is minor, because both of them are tightly watched by legislative agencies and the penalty of collusion is severe.

In practice, to be conservative, we may still need some mechanism to protect privacy even when users may collude. We consider two types of collusion.

Definition 5 (Collusion types). **Data sharing collusion** is that two or multiple users share the k -anonymous tables respectively published to them. **Background knowledge sharing collusion** is that two or multiple users share their background knowledge which can be used to issue re-identification attacks. ■

As illustrated in Example 1, the multiple release method cannot protect data sharing collusion. In the butterfly method and the union QID method, all user receive the same k -anonymous table, and thus data sharing collusion cannot help re-identification attacks at all.

Now, let us focus on background knowledge sharing collusion. When two users carrying the capability of re-identification using QIDs S_1 and S_2 collude, they can re-identify individuals using the joint QID $S_1 \cup S_2$. In the worst case, if all users U_1, \dots, U_m collude, the re-identification attack may use the union QID $\cup_{i=1}^m S_i$.

Obviously, we should guarantee k -anonymity with respect to each user’s QID. Do we, however, need as strong protection against their collusion? Since collusion is unlikely to happen, why not lower the protection level to achieve less information loss in anonymization and retain better quality in data utilization? Here, we advocate a framework where the privacy preserving requirement on re-identification attacks using joint QIDs should be lower than the requirement for individual users. We propose a two level anonymization framework.

- To release the micro data in a table T for multiple users U_1, \dots, U_m , the table should be anonymized to T' so that T' is k -anonymous with respect to S_1, \dots, S_m where S_i ($1 \leq i \leq m$) is the QID modeling the background knowledge of user U_i .
- To protect privacy even when collusion happens, table T' should also be k' -anonymous with respect to $\cup_{i=1}^m S_i$, where $k' < k$ reflects a weaker requirement on privacy preservation due to the low chance of collusion.

Our butterfly solution can be easily adapted to handle collusion based on the above two level framework. The adapted solution consists of the following steps.

First, to anonymize a table T with respect to QIDs S_1, \dots, S_m , we can first conduct k' -anonymization on the union QID $\cup_{i=1}^m S_i$ to obtain table T_1 . After this step, the requirement of k' -anonymity with respect to the union QID is satisfied.

Second, we obtain T'_1 by removing duplicate tuples from T_1 . That is, each EC in T_1 has only one representative in T'_1 . Then, we conduct $\lceil \frac{k}{k'} \rceil$ -anonymization with respect to S_1, \dots, S_m using butterflies on T'_1 to obtain table T_2 .

Last, we make up table T' as follows. For each tuple t in T'_1 , which represents an EC E in T_1 , if t is anonymized to t' in T'_1 , we duplicate t' $\|E\|$ times in T' . Since $\|E\| \geq k'$ and $\lceil \frac{k}{k'} \rceil k' \geq k$, T' is at least k -anonymous with respect to S_1, \dots, S_m , respectively.

6 Empirical Study

In this section, we report a systematic empirical study using the Adults census data set from the UC Irvine machine learning repository. This data set has become a de facto benchmark in evaluations of k -anonymization methods. In our experiments, we use a randomly selected subset of 300 thousand tuples. Following the similar way to use the data set in the previous studies, we use 9 attributes as shown in Table 5. The domain of each attribute is integers in the range shown in Table 5.

All our experiments are conducted on a PC computer with a Pentium 3.0 GHz CPU and 1 GB main memory, running the Microsoft XP operating system. All our programs are written in C++.

We use the uncertainty penalty (Definition 1) as the information loss measure. We compare two methods: the union QID method (defined in Section 2.2) and the butterfly method in Section 4. Limited by space, we only report the results on k -anonymization with respect to 2 QIDs here.

Table 5: The attributes and their domains in our experiments.

Attribute	Domain
Age	[16, 93]
Occupation	[1, 25]
Birthplace	[1, 83]
Marital	[1, 6]
Gender	[1, 2]
Salary	[0, 50]
Work-class	[2, 10]
Education	[1, 17]
Race	[1, 9]

Except for Section 6.7, to enforce the two level anonymization framework discussed in Section 5.2, we always set $k' = 2$ and vary k . That is, we first conduct a 2-anonymization on the union QID and then apply the butterfly method to achieve k -anonymity with respect to individual QIDs. The reason that we set k' to a small number in the experiments is to leave more space to verify the effectiveness of butterflies. We will examine the effectiveness of k' on two level anonymization in Section 6.7.

6.1 Effectiveness of Butterflies

Figure 5 shows the percentage of tuples in the whole data set anonymized by non-trivial butterflies (i.e., not anonymized into an EC with respect to the union QID). Two cases are tested. In the case $d_{com} = 2$, there are two common attributes, **Education** and **Marital**, between the two QIDs, while in the case $d_{com} = 3$, there are three common attributes, **Education**, **Marital** and **Gender**. In addition to the common attributes, in both cases, S_1 also contains attributes **Age**, **Occupation**, and **Birthplace**, and S_2 contains attributes **Salary**, **Work-class**, and **Race**.

The results clearly show that most of the tuples can be anonymized by butterflies to achieve information loss less than that in the union QID method. This strongly verifies the effectiveness of the butterflies.

As explained in Sections 4.2.4 and 4.2.5, the butterfly method uses Theorem 4 to check whether constructing butterflies may have a chance to reduce information loss in anonymization. To test the effectiveness of Theorem 4, Figure 6 shows the success rate of butterfly construction with respect to different values of k . The success rate is defined as $\frac{b}{a}$ where a is the number of internal nodes in the hierarchy of ECs which are indicated by Theorem 4 that constructing butterflies may reduce information loss, and b is the number of nodes where constructing butterflies actually reduces information loss.

The success rate in both cases increases monotonically as k increases. With a larger k , more tuples need to be grouped together in the union method and thus lead to heavier information loss. The butterflies can have more opportunities to reduce information loss when k is larger, since butterflies can identify the local areas where the tuples can be re-grouped to reduce information loss.

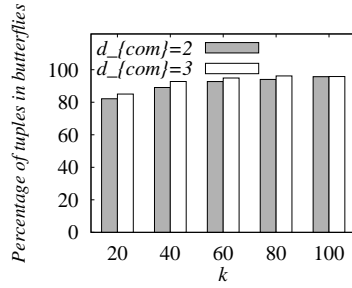


Figure 5: The percentage of tuples anonymized by non-trivial butterflies.

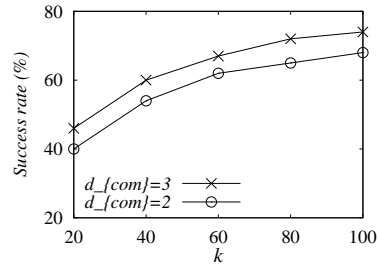


Figure 6: The success rate of butterfly construction.

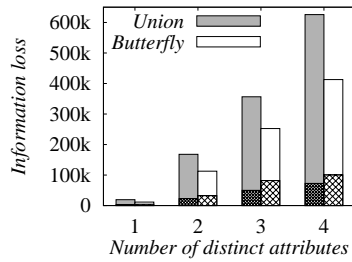


Figure 7: The information loss w.r.t. diversity of QIDs.

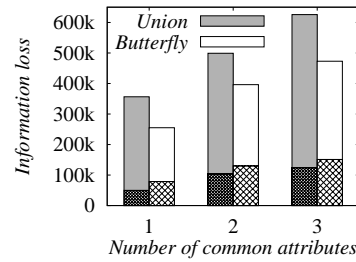


Figure 8: The information loss w.r.t. complexity of common attributes.

6.2 Diversity of QIDs

Intuitively, the more diverse the QIDs (i.e., the more attributes that are not shared between the two QIDs in question), the better chance butterflies have to reduce information loss. To verify the effectiveness, we set `Education` as the common attribute between QIDs S_1 and S_2 , and vary the number of attributes in S_1 and S_2 from 2 to 5. In addition to the common attribute `Education`, the attributes added into S_1 are `Age`, `Occupation`, `Birthplace`, and `Marital` in the order, and the ones added into S_2 are `Salary`, `Work-class`, `Race`, and `Gender` in the order. For example, when the number of distinct attributes is 1, $S_1 = \{\text{Education, Age}\}$ and $S_2 = \{\text{Education, Salary}\}$. When the number of distinct attributes is 2, $S_1 = \{\text{Education, Age, Occupation}\}$ and $S_2 = \{\text{Education, Salary, Work-class}\}$.

Figure 7 shows the result. The information loss in both the union method and the butterfly method is divided into two parts: the loss on the common attributes (the lower parts in the bars) and the loss on the distinct attributes (the upper parts in the bars).

When the number of distinct attributes in each QID increases, the union QID has to include more attributes and thus the information loss increases dramatically. However, the butterfly method only needs to conduct k -anonymization with respect to individual QIDs, which have much fewer attributes. As shown in the figure, the difference in information loss between the union QID method and the butterfly method increases dramatically as the number of distinct attributes in each QID increases.

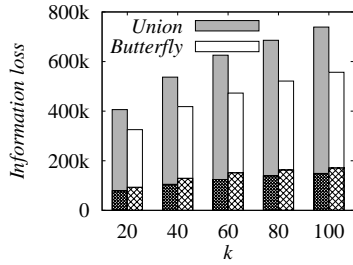


Figure 9: The information loss w.r.t. parameter k .

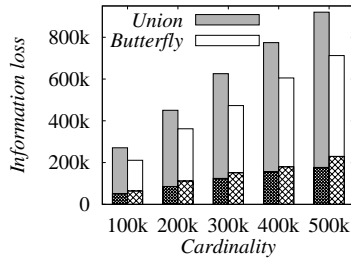


Figure 10: The information loss w.r.t. number of tuples.

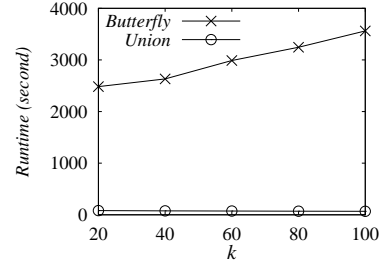


Figure 11: The runtime versus k .

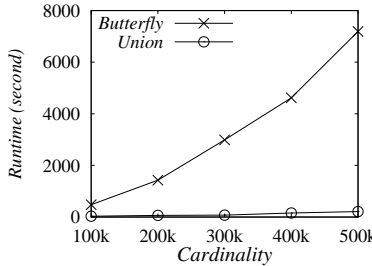


Figure 12: The scalability.

Importantly, the results clearly show that the information loss on the common attributes in the butterfly method and that in the union method are highly comparable. The extra lost in the butterfly method is very small comparing to the overall reduction in information loss. In other words, butterflies pay only very minor extra information loss on the common attributes and save substantial information loss on the distinct attributes.

6.3 Number of the Common Attributes

As explained in Section 4.2.1, for QIDs S_1 and S_2 , the effectiveness of the butterfly method depends on the tradeoff between the gain on anonymizing $S_1 - S_2$ and $S_2 - S_1$ independently versus the loss on anonymizing all tuples in a butterfly to be the same on $S_1 \cap S_2$. If the common attributes between S_1 and S_2 has small cardinality (i.e., many tuples share the same values on them), building butterflies can gain more.

In Figure 8, we test the effectiveness of the butterfly method with respect to the number of common attributes between the two QIDs. We add attributes **Education**, **Marital**, and **Gender** in the order into both S_1 and S_2 , the two QIDs. In addition, S_1 contains attributes **Age**, **Occupation**, and **Birthplace**, and S_2 contains **Salary**, **Work-class**, and **Race**. This experiment also tests the effect of the cardinality of the common attributes between the two QIDs, since we can treat multiple common attributes as one composite attribute whose cardinality is the product of the cardinalities of the common attributes. Again, we show the breakdown of the information loss including the loss on the common attributes and that on the distinct attributes.

The difference between the two methods in information loss remains relatively stable as the number of common attributes increases. On the one hand, with more common attributes, the union QID becomes larger and thus incurs more information loss. On the other hand, the butterfly method has to pay more information loss in anonymizing tuples in a butterfly to the same on the common attributes. Interestingly and importantly, comparing to the union method, the butterfly method only incurs very small extra information loss on the common attributes. The butterflies can balance the tradeoff between the loss on common attributes and that on distinct attributes well.

6.4 Effect of k

To test the effect of parameter k on anonymization quality, in Figure 9, we vary k from 20 to 100, and plot the information loss. We set $S_1 = \{\text{Age, Occupation, Birthplace, Gender, Marital, Education}\}$ and $S_2 = \{\text{Gender, Marital, Education, Salary, Work-class, Race}\}$. There are three common attributes, **Education**, **Marital** and **Gender**, between S_1 and S_2 .

With a larger value of k , more tuples need to be anonymized to an EC in order to satisfy the k -anonymity requirement, and thus more information loss happens. The advantage of the butterfly method against the union QID method is insensitive to the value of k . In other words, the butterfly method can improve the anonymization quality remarkably no matter to what value k is set.

6.5 Effect of Data Set Size

We test the anonymization quality of the two methods on data sets of various number of tuples. The QIDs are the same as those in Section 6.4. The cardinality of the data sets (i.e., the number of tuples) varies from 100,000 to 500,000. The results are shown in Figure 10.

The information loss in both methods increases as the number of tuples increases, since more tuples are involved, more anonymization is conducted. However, the average information loss per tuple in both methods decreases in both methods as more tuples are added into the data set. Limited by space, we omit the figure here.

In anonymization, tuples are organized into local groups as ECs. Both the union QID method and the butterfly method can identify tuples that are similar to each other to form ECs. When there are more tuples, the data set becomes denser, and thus the local groups become more compact. Generalizing such a more compact group leads to less average information loss per tuple.

The increase of information loss in the butterfly method is slower than that in the union QID method. This is consistent with the results from the other experiments showing that the butterfly method can exploit more opportunities to preserve information in anonymization.

6.6 Anonymization Efficiency

Since the butterfly method uses the union QID method as the first step, the runtime of the butterfly method is expected longer than that of the union QID method. Figures 11 and 12 show the runtime of the two methods with respect to parameter k and the database size, respectively. The settings are the same as the experiments in Figures 9 and 10, respectively.

With a larger value of k , there are less ECs in the union QID method and thus the runtime of the union QID method decreases mildly. However, the runtime of the butterfly method increases mildly

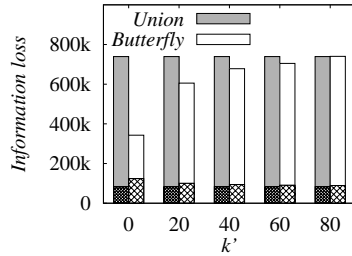


Figure 13: The information loss of two level anonymization.

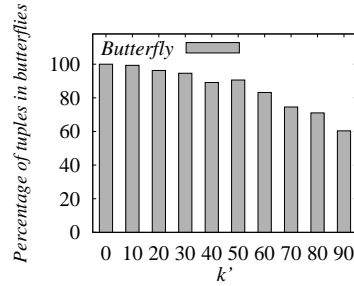


Figure 14: The percentage of tuples anonymized by non-trivial butterflies in two level anonymization.

since the corresponding butterflies are larger and thus the number of possible ways of re-organizing tuples also increases. Both methods are scalable with respect to the database size.

As anonymization is off-line, background processing, the butterfly method, which provides remarkable information loss reduction, is highly preferable to the union QID method.

6.7 Effectiveness of Two Level Anonymization

To test the effectiveness of two level anonymization, we set $S_1 = \{\text{Age, Occupation, Birthplace, Marital, Education}\}$ and $S_2 = \{\text{Education, Gender, Salary, Work-class, Race}\}$. Education is the common attribute between S_1 and S_2 . We set $k = 100$ and vary k' from 0 to 90.

Figure 13 shows the information loss of the two methods with respect to k' . The union method cannot take the advantage of the two level anonymization and thus the information loss is constant. The information loss in the butterfly method increases as k' increases, since tuples have to be grouped into ECs of size k' before butterfly construction can be applied. Interestingly, the gain of the butterfly method is notable even when k' is 60% of k , which indicates that the two level anonymization mechanism in the butterfly method can reduce information loss effectively even when k' is close to k . Moreover, Figure 14 shows the percentage of tuples anonymized by non-trivial butterflies in two level anonymization. Even when $k' = 0.9k$, more than 60% of tuples are in non-trivial butterflies. The results strongly suggest that butterflies can be used to reduce information loss extensively.

7 Conclusions

In this paper, we tackled a novel problem of privacy preserving publishing on multiple QIDs. The problem is challenging and the straightforward union QID method is ineffective. We developed an elegant solution – the butterfly approach. The critical idea is that we can anonymize data with respect to individual QIDs as long as we can keep the anonymization consistent on the attributes common to the QIDs. We proved that finding the optimal butterflies is NP-hard. A fast and practical algorithm was proposed. Our solution can handle multiple QIDs and provide proper protection against possible collusion. We use a de facto benchmark data set to test the effectiveness

of our solution.

Privacy preserving publishing for multiple users can find many applications. Our approach takes only the initial step. A few very interesting problems remain open. For example, in some applications, l -diversity or other more specific privacy preservation requirements are needed. Then, how can we anonymize data properly for multiple users and reduce information loss as much as possible? Those stimulating questions need to be explored systematically by future work.

References

- [1] [Http://privacy.cs.cmu.edu/datafly/index.html](http://privacy.cs.cmu.edu/datafly/index.html).
- [2] Charu C. Aggarwal, Jian Pei, and Bo Zhang. On privacy preservation against adversarial data mining. In *Proc. of ACM Knowledge Discovery and Data Mining (SIGKDD)*, pages 510–516, 2006.
- [3] Charu C. Aggarwal and Philip S. Yu. A condensation approach to privacy preserving data mining. In *Proc. of Extending Database Technology (EDBT)*, pages 183–199, 2004.
- [4] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Anonymizing tables. In *Proc. of International Conference on Database Theory (ICDT)*, pages 246–258, 2005.
- [5] Gagan Aggarwal, Tomas Feder, Krishnaram Kenthapadi, Samir Khuller, Rina Panigrahy, Dilys Thomas, and An Zhu. Achieving anonymity via clustering. In *Proc. of ACM Symposium on Principles of Database Systems (PODS)*, pages 153–162, 2006.
- [6] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu. Hippocratic databases. In *Proc. of Very Large Data Bases (VLDB)*, pages 143–154, 2002.
- [7] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. In *Proc. of ACM Management of Data (SIGMOD)*, pages 439–450, 2000.
- [8] R.J. Bayardo and R. Agrawal. Data privacy through optimal k -anonymization. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 217–228, 2005.
- [9] Elisa Bertino, Claudio Bettini, Elena Ferrari, and Pierangela Samarati. An access control model supporting periodicity constraints and temporal reasoning. *ACM Transactions on Database Systems (TODS)*, 23(3):231–285, 1998.
- [10] Elisa Bertino and Elena Ferrari. Secure and selective dissemination of xml documents. *ACM Trans. Inf. Syst. Secur.*, 5(3):290–331, 2002.
- [11] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the sulq framework. In *Proc. of ACM Symposium on Principles of Database Systems (PODS)*, pages 128–138, 2005.

- [12] Bee-Chung Chen, Raghu Ramakrishnan, and Kristen LeFevre. Privacy skyline: Privacy with multidimensional adversarial knowledge. In *Proc. of Very Large Data Bases (VLDB)*, pages 770–781, 2007.
- [13] Chris Clifton. Using sample size to limit exposure to data mining. *Journal of Computer Security*, 8(4), 2000.
- [14] Yang Du, Tian Xia, Yufei Tao, Donghui Zhang, and Feng Zhu. On multidimensional k -anonymity with local recoding generalization. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 1422–1424, 2007.
- [15] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference (TCC)*, pages 265–284, 2006.
- [16] Alexandre Evfimievski, Johannes Gehrke, and Ramakrishnan Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proc. of ACM Symposium on Principles of Database Systems (PODS)*, pages 211–222, 2003.
- [17] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 205–216, 2005.
- [18] Gabriel Ghinita, Panagiotis Karras, Panos Kalnis, and Nikos Mamoulis. Fast data anonymization with low information loss. In *Proc. of Very Large Data Bases (VLDB)*, pages 758–769, 2007.
- [19] Tochukwu Iwuchukwu and Jeffrey F. Naughton. K -anonymization as spatial indexing: Toward scalable and incremental anonymization. In *Proc. of Very Large Data Bases (VLDB)*, pages 746–757, 2007.
- [20] V. Iyengar. Transforming data to satisfy privacy constraints. In *Proc. of ACM Knowledge Discovery and Data Mining (SIGKDD)*, pages 279–288, 2002.
- [21] Wei Jiang and Chris Clifton. A secure distributed framework for achieving k -anonymity. *The VLDB Journal*, 15(4):316–333, 2006.
- [22] Daniel Kifer and Johannes Gehrke. Injecting utility into anonymized datasets. In *Proc. of ACM Management of Data (SIGMOD)*, pages 217–228, 2006.
- [23] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k -anonymity. In *Proc. of ACM Management of Data (SIGMOD)*, pages 49–60, 2005.
- [24] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k -anonymity. In *Proc. of International Conference on Data Engineering (ICDE)*, 2006.
- [25] Kristen LeFevre, David DeWitt, and Raghu Ramakrishnan. Workload-aware anonymization. In *Proc. of ACM Knowledge Discovery and Data Mining (SIGKDD)*, 2006.

- [26] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t -closeness: Privacy beyond k -anonymity and l -diversity. In *Proc. of International Conference on Data Engineering (ICDE)*, 2007.
- [27] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramanian. l -diversity: Privacy beyond k -anonymity. In *Proc. of International Conference on Data Engineering (ICDE)*, 2006.
- [28] David J. Martin, Daniel Kifer, Ashwin Machanavajjhala, Johannes Gehrke, and Joseph Y. Halpern. Worst-case background knowledge for privacy-preserving data publishing. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 126–135, 2007.
- [29] Adam Meyerson and Ryan Williams. On the complexity of optimal k -anonymity. In *Proc. of ACM Symposium on Principles of Database Systems (PODS)*, pages 223–228, 2004.
- [30] Shubha U. Nabar, Bhaskara Marthi, Krishnaram Kenthapadi, Nina Mishra, and Rajeev Motwani. Towards robustness in query auditing. In *Proc. of Very Large Data Bases (VLDB)*, pages 151–162, 2006.
- [31] Mehmet Ercan Nergiz, Maurizio Atzori, and Chris Clifton. Hiding the presence of individuals from shared databases. In *Proc. of ACM Management of Data (SIGMOD)*, pages 665–676, 2007.
- [32] Hyoungmin Park and Kyuseok Shim. Approximate algorithms for k -anonymity. In *Proc. of ACM Management of Data (SIGMOD)*, pages 67–78, 2007.
- [33] Vibhor Rastogi, Sungho Hong, and Dan Suciu. The boundary between privacy and utility in data publishing. In *Proc. of Very Large Data Bases (VLDB)*, pages 531–542, 2007.
- [34] Jorg Rothe. Some facets of complexity theory and cryptography: A five-lecture tutorial. *ACM Computing Surveys*, 34(4):504–549, 2002.
- [35] P. Samarati. Protecting respondents’ identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 13(6):1010–1027, 2001.
- [36] Timos K. Sellis, Nick Roussopoulos, and Christos Faloutsos. The $r+$ -tree: A dynamic index for multi-dimensional objects. In *Proc. of Very Large Data Bases (VLDB)*, pages 507–518, 1987.
- [37] L. Sweeney. Achieving k -anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):571–588, 2002.
- [38] Latanya Sweeney. k -anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness, and Knowledge-Based Systems*, 10(5):557–570, 2002.
- [39] Jaideep Vaidya and Chris Clifton. Privacy-preserving k -means clustering over vertically partitioned data. In *Proc. of ACM Knowledge Discovery and Data Mining (SIGKDD)*, pages 206–215, 2003.

- [40] Vassilios S. Verykios, Ahmed K. Elmagarmid, Elisa Bertino, Yucel Saygin, and Elena Dasseni. Association rule hiding. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(4):434–447, 2004.
- [41] Ke Wang and Benjamin C. M. Fung. Anonymizing sequential releases. In *Proc. of ACM Knowledge Discovery and Data Mining (SIGKDD)*, pages 414–423, 2006.
- [42] Ke Wang, Benjamin C. M. Fung, and Philip S. Yu. Template-based privacy preservation in classification problems. In *Proc. of International Conference on Management of Data (ICDM)*, pages 466–473, 2005.
- [43] Raymond Chi-Wing Wong, Ada Wai-Chee Fu, Ke Wang, and Jian Pei. Minimality attack in privacy preserving data publishing. In *Proc. of Very Large Data Bases (VLDB)*, pages 543–554, 2007.
- [44] Raymond Chi-Wing Wong, Jiuyong Li, Ada Wai-Chee Fu, and Ke Wang. (α , k)-anonymity: an enhanced k -anonymity model for privacy preserving data publishing. In *Proc. of ACM Knowledge Discovery and Data Mining (SIGKDD)*, pages 754–759, 2006.
- [45] Xiaokui Xiao and Yufei Tao. Anatomy: Simple and effective privacy preservation. In *Proc. of Very Large Data Bases (VLDB)*, pages 139–150, 2006.
- [46] Xiaokui Xiao and Yufei Tao. Personalized privacy preservation. In *Proc. of ACM Management of Data (SIGMOD)*, pages 229 – 240, 2006.
- [47] Xiaokui Xiao and Yufei Tao. M -invariance: towards privacy preserving re-publication of dynamic datasets. In *SIGMOD Conference*, pages 689–700, 2007.
- [48] Jian Xu, Wei Wang, Jian Pei, Xiaoyuan Wang, Baile Shi, and Ada Wai-Chee Fu. Utility-based anonymization using local recoding. In *Proc. of ACM Knowledge Discovery and Data Mining (SIGKDD)*, pages 785–790, 2006.
- [49] Qing Zhang, Nick Koudas, Divesh Srivastava, and Ting Yu. Aggregate query answering on anonymized tables. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 116–125, 2007.