m-Invariance: Towards Privacy Preserving Re-publication of Dynamic Datasets

Xiaokui Xiao Yufei Tao
Department of Computer Science and Engineering
Chinese University of Hong Kong
Sha Tin, New Territories, Hong Kong
{xkxiao, taoyf}@cse.cuhk.edu.hk

ABSTRACT

The previous literature of privacy preserving data publication has focused on performing "one-time" releases. Specifically, none of the existing solutions supports *re-publication* of the microdata, after it has been updated with insertions and deletions. This is a serious drawback, because currently a publisher cannot provide researchers with the most recent dataset continuously.

This paper remedies the drawback. First, we reveal the characteristics of the re-publication problem that invalidate the conventional approaches leveraging k-anonymity and l-diversity. Based on rigorous theoretical analysis, we develop a new generalization principle m-*invariance* that effectively limits the risk of privacy disclosure in re-publication. We accompany the principle with an algorithm, which computes privacy-guarded relations that permit retrieval of accurate aggregate information about the original microdata. Our theoretical results are confirmed by extensive experiments with real data.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Retrieval Models.
General Terms: Algorithms, Theory.
Keywords: Privacy, Generalization, m-invariance.

1. INTRODUCTION

Privacy preservation has received considerable attention from the database community in the past few years. Depending on the roles of the underlying server, the previous research can be classified into two categories: *centralized publication* and *distributed collection*. The first category assumes that the dataset, called *microdata*, is stored at a *trustable* server. The server releases the data in a manner that protects personal privacy, and permits effective mining on the microdata. The second category addresses a different scenario, where an *un-trustable* server independently contacts a set of individuals, and solicits a tuple from each person. The objective is to devise an approach that allows each person to randomize her/his tuple, such that the server can use the collected dataset for research, yet cannot accurately infer the original form of any tuple.

This paper concerns centralized publication. Consider that a hospital releases the diagnosis records in Table 1a to medical researchers, after discarding the attribute *Name* (we include this column for row referencing). Column *Disease* is *sensitive*, as it contains patients' private data to be protected. Assume that an adversary knows Bob's age 21, Zipcode 12000, and the fact that Bob has been hospitalized before (and thus has a tuple in the microdata). S/he can find out that the first tuple is associated with Bob, namely, Bob must have contracted *dyspepsia*. Here, columns *Age* and *Zipcode* are *quasi-identifier* (QI) attributes, since they can be combined to pinpoint an individual.

Generalization [19] is a popular methodology of privacy preservation. Its rationale is to divide the tuples into several *QI groups*, and then generalize the QI values in each group to a uniform format. A generalized version of Table 1a is presented in Table 1b. The transformation is based on five QI groups, each of which is assigned a group ID as indicated in the first column of Table 1b. Imagine that the hospital publishes Table 1b instead. The previous adversary can no longer uniquely decide Bob's disease, since either of the first two tuples can belong to Bob, i.e., his disease may be *dyspepsia* or *bronchitis*.

A generalized table is considered privacy preserving, if it satisfies a *generalization principle*. The earliest principle, k-*anonymity* [18, 20], requires each QI group to include at least k tuples (e.g., Table 1b is 2-anonymous). Machanavajjhala et al. [15] propose an improved principle, l-*diversity*, which demands every QI group to contain at least l "well-represented" sensitive values. The notion of well-represented can be interpreted in several ways [15]. A popular interpretation is that, in each QI group, at most $1/l$ of the tuples should possess the same sensitive value. By this definition, Table 1b is 2-diverse. In general, stronger privacy protection is ensured, when a larger k or l is deployed.

1.1 Motivation

With a single exception [6], all the existing methods for centralized publication focus on static microdata. Specifically, they are restricted to only one-time publication, and do not support re-publication, after new (existing) tuples are inserted in (deleted from) the microdata. The seminal work by Byun et al. [6] is the first to identify possible privacy attacks due to re-publication, and develops a solution to effectively prevent those attacks. However, that solution supports only insertions, and is inapplicable in the presence of deletions. Privacy preserving re-publication of a fully-dynamic dataset remains an open problem.

To explain the difficulty of the problem, consider that a hospital releases patients' records quarterly, but each publication includes only the results of diagnoses in the 6 months preceding the publication time. Table 1a shows the microdata for the first release, at which time the hospital publishes the generalized relation in Table 1b. The microdata at the second release is presented in Table 2a. The tuples of Alice, Andy, Helen, Ken, and Paul have been deleted

Name	Age	Zip.	Disease
Bob	21	12000	dyspepsia
Alice	22	14000	bronchitis
Andy	24	18000	flu
David	23	25000	gastritis
Gary	41	20000	flu
Helen	36	27000	gastritis
Jane	37	33000	dyspepsia
Ken	40	35000	flu
Linda	43	26000	gastritis
Paul	52	33000	dyspepsia
Steve	56	34000	gastritis

(a) Microdata $T(1)$

G. ID	Age	Zip.	Disease
1	[21, 22]	[12k, 14k]	dyspepsia
1	[21, 22]	[12k, 14k]	bronchitis
2	[23, 24]	[18k, 25k]	flu
2	[23, 24]	[18k, 25k]	gastritis
3	[36, 41]	[20k, 27k]	flu
3	[36, 41]	[20k, 27k]	gastritis
4	[37, 43]	[26k, 35k]	dyspepsia
4	[37, 43]	[26k, 35k]	flu
4	[37, 43]	[26k, 35k]	gastritis
5	[52, 56]	[33k, 34k]	dyspepsia
5	[52, 56]	[33k, 34k]	gastritis

(b) Generalization $T^*(1)$

Table 1: Microdata and its generalization at the 1st release

Name	Age	Zip.	Disease
Bob	21	12000	dyspepsia
David	23	25000	gastritis
Emily	25	21000	flu
Jane	37	33000	dyspepsia
Linda	43	26000	gastritis
Gary	41	20000	flu
Mary	46	30000	gastritis
Ray	54	31000	dyspepsia
Steve	56	34000	gastritis
Tom	60	44000	gastritis
Vince	65	36000	flu

(a) Microdata $T(2)$

G. ID	Age	Zip.	Disease
1	[21, 23]	[12k, 25k]	dyspepsia
1	[21, 23]	[12k, 25k]	gastritis
2	[25, 43]	[21k, 33k]	flu
2	[25, 43]	[21k, 33k]	dyspepsia
2	[25, 43]	[21k, 33k]	gastritis
3	[41, 46]	[20k, 30k]	flu
3	[41, 46]	[20k, 30k]	gastritis
4	[54, 56]	[31k, 34k]	dyspepsia
4	[54, 56]	[31k, 34k]	gastritis
5	[60, 65]	[36k, 44k]	gastritis
5	[60, 65]	[36k, 44k]	flu

(b) Generalization $T^*(2)$

Table 2: Microdata and its generalization at the 2nd release

(as they describe diagnoses over six months ago), while 5 new tuples (with names italicized) have been inserted. Accordingly, the hospital publishes the generalized relation in Table 2b.

Even though both published relations (Tables 1b and 2b) are 2-anonymous and 2-diverse, an adversary can still precisely determine the disease of a patient, by exploiting the correlation between the two "snapshots". To illustrate this, assume, again, an adversary who has Bob's age and Zipcode, and knows that Bob has a record in both Tables 1b and 2b (i.e., Bob was admitted for treatment, within 6 months before both publication times). Based on Table 1b, the adversary is certain that Bob must have contracted either *dyspepsia* or *bronchitis*. From Table 2b, s/he finds out that Bob's disease must be either *dyspepsia* or *gastritis*. By combining the above knowledge, the adversary correctly captures Bob's real disease *dyspepsia*.

The situation in practice, unfortunately, is much worse. Since each tuple in the microdata may be involved in any number of subsequent publications (until the tuple is deleted), there are simply too many potential correlations among various snapshots that may be utilized to derive sensitive values. Therefore, we need a new generalization principle, which guards privacy against inferences leveraging any possible correlations. The dilemma, however, is that such a principle may not exist at all, due to a phenomenon we refer to as *critical absence*.

Let us explain the phenomenon by re-examining the first release from the hospital. Given Table 1b, an adversary (having Bob's QI particulars) is sure that Bob contracted *dyspepsia* or *bronchitis*. The value *bronchitis*, unfortunately, is absent in the microdata (Table 2a) at the second release. As a result, *no matter how Table 2a is generalized, publishing the generalized version always enables the adversary to eliminate the possibility that Bob contracted bronchitis*. Therefore, Bob's privacy will necessarily be breached after the second release.

Note that critical absence never occurs, if only insertions are allowed in the microdata. This is why re-publication is more challenging, when deletions must also be supported. One straightforward attempt to tackle deletions is to simply ignore them. Namely, deleted tuples are allowed to remain in the microdata, and published in future releases together with the authentic tuples that have not been deleted. Although this approach allows the application of the insertion-only solution in [6], it has two obvious defects. First, the amount of data published at each release grows monotonically with time, due to the increasingly-large number of garbage tuples that have been removed. Second, it becomes questionable whether the privacy guarantees in [6] are still valid, when an adversary is aware of tuples' deletion timestamps.

Name	G. ID	Age	Zip.	Disease
Bob	1	[21, 22]	[12k, 14k]	dyspepsia
c_1	1	[21, 22]	[12k, 14k]	bronchitis
David	2	[23, 25]	[21k, 25k]	gastritis
Emily	2	[23, 25]	[21k, 25k]	flu
Jane	3	[37, 43]	[26k, 33k]	dyspepsia
c_2	3	[37, 43]	[26k, 33k]	flu
Linda	3	[37, 43]	[26k, 33k]	gastritis
Gary	4	[41, 46]	[20k, 30k]	flu
Mary	4	[41, 46]	[20k, 30k]	gastritis
Ray	5	[54, 56]	[31k, 34k]	dyspepsia
Steve	5	[54, 56]	[31k, 34k]	gastritis
Tom	6	[60, 65]	[36k, 44k]	gastritis
Vince	6	[60, 65]	[36k, 44k]	flu

(a) $T^*(2)$ with counterfeits

Group-ID	Count
1	1
3	1

(b) Published counterfeit statistics

Table 3: Remedying critical absence with counterfeits

1.2 Contributions

This paper presents the first study on privacy preserving publication of fully-dynamic datasets, which can be modified by any sequence of insertions and deletions. The core of our solution is the integration of two novel concepts: *m-invariance* and *counterfeited generalization*. The former is a new generalization principle, whose satisfaction ensures strong protection of sensitive information in re-publication. The latter is a technique that facilitates the enforcement of m-invariance, in the presence of critical absence.

To illustrate the idea, let us revisit the scenario, where the hospital has published Table 1b (with respect to the microdata Table 1a), and tries to release an anonymized version of Table 2a. Our method leads to publication of Table 3a, and an auxiliary Table 3b. Specifically, Table 3a involves a generalized tuple for every row in Table 2a, together with two *counterfeit tuples* c_1 and c_2 (names are not published; they are included for row referencing). The 13 tuples are partitioned into six QI groups. Table 3b indicates that a counterfeit is placed in QI groups 1 and 3, respectively (the purpose of releasing such statistics is to enhance the effectiveness of data analysis).

From an adversary's perspective, a counterfeit tuple is indistinguishable from the other rows in the QI group (that contains the counterfeit). Let us consider once more the adversary who has the precise QI details of Bob, and attempts to infer the disease of Bob from Tables 1b, 3a, and 3b. S/he knows that the tuple of Bob must have been generalized to the first QI groups of Tables 1b and 3a, respectively. These groups encompass the same set of sensitive values {*dyspepsia*, *bronchitis*}. Therefore, the adversary cannot eliminate any disease that Bob cannot have contracted. Note that, although the adversary learns (from Table 3b) that a counterfeit exits in QI group 1 of Table 2a, s/he still cannot narrow down the possible diseases of Bob. In fact, to the adversary, there is a 50% chance that the first tuple of Table 2a would be the counterfeit.

The two releases (Tables 1b and 3a) have an important property. If a tuple appears in the microdata at both publication timestamps, it is generalized to two QI groups (one per timestamp) containing the same sensitive values. For instance, the tuple ⟨Jane, 37, 33k, *dyspepsia*⟩ belongs to both Tables 1a and 2a. It is generalized to QI groups 4 and 3 in Tables 1b and 3a, respectively. The two groups include an equivalent set of diseases: {*dyspepsia*, *flu*, *gastritis*} (as is achieved via a counterfeit c_2). As a result, even if an adversary finds out both QI groups, s/he can only conjecture that Jane's disease may be an element in that equivalent set.

Indeed, the key to privacy preserving re-publication is to ensure certain "invariance" in all the QI groups that a tuple is generalized to in different snapshots. In this paper, we establish this finding through a systematic study of the re-publication problem. First, we formalize several important concepts that constitute the foundation of investigating privacy disclosure in re-publication. Our formalization captures all the existing generalization schemes (such as full domain/subtree generalization, single-/multi-dimension recoding, global/local recoding; see [12] for the semantics of these schemes), and generalization principles (e.g., k-anonymity, l-diversity, etc.).

As the second step, we present a careful analysis on the theory of privacy protection. Specifically, we elaborate how an adversary can reconstruct the microdata by combining all the published tables. This allows us to calculate the risk of privacy disclosure in an educated manner. The resulting formulae explain the failure of the previous generalization principles, and lead to the development of m-invariance. We show that the new principle effectively limits the disclosure risk.

Finally, we design an efficient algorithm for computing publishable relations that conform to m-invariance. Our algorithm maximizes the utility of the released data, by minimizing (i) the number of counterfeit tuples, and (ii) the amount of generalization on the QI attributes. Furthermore, the algorithm is *incremental*, namely, it enables the publisher to complete the n-th publication, by consulting only the data of the last release. As a result, information related to the j-th publication (for any $1 \leq j \leq n-2$) does not need to be retained.

The rest of the paper is organized as follows. Section 2 formalizes the underlying concepts and the re-publication problem. Section 3 presents our theoretical results on privacy preservation. Section 4 proposes an algorithm for finding m-invariant generalization. Section 5 experimentally demonstrates the inadequacy of k-anonymity and l-diversity, and the effectiveness of the proposed technique. Section 6 reviews the previous research related to ours. Section 7 concludes the paper with directions for future work.

2. FUNDAMENTAL DEFINITIONS

Let T be a microdata table maintained by the publisher. We classify the columns of T into: (i) an identifier attribute A^{id}, which is the primary key of T, (ii) d quasi-identifier (QI) attributes A_1^{qi}, ..., A_d^{qi}, and (iii) a sensitive attribute A^s. Following the literature's convention [15], we require A^s to be categorical, while the other attributes can be either numerical or categorical. For each tuple t, $t[A]$ denotes its value on attribute A.

As time evolves, T is updated with insertions and deletions, which can arrive in any order. The publisher may release an anonymized version of T at any time, as long as it is possible to do so without compromising privacy. We use an integer j to denote the timestamp of the j-th publication.

Let $T(j)$ be the snapshot of T at time j. The publisher releases a pair of relations $\{T^*(j), R(j)\}$, where $T^*(j)$ anonymizes $T(j)$, and $R(j)$ is an auxiliary table providing some statistics about $T^*(j)$. In particular, anonymization is achieved through *counterfeited generalization*. Before formalizing this new concept, we clarify several basic notions:

DEFINITION 1 (QI GROUP / PARTITION). *For a microdata table $T(j)$, a* **QI group** *is a subset of the tuples in $T(j)$. A* **partition** *of $T(j)$ consists of disjoint QI groups whose union equals $T(j)$. Each QI group is assigned an ID that is unique in the partition.*

For a tuple $t \in T(j)$, $t.QI(j)$ denotes the QI group that contains t. We refer to $t.QI(j)$ as the **hosting group** *of t in $T(j)$.*

We are ready to formulate the $T^*(j)$ published at time j.

DEFINITION 2 (COUNTERFEITED GENERALIZATION). *The anonymized version $T^*(j)$ of $T(j)$ is computed based on a partition of $T(j)$, and has the following properties:*

1. *$T^*(j)$ contains a column A^g named "Group-ID", and all attributes in $T(j)$ except A^{id}.*
2. *Each tuple $t \in T(j)$ has a* **generalized tuple** *$t^* \in T^*(j)$ such that $t^*[A^s] = t[A^s]$, $t[A^g]$ is the ID of the hosting group of t in $T(j)$, and $t^*[A_i^{qi}]$ $(1 \leq i \leq d)$ is an interval[1] covering $t[A_i^{qi}]$. The value of $t^*[A_i^{qi}]$ also satisfies Property 4.*
3. *For each QI group QI of $T(j)$, $T^*(j)$ may contain any number of* **counterfeit tuples** *t_c^* such that $t_c^*[A^s]$ is a value in the domain of A^s, $t_c^*[A^g]$ equals the ID of QI, and $t^*[A_i^{qi}]$ $(1 \leq i \leq d)$ is an interval subject to Property 4.*
4. *All tuples in $T^*(j)$ with the same A^g have an identical value on every QI attribute. These tuples form a* **QI group** *in $T^*(j)$ whose ID is the A^g value in the group.*

For a tuple $t \in T(j)$, $t.QI^*(j)$ denotes the QI group in $T^*(j)$ that contains the generalized tuple of t. We refer to $t.QI^*(j)$ as the **generalized hosting group** *of t in $T^*(j)$.*

Clearly, counterfeited generalization captures existing generalization schemes in the literature as special cases, when there is no counterfeit. The next definition clarifies the $R(j)$ released along with $T^*(j)$:

DEFINITION 3 (AUXILIARY RELATION). *The* **auxiliary relation** *$R(j)$ accompanying $T^*(j)$ has two columns "Group-ID" and "Count". For each QI group QI^* in $T^*(j)$ that contains at least a counterfeit, there is a row $\langle g, c\rangle$ in $R(j)$, where g is the ID of QI^* and c is the number of counterfeit tuples in QI^*.*

$R(j)$ is empty, if no counterfeit is present in $T^*(j)$. As explained in Section 5.2, the counterfeit information in $R(j)$ is necessary for a researcher to derive accurate understanding about the microdata from the published data.

EXAMPLE 1. We illustrate the previous definitions by setting $T(j)$ to Table 2a, $T^*(j)$ to Table 3a, and $R(j)$ to Table 3b. Consider t as the tuple ⟨Bob, 21, 12k, *dyspepsia*⟩ in $T(j)$. Its hosting group $t.QI(j)$ consists of the first two rows of $T(j)$. Its generalized tuple in $T^*(j)$ is ⟨1, [21, 22], [12k, 14k], *dyspepsia*⟩. The generalized hosting group $t.QI^*(j)$ of t contains the two rows with *Group-ID* = 1 in $T^*(j)$. In particular, the tuple c_1 in $t.QI^*(j)$ is a counterfeit, which does not have any corresponding tuple in $T(j)$. There is another counterfeit c_2 in the QI group with ID 3. $R(j)$ summarizes the number of counterfeits in each QI group. ■

[1]To make the interval well-defined on a categorical A_i^{qi}, we transform A_i^{qi} into a numerical attribute, by placing a total ordering on its values. When there is a generalization hierarchy on A_i^{qi} (a common assumption in the literature [12]), this ordering lists all the leaves of the hierarchy from left to right.

DEFINITION 4 (GENERALIZATION PRINCIPLE). *A* **generalization principle** *is a set of constraints that must be satisfied by the QI groups in $T^*(1)$, ..., $T^*(n)$.*

For instance, k-anonymity and l-diversity are two generalization principles. Specifically, the former imposes the constraint that each QI group in $T^*(j)$ ($1 \leq j \leq n$) must have at least k tuples; the latter demands that the sensitive values in each QI group are well-represented. In general, a generalization principle may also require that the QI groups of multiple $T^*(j)$ (with different $j \in [1, n]$) should jointly fulfill certain conditions.

When the publisher is preparing $\{T^*(n), R(n)\}$ (where $n \geq 1$), it must take into account the information that an adversary can combine with $T^*(n)$ to intrude privacy. Apparently, such information includes all the data in $T^*(1)$, ..., $T^*(n-1)$ released previously. Furthermore, the adversary may also possess "background knowledge" that does not exist in those relations. To formalize such knowledge, we first introduce a notation $U(n)$:

DEFINITION 5 (HISTORICAL UNION). *At time $n \geq 1$, the* **historical union** *$U(n)$ contains all the tuples in T at timestamps 1, 2, ..., n, respectively. Formally:*

$$U(n) = \bigcup_{j=1}^{n} T(j). \tag{1}$$

Each tuple $t \in U(n)$ is implicitly associated with a **lifespan** *$[x, y]$, where x (y) is the smallest (largest) integer j such that t appears in $T(j)$.*

$U(n)$ can be regarded as a table with the same schema as T. Note that, if a tuple appears in several $T(j)$ with different j, it is included only once in $U(n)$. Now we can define the background knowledge that can be tackled by our technique.

DEFINITION 6 (PRIOR KNOWLEDGE). *At time n, an adversary's* **prior knowledge** *includes*

- *the deployed generalization principle, and*
- *a* **knowledge table** *$B(n)$, which has a column A^g named "Group-ID", a column A^l named "Lifespan", and all the attributes of $U(n)$ except A^s.*

  For every tuple $t \in U(n)$, there is a row $b \in B(n)$ such that $b[A^g] = *$, $b[A^{id}] = t[A^{id}]$, $b[A_i^{qi}] = t[A_i^{qi}]$ for all $1 \leq i \leq d$, and $b[A^l]$ equals the lifespan of t.

  For every counterfeit t_c in each $T^*(j)$ (where $1 \leq j \leq n$), there is a row $b_c \in B(n)$ such that $b_c[A^g] = t_c[A^g]$, $b_c[A^{id}]$ is any counterfeit identification unique in $B(n)$, $b_c[A_i^{qi}] = \emptyset$ for all $1 \leq i \leq d$, and $b_c[A^l] = [j, j]$.

Equivalently, $B(n)$ incorporates (i) everything in $U(n)$ except column A^s, (ii) the lifespan of each tuple in $U(n)$, and (iii) the published details of all the counterfeits.

EXAMPLE 2. We explain Definitions 5 and 6 by assuming $n = 2$, and that Tables 1a, 1b, 2a, 3a, and 3b are $T(1)$, $T^*(1)$, $T(2)$, $T^*(2)$, and $R(2)$ respectively.

$U(2)$ includes all the tuples in $T(1)$ and $T(2)$, after eliminating the duplicates. The lifespan of the tuple ⟨Bob, 21, 12k, *dyspepsia*⟩ is $[1, 2]$, since it remains in T during the entire history. On the other hand, the lifespan of ⟨Alice, 22, 14k, *bronchitis*⟩ is $[1, 1]$, because the tuple is inserted at time 1 and deleted at time 2.

Table 4 demonstrates part of the knowledge table $B(n)$. As an example, for tuple t = ⟨Bob, 21, 12k, *dyspepsia*⟩ in $U(2)$, the corresponding row $b \in B(n)$ is the first tuple of Table 4. b[*Group-ID*]=$*$ means that the adversary is not sure about the generalized hosting groups of t. In general, for *every* individual involved in $U(n)$, the adversary knows her/his identity, exact QI particulars, and which of the published relations contain her/his record.

Group-ID	Name	Age	Zip	Lifespan
$*$	Bob	21	12000	[1, 2]
$*$	Alice	22	14000	[1, 1]
...	...	...	...	...
$*$	Helen	36	27000	[1, 1]
$*$	David	23	25000	[2, 2]
...	...	...	...	...
1	c_1	$\emptyset$	$\emptyset$	[2, 2]
3	c_2	$\emptyset$	$\emptyset$	[2, 2]

Table 4: Adversaries' background knowledge

Let t_c be the counterfeit c_1 in Table 3a. Its corresponding row $b_c \in B(n)$ is the last-but-one tuple in Table 4. The adversary knows b_c[*Group-ID*] = 1 because this value is explicitly indicated in $R(2)$. The QI values of b_c are $\emptyset$, since, in general, counterfeits do not have any "original QI values" before generalization. The lifespan of every counterfeit covers only a single timestamp. For instance, b_c[*Lifespan*] implies that t_c is in $T^*(2)$. ∎

Notice that, if we eliminate *Group-ID* and *Lifespan*, $B(n)$ degenerates into an external database (most popularly, a voter registration list) commonly assumed [18, 20] as the adversary's prior knowledge, for carrying out privacy attacks on a k-anonymous/l-diverse relation. In reality, an adversary's knowledge is typically much weaker than that represented in $B(n)$. In the application context of Table 1, for instance, $B(n)$ essentially includes (i) the names, ages and Zipcodes of *all* patients, and (ii) the exact dates of their visits to the hospital! In other words, by guarding against attacks leveraging the background knowledge in Definition 6, we aim at privacy preservation under more hostile circumstances than would be encountered in practice.

Observe that all the information in $R(1)$, ..., $R(n)$ has been captured by $B(n)$. Thus, we formulate privacy disclosure as follows:

DEFINITION 7 (PRIVACY BREACH). *A* **privacy breach** *occurs if an adversary correctly finds out the sensitive value of any tuple $t \in U(n)$, utilizing $T^*(1)$, ..., $T^*(n)$, and $B(n)$.*

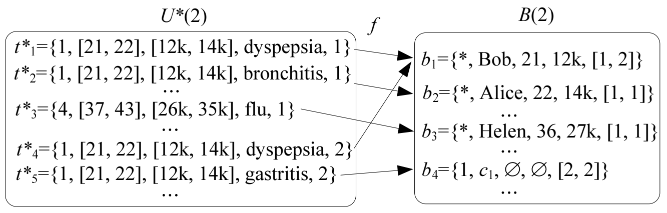
For instance, in Example 2, it is a privacy breach, if an adversary can reconstruct the sensitive value *dyspepsia* in tuple t = ⟨Bob, 21, 12k, *dyspepsia*⟩ $\in U(2)$, from Tables 1b, 3a, and 4.

We close this section with an inductive definition of the re-publication problem.

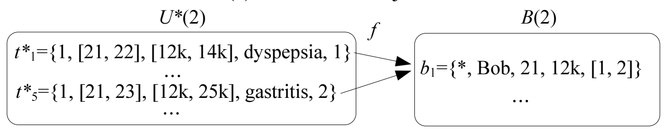
DEFINITION 8 (RE-PUBLICATION). *Assume that the publisher has released $n - 1$ anonymized versions of the microdata T: $\{T^*(1), R(1)\}$ ..., $\{T^*(n-1), R(n-1)\}$, where n is an integer ≥ 1, and $\{T^*(j), R(j)\}$ ($1 \leq j \leq n-1$) is defined in Definitions 2 and 3. The objective of* **privacy preserving re-publication** *is to compute a pair of $\{T^*(n), R(n)\}$ that minimizes the risk of privacy disclosure, yet captures as much information in the microdata as possible.*

3. THEORY OF PRIVACY PROTECTION

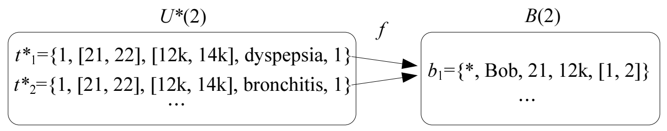
Before releasing $T^*(n)$, the publisher must guarantee that the privacy of every tuple in $U(n)$ has been adequately protected. This section provides the underlying theory towards that objective. In Section 3.1, we quantify the risk of privacy disclosure. Then, Section 3.2 establishes the importance of ensuring "persistent invariance" in re-publication, which leads to the development of a novel generalization principle in Section 3.3.


U*(2)
f
B(2)
t*1={1, [21, 22], [12k, 14k], dyspepsia, 1}
t*2={1, [21, 22], [12k, 14k], bronchitis, 1}
t*3={4, [37, 43], [26k, 35k], flu, 1}
t*4={1, [21, 22], [12k, 14k], dyspepsia, 2}
t*5={1, [21, 22], [12k, 14k], gastritis, 2}
b1={*, Bob, 21, 12k, [1, 2]}
b2={*, Alice, 22, 14k, [1, 1]}
b3={*, Helen, 36, 27k, [1, 1]}
b4={1, c1, ∅, ∅, [2, 2]}

(a) Reasonable surjection


U*(2)
f
B(2)
t*1={1, [21, 22], [12k, 14k], dyspepsia, 1}
t*5={1, [21, 23], [12k, 25k], gastritis, 2}
b1={*, Bob, 21, 12k, [1, 2]}

(b) Unreasonable surjection violating Condition 1.1 of Definition 13


U*(2)
f
B(2)
t*1={1, [21, 22], [12k, 14k], dyspepsia, 1}
t*2={1, [21, 22], [12k, 14k], bronchitis, 1}
b1={*, Bob, 21, 12k, [1, 2]}

(c) Unreasonable surjection violating Condition 1.2

Figure 1: Microdata reconstruction from surjective functions

3.1 Microdata Reconstruction

We will explain how an adversary can reconstruct the microdata tables $T(1)$, ..., $T(n)$ from the published $T^*(1)$, ..., $T^*(n)$ and the knowledge table $B(n)$. For this purpose, we need a generalized counterpart of Definition 5:

DEFINITION 9 (GENERALIZED HISTORICAL UNION). *Given a generalized relation $T^*(j)$ ($1 \le j \le n$), we convert each row $t^* \in T^*(j)$ to a* **timestamped tuple** *$\langle t^*, j\rangle$, which augments t^* with another attribute A^{tm}, called "Timestamp", storing j.*

The **generalized historical union** *$U^*(n)$ includes all the timestamped tuples converted from $T^*(1)$, ..., $T^*(n)$, or formally:*

$$U^*(n) = \bigcup_{j=1}^{n} \Big(\bigcup_{t^* \in T^*(j)} \langle t^*, j\rangle \Big). \tag{2}$$

Note that counterfeits in $T^*(j)$ ($1 \le j \le n$) also have converted tuples in $U^*(n)$. Regarding $U^*(n)$ and $B(n)$ as two sets, next we define a class of surjective functions from $U^*(n)$ to $B(n)$.

DEFINITION 10 (REBUILDING SURJECTION). *Mapping f: $U^*(n) \to B(n)$ is a* **rebuilding surjective function** *if it fulfills these requirements:*

1. *it maps each tuple $t^* \in U^*(n)$ to a row $b \in B(n)$, which is represented as $f(t^*) = b$;*
2. *for any row $b \in B(n)$, there exists at least a tuple $t^* \in U^*(n)$ such that $f(t^*) = b$;*
3. *if $f(t^*) = b$, then*
   - 3.1. *$b[A^l]$ contains $t^*[A^{tm}]$;*
   - 3.2. *$t^*[A^g] = b[A^g]$ (the equality always holds if $b[A^g] = *$);*
   - 3.3. *$t^*[A^{qi}_i]$ covers $b[A^{qi}_i]$ along every QI attribute A^{qi}_i (the covering relationship always holds if $b[A^{qi}_i] = \emptyset$).*

Conditions 1 and 2 constitute the standard mathematical definition of surjection. Conditions 3.1-3.3 will be explained with a concrete example.

EXAMPLE 3. We illustrate Definitions 9 and 10 by setting n to 2, $T^*(1)$ and $T^*(2)$ to Tables 1b and 3a respectively, and $B(2)$ to Table 4.

The left box in Figure 1a encloses five timestamped tuples t^*_1, ..., t^*_5 in the generalized historical union $U^*(2)$. For instance, t^*_1 = $\langle$1, [21, 22], [12k, 14k], *dyspepsia*, 1$\rangle$ augments the first tuple in $T^*(1)$ with timestamp 1, that is, $t^*_1[A^{tm}] = 1$. Similarly, t^*_4 augments the first tuple in $T^*(2)$ with timestamp 2. The right box contains four rows b_1, ... b_4 from $B(2)$, including a counterfeit b_4.

The arrows depict five mappings in a surjective function f, namely, $f(t^*_1) = b_1$, $f(t^*_2) = b_2$, $f(t^*_3) = b_3$, $f(t^*_4) = b_1$ and $f(t^*_5) = b_4$. Mapping $f(t^*_1) = b_1$, for example, qualifies Condition 3.1 in Definition 10 because $b_1[A^l]$ = [1, 2] encloses $t^*_1[A^{tm}]$ = 1. It satisfies Condition 3.2 as $b_1[A^g]$ = *. Finally, it fulfills Condition 3.3 since t^*_1[*Age*] = [21, 22] and t^*_1[*Zipcode*] = [12k, 14k] contain b_1[*Age*] = 21 and b_1[*Zipcode*] = 12k, respectively. ■

Rebuilding surjection can be used to reconstruct a possible version of the microdata tables $T(1)$, ..., $T(n)$:

DEFINITION 11 (POSSIBLE MICRODATA INSTANCE). *Let f be a function in Definition 10. For each mapping $f(t^*) = b$ such that b^{id} is not counterfeit identification, we first obtain the timestamp $j = t^*[A^{tm}]$, and then reconstruct a tuple in $T(j)$ as*

$$\langle b[A^{id}], b[A^{qi}_1], ..., b[A^{qi}_d], t^*[A^s]\rangle. \tag{3}$$

All the reconstructed tuples constitute a **possible microdata instance**.

The instance rebuilt from f is a possible version of the microdata that can result in $T^*(1)$, ..., $T^*(n)$ through counterfeited generalization (Definition 2), as elaborated next.

DEFINITION 12 (POSSIBLE GENERALIZATION INSTANCE). *Let f be a rebuilding surjective function. For any row $b \in B$, $f^{-1}(b)$ denotes the set of tuples $t^* \in U^*(n)$ satisfying $f(t^*) = b$.*

Mapping f^{-1}: $B(n) \to U^*(n)$ determines a counterfeited generalization process of obtaining $T^*(1)$, ..., $T^*(n)$ from the possible microdata instance rebuilt by f. Specifically, given each $b \in B(n)$, we distinguish two cases:

- *[b^{id} is personal identification] For every $t^* \in f^{-1}(b)$, regard t^* as the generalized tuple in $T^*(j)$ of the microdata tuple in Formula 3 (reconstructed from mapping $f(t^*) = b$), where $j = t^*[A^{tm}]$.*
- *[b^{id} is counterfeit identification] For every $t^* \in f^{-1}(b)$, regard t^* as a counterfeit tuple in $T^*(j)$ where $j = t^*[A^{tm}]$.*

The above process is a **possible generalization instance**.

EXAMPLE 3 (CONTINUED). We explain Definitions 11 and 12 using Figure 1a. The meanings of t^*_1, ..., t^*_5, b_1, ..., b_4 are as mentioned in Example 3.

The five mappings (indicated by arrows) reconstruct four tuples in the microdata tables $T(1)$ and $T(2)$ (no reconstruction from $f(t^*_5) = b_4$, as $b_4[A^{id}] = c_1$ is counterfeit identification). For example, $f(t^*_1) = b_1$ implies taking the QI values in b_1 as the original forms of those in t^*_1 before generalization. This way, we reconstruct a tuple $\langle$Bob, 21, 12k, *dyspepsia*$\rangle$ in $T(1)$, conforming to Formula 3. Similarly, $f(t^*_4) = b_1$ rebuilds the same tuple in $T(2)$; $f(t^*_2) = b_2$ indicates a tuple $\langle$Alice, 22, 14k, *bronchitis*$\rangle$ in $T(2)$.

Reconstruction is not always accurate. For instance, $f(t^*_3) = b_3$ gives $\langle$Helen, 36, 27k, *flu*$\rangle$ in $T(1)$, but the real disease of Helen is *gastritis*. In other words, for each $j \in [1, n]$, function f decides only a possible, but not necessarily the actual, $T(j)$.

We proceed to illustrate the counterfeited generalization determined by f^{-1}. $f^{-1}(b_1)$ equals $\{t_1^*, t_4^*\}$. Hence, by Definition 12, t_1^* is the generalized tuple of ⟨Bob, 21, 12k, *dyspepsia*⟩ in $T^*(1)$, which, as mentioned earlier, is restored from $f(t_1^*) = b_1$. By the same reasoning, t_4^* is the generalized tuple of ⟨Bob, 21, 12k, *dyspepsia*⟩ in $T^*(2)$. As another example, consider $f^{-1}(b_4) = \{t_5^*\}$. Since $b_4[A^{id}]$ is counterfeit identification c_1, t_5^* is a counterfeit tuple added to $T^*(2)$ in the generalization process. ■

There is a *genuine surjective function*, which exactly reconstructs the original microdata. If the adversary *were* able to discover this surjection, s/he would reveal the sensitive information of all individuals. Fortunately, as the tuples in $U^*(n)$ have been generalized, there exist a huge number of possible surjective functions from $U^*(n)$ to $B(n)$. The best the adversary can do is to eliminate as many "unreasonable" functions as possible, in order to increase her/his chance of identifying the genuine surjection.

DEFINITION 13 (REASONABLE SURJECTION). *A function f in Definition 10 is* **reasonable**, *if it satisfies these conditions:*

1. *the following holds for each row $b \in B(n)$: Given the set $f^{-1}(b)$ of tuples $t^* \in U^*(n)$ satisfying $f(t^*) = b$, then*
   - 1.1. *all tuples in $f^{-1}(b)$ carry the same sensitive value;*
   - 1.2. *for any timestamp j in the lifespan $b[A^l]$ of b, there exists a unique tuple $t^* \in f^{-1}(b)$ with $t^*[A^{tm}] = j$.*
2. *f^{-1} decides a possible generalization instance that conforms to the deployed generalization principle (Definition 4).*

EXAMPLE 4. Condition 2 in Definition 13 is straightforward. Next we clarify Conditions 1.1 and 1.2 by assuming the same n, $T^*(1)$, $T^*(2)$ and $B(2)$ as in Example 3.

Consider the surjection in Figure 1b, where tuples t_1^*, t_5^* and b_1 are identical to those in Figure 1a. In the reconstructed possible microdata instance, Bob has two different records in $T(1)$ and $T(2)$, carrying sensitive values *dyspepsia* and *gastritis*, respectively. This contradicts the semantic of b_1, whose lifespan $[1, 2]$ indicates that Bob should have the same tuple in $T(1)$ and $T(2)$. Thus, the surjection is unreasonable, which is captured by Definition 13. Specifically, both t_1^* and t_5^* belong to $f^{-1}(b_1)$; hence, Condition 1.1 is violated.

The surjection in Figure 1c is also unreasonable. Here, $f^{-1}(b_1)$ equals $\{t_1^*, t_2^*\}$ (both tuples have timestamp 1), which breaches Condition 1.2 for two reasons. First, more than one tuple in $T(1)$ is mapped to b_1. This is impossible, because Bob (in general, any individual) has at most one record in the microdata at time 1. Second, no tuple in $T(2)$ is mapped to b_1, contradicting the fact that Bob has a record at time 2. ■

Naturally, we arrive at the following formulation of privacy leakage risk.

DEFINITION 14 (PRIVACY DISCLOSURE RISK). *Let t be a tuple in the historical union $U(n)$. The* **privacy disclosure risk** *$risk(t)$ of t is represented as*

$$risk(t) = n_{breach}(t)/n_{total}, \tag{4}$$

where n_{total} is the number of reasonable surjective functions, and $n_{breach}(t)$ is the number of those functions that correctly reconstruct the sensitive value of t.

A surjective function may correctly reconstruct the sensitive value of a tuple, but at the same time, incorrectly reconstruct that of another. For instance, the surjection in Figure 1a precisely restores the disease *dyspepsia* of Bob, but indicates a wrong disease *flu* of Helen, as explained in Example 3. Therefore, various tuples in $U(n)$ can have different privacy disclosure risks.

Specially, if $n_{breach}(t) = n_{total}$ (equivalently, $risk(t) = 1$), an adversary (with the background knowledge in $B(n)$) will discover the true sensitive value of t with 100% confidence. In general, under the random-world assumption [4] where every reasonable surjection is equally likely, $risk(t)$ is the probability that an adversary can breach the privacy of t.

3.2 Persistent Invariance

In the sequel, we use the analysis in Section 3.1 to discuss the properties that should be possessed by privacy preserving re-publication. Our discussion utilizes a crucial concept:

DEFINITION 15 (CANDIDATE SENSITIVE SET). *Let t be a tuple in $U(n)$. The* **candidate sensitive set** *$t.CSS(j)$ of t at time j is the union of the sensitive values in each QI group QI^* of $T^*(j)$, such that $QI^*[A_i^{qi}]$ contains $t[A_i^{qi}]$ on every attribute A_i^{qi} $(1 \le i \le d)$, where $QI^*[A_i^{qi}]$ is the A_i^{qi} value in QI^*.*

The next result points out the tuples in $U(n)$ whose privacy is the most vulnerable in re-publication.

LEMMA 1. *Let t be a tuple in $U(n)$ with a lifespan $[x, y]$. Regardless of the generalization principle applied, $risk(t) = 1$, if only a single element exists in $t.CSS(x) \cap t.CSS(x + 1) \cap ... \cap t.CSS(y)$.*

PROOF. The proofs of all lemmas can be found in the appendix. □

EXAMPLE 5. We demonstrate the lemma, by setting n to 2, $T(1)$ and $T(2)$ to Tables 1a and 2a respectively, $T^*(1)$ to Table 1b, and $T^*(2)$ to Table 2b (note: *not* Table 3a).

Let t be the tuple ⟨Bob, 21, 12k, *dyspepsia*⟩, which belongs to $U(2)$ and has a lifespan $[1, 2]$. The candidate sensitive set $t.CSS(1)$ at time 1 includes the sensitive values *dyspepsia* and *bronchitis* in QI group 1 of $T^*(1)$. Similarly, $t.CSS(2)$ is $\{$*dyspepsia*, *gastritis*$\}$. Therefore, by Lemma 1, the privacy disclosure risk of t is 1. Namely, an adversary can precisely derive its real disease, confirming the discussion in Section 1.1. ■

Lemma 1 reveals the reason behind the failure of k-anonymity and l-diversity in re-publication: *neither generalization principle can prevent the situation stated in the lemma*. Each principle constraints only individual $t.CSS(j)$ (for $j \in [x, y]$), but ignores the relationships among these sets. Specifically, k-anonymity ensures that each $t.CSS(j)$ has a size at least k, while l-diversity guarantees that the sensitive values in every $t.QI^*(j)$ are well-represented. Nevertheless, it is still possible that only a single value appears in $\cap_{j=x}^{y} t.CSS(j)$.

A general version of Lemma 1 is:

LEMMA 2. *$\cap_{j=x}^{y} t.CSS(j)$ includes all the possible sensitive values of t, reconstructed from reasonable surjective functions.*

The lemma theoretically justifies our intuition: from $T^*(1)$, .., $T^*(n)$ and $B(n)$, an adversary can learn that the real sensitive value of t must fall in $\cap_{j=x}^{y} t.CSS(j)$. Hence, to protect privacy, re-publication must ensure a sufficiently large $\cap_{j=x}^{y} t.CSS(j)$ after every publication. In other words, an *invariant* set of sensitive values should *persistently* appear in $\cap_{j=x}^{y} t.CSS(j)$ at each publication timestamp, until t is deleted from the microdata. This *persistent-invariance* observation motivates a new generalization principle in the next subsection.

3.3 m-Invariance

We will need the following concept frequently.

DEFINITION 16 (SIGNATURE). *Let QI^* be a QI group in $T^*(j)$ for any $j \in [1, n]$. The* **signature** *of QI^* is the set of distinct sensitive values in QI^*.*

Note that if a sensitive value is carried by multiple tuples in QI^*, the value appears only once in the signature. Next, we propose *m-invariance*, which is the key to privacy protection in re-publication.

DEFINITION 17 (m-INVARIANCE). *A generalized table $T^*(j)$ $(1 \leq j \leq n)$ is* **m-unique***, if each QI group in $T^*(j)$ contains at least m tuples, and all the tuples in the group have different sensitive values.*

A sequence of published relations $T^*(1)$, ..., $T^*(n)$ (where $n \geq 1$) is **m-invariant** *if the following conditions hold:*

1. *$T^*(j)$ is m-unique for all $j \in [1, n]$.*
2. *For any tuple $t \in U(n)$ with lifespan $[x, y]$, $t.QI^*(x)$, $t.QI^*(x+1)$, ..., $t.QI^*(y)$ have the same signature, where $t.QI^*(j)$ is the generalized hosting group (see Definition 2) of t at time $j \in [x, y]$.*

m-uniqueness demands that each sensitive value should appear at most once in every QI-group. Apparently, m-uniqueness implies m-diversity, but not the vice versa. The rationale of m-invariance is that, if a tuple t (from the microdata) is published several times, all its generalized hosting groups must contain the same sensitive values. In Example 5, Tables 1b and 2b are 2-unique, but they do *not* constitute a 2-invariant sequence. To understand this, let t be the tuple of Bob in the microdata. $t.QI^*(1)$ and $t.QI^*(2)$ are the first QI-group in Tables 1b and 2b, respectively. The two QI-groups do not encompass the same sensitive values, thus violating the second condition of Definition 17. On the other hand, in Example 3, {Table 1b, Table 2b} is a 2-invariant sequence.

LEMMA 3. *If $\{T^*(1), ..., T^*(n)\}$ is m-invariant, then*

$$risk(t) \leq 1/m$$

for any $t \in U(n)$, where $risk(t)$ is given in Equation 4.

Therefore, the publisher can simply set m to a sufficiently large value to achieve the target extent of privacy preservation.

LEMMA 4. *If $\{T^*(1), ..., T^*(n-1)\}$ is m-invariant, then $\{T^*(1), ..., T^*(n-1), T^*(n)\}$ is also m-invariant if and only if:*

1. *$T^*(n)$ is m-unique;*
2. *for any tuple $t \in T(n-1) \cap T(n)$, its generalized hosting groups $t.QI^*(n-1)$ and $t.QI^*(n)$ have the same signature.*

Lemma 4 points to an incremental approach for performing re-publication. Specifically, to prepare $T^*(n)$, the publisher only needs to consult the microdata tables $T(n-1)$, $T(n)$, and the last released version $T^*(n-1)$. The older microdata tables $T(1)$, ..., $T(n-2)$, as well as their published counterparts $T^*(1)$, ..., $T^*(n-2)$, do not even need to be retained.

4. M-INVARIANT GENERALIZATION

This section elaborates computation of the $\{T^*(n), R(n)\}$ released at the n-th publication. We focus on $T^*(n)$ because once it is ready, producing $R(n)$ is trivial, as shown in Definition 3. Section 4.1 first describes our solution at a high level. Then, Sections 4.2 and 4.3 provide detailed explanation to two components of the solution. Finally, Section 4.4 clarifies the recoding scheme underlying our solution.

4.1 The Algorithm

We aim at achieving two intuitive goals. First, the number of counterfeit tuples should be minimized, because they do not correspond to any records in the microdata. Second, we use the least generalization to distort QI values. Specifically, for each tuple t, we attempt to reduce, as much as possible, the length of $t^*[A_i^{qi}]$ (for each $1 \leq i \leq d$), where t^* is the generalized tuple of t in $T^*(n)$. Clearly, a shorter $t^*[A_i^{qi}]$ implies less information loss.

We permit the n-th publication, only if $T(n) - T(n-1)$ is *m-eligible*, that is, at most $1/m$ of the tuples in $T(n) - T(n-1)$ have an identical sensitive value. Note that $T(n) - T(n-1)$ is essentially the set of new tuples in $T(n)$. For example, if $m = 10$, in the application of Table 1, the requirement is that at most 10% of the new patients in $T(n)$ contracted the same disease, which is fairly reasonable. Note that this publishability constraint actually already exists in the literature [15]: no "recursive $(\frac{1}{m-1}, 2)$-diverse" publication is possible, if the microdata is not m-eligible.

According to Lemma 4, calculation of $T^*(n)$ requires only microdata tables $T(n-1)$, $T(n)$, and the last published relation $T^*(n-1)$. Let us divide the tuples in $T(n)$ into two disjoint sets $S_\cap = T(n) \cap T(n-1)$ and $S_- = T(n) - T(n-1)$. Our algorithm ensures two properties:

1. For any tuple $t \in S_\cap$, its generalized hosting groups $t.QI^*(n-1)$ and $t.QI^*(n)$ have the same signature (see Definition 16).
2. For any tuple $t \in S_-$, its generalized tuple t^* in $T^*(n)$ is in a QI group which has at least m tuples, and all the tuples have distinct sensitive values.

By Lemma 4, these properties establish the correctness of $T^*(n)$.

We produce $T^*(n)$ in four phases: *division*, *balancing*, *assignment*, and *split*. The rest of this subsection elaborates each phase in turn. We use a running example where $m = 2$, $n = 2$, $T(1)$ and $T^*(1)$ are Tables 1a and 1b, respectively. Given $T(2)$ = Table 2a, we will show how $T^*(2)$ = Table 3a is computed.

Division. For each $t \in S_\cap$, we define its *signature* as the signature of its generalized hosting group in $T^*(n-1)$. This phase simply partitions $S_\cap$ into several *buckets*, such that each bucket contains only the tuples with the same signature.

In the running example, $S_\cap$ contains the tuples of Bob, David, Jane, Linda, Gary, and Steve. Figure 2a shows the contents of the buckets after this phase. The tuple of Bob, for example, has a signature {*dyspepsia*, *bronchitis*} (i.e., the sensitive values in Group 1 of Table 1b). It is the only element in bucket BUC_3. A bucket can have multiple tuples. For example, BUC_1 contains Gary and David, since they share an equivalent signature {*flu*, *gastritis*}.

Balancing. Unlike in the previous phase, we will work with tuples' sensitive values, as opposed to their signatures. We say that a bucket BUC is *balanced*, if every sensitive value in its signature is owned by the same number of tuples in BUC. For example, in Figure 2a, BUC_1 is balanced, since its signature has two values *flu* and *gastritis*, each of which is possessed by a tuple. The objective of this phase is to balance all buckets.

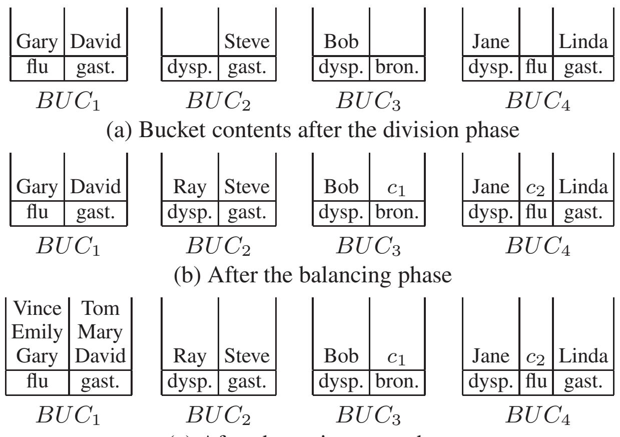


Figure 2: Illustration of our generalization algorithm

Each bucket BUC is inspected in turn. If BUC is not balanced, there is a "shortage" of some sensitive values in BUC. In this case, we attempt to fill the shortage by moving the tuples in S_- into BUC, *as long as the resulting S_- is still m-eligible* (the reason will be clear in a while).

In Figure 2a, BUC_2 is unbalanced, because there is one (tuple with) *gastritis* but no *dyspepsia*. S_- equals {Emily, Mary, Ray, Tom, Vince}. We can move Ray (whose *Disease* value is *dyspepsia*) to BUC_2, because there remain 2 *flu* and 2 *gastritis* in S_-, which is still 2-eligible. The updated BUC_2, shown in Figure 2b, becomes balanced.

If S_- cannot be used to fix an unbalanced bucket BUC, there are two possibilities: (i) no tuple in S_- carries the required sensitive value(s), or (ii) S_- is no longer m-eligible after a tuple removal. In both cases, we insert counterfeits to balance BUC.

Continuing our example in Figure 2a, both BUC_3 and BUC_4 are unbalanced, but neither of them can be remedied with S_-. Specifically, BUC_3 needs a *bronchitis*, which is absent in S_-. BUC_4 needs a *flu*; although there are tuples with *flu* in S_-, removing any of them leaves 2 *gastritis* and 1 *flu* in S_-, violating the 2-eligibility constraint. Therefore, as in Figure 2b, two counterfeits c_1 and c_2 (with sensitive values *bronchitis* and *flu*) are added to BUC_3 and BUC_4 respectively, both of which are now balanced. Recall that each counterfeit has a value $\emptyset$ on every QI attribute.

Assignment. In this phase, we assign the remaining tuples in S_- to buckets, subject to two rules. First, each tuple $t \in S_-$ can be placed only in a bucket whose signature includes $t[A^s]$. Second, at the end of the phase, all buckets are still balanced. If necessary, new buckets (each bucket's signature contains at least m values) may be generated, and they also obey these rules. As proved later, such an assignment scheme always exists, as long as S_- is m-eligible (which is why we insist on its m-eligibility in balancing).

In the running example, S_- = {Emily, Mary, Tom, Vince} after the balancing phase. Figure 2c illustrates the buckets after all assignments. The 4 tuples in S_- are all placed in BUC_1, which remains balanced. We will discuss assignment in detail in Section 4.2.

Split. This last phase processes each bucket BUC individually. It splits BUC into $|BUC|/s$ QI groups, where s ($\geq m$) is the number of values in the signature of BUC. Each group has s tuples, taking the s sensitive values in the signature, respectively.

Splitting optimizes the quality of generalization. Let $t_1, t_2, ..., t_s$ be the tuples in a group. Their generalized tuples form a QI group QI^* in the published $T^*(n)$. On each QI attribute A_i^{qi} ($1 \leq i \leq d$), $QI^*[A_i^{qi}]$ is the *minimum interval* enclosing all $t_1[A_i^{qi}], ..., t_s[A_i^{qi}]$. Therefore, splitting aims at minimizing the length sum of intervals $QI^*[A_1^{qi}], ..., QI^*[A_d^{qi}]$, as will be explained in Section 4.3.

Given BUC_1 in Figure 2c, our split algorithm creates three QI groups: {David, Emily}, {Gary, Mary}, and {Vince, Tom}. They lead to QI groups 2, 4, and 6 in Table 3a. Similarly, BUC_2, BUC_3, and BUC_4 result in QI groups 5, 1, 3, respectively.

A few last words concern the age [21, 22] of QI group 1 in Table 3a. This group covers the tuple of Bob (age 21) and a counterfeit (age $\emptyset$). We would have published 21, if the minimum-interval generalization were followed. In practice, however, personal particulars should not be released directly, for several reasons discussed in [16]. Hence, we require each QI value in $T^*(n)$ to be an interval whose length is at least a threshold (e.g., 2 for *Age*). This threshold may vary for different QI attributes (e.g., 2k for *Zipcode*).

4.2 The Assignment Phase

The algorithm of the assignment phase accepts as a parameter the set S_- passed from the previous phase, and runs in iterations. Each iteration moves a set S_{rmv} of $\alpha \cdot \beta$ tuples from S_- to a bucket BUC whose signature contains β ($\geq m$) sensitive values. BUC perhaps already exists (since it may have been generated in the balancing phase or an earlier iteration of this phase); otherwise, we create it. To keep BUC balanced, S_{rmv} must satisfy a property: every value in the signature of BUC should be possessed by exactly α tuples in S_{rmv}.

The crucial part is the selection of the integers α, β, and the signature of BUC. We aim at maximizing α to reduce the number of iterations. On the other hand, we should minimize β, since m-invariance can be enforced more easily on QI groups with less sizable signatures. The constraint is that, after eliminating the tuples in S_{rmv}, S_- must remain m-eligible, to make sure that all its remaining tuples can be eventually assigned (see the proof of Lemma 5).

The above observations motivate the following strategy. Let v_1, v_2, ..., v_λ be the distinct sensitive values in S_-. At the beginning of an iteration, we collect the number n_i ($1 \leq i \leq \lambda$) of tuples in the current S_- that have value v_i, and sort those numbers in descending order. The sorted order may vary in each iteration, and, without loss of generality, is assumed to be $n_1, n_2, ..., n_\lambda$. We use γ to denote $\sum_{i=1}^{\lambda} n_i$.

Now suppose that β has been determined (its determination will be clarified shortly). We choose the β most frequent sensitive values $v_1, ..., v_\beta$ to form the signature of BUC. For each $i \in [1, \beta]$, we randomly pick α tuples in S_- having the value v_i, and add them to S_{rmv}. Since there are only n_β tuples with value v_β, we have

$$\alpha \leq n_\beta. \tag{5}$$

After S_{rmv} is discarded, the *remaining* S_- has cardinality $\gamma - \alpha \cdot \beta$. In that S_-, (i) there are $n_1 - \alpha$ tuples with value v_1, and (ii) the most frequent sensitive value can only be either v_1 and $v_{\beta+1}$. For the remaining S_- to be m-eligible, we need

$$n_1 - \alpha \leq (\gamma - \alpha \cdot \beta)/m \tag{6}$$

$$n_{\beta+1} \leq (\gamma - \alpha \cdot \beta)/m. \tag{7}$$

α is set to the largest positive integer that satisfies Inequalities 5-7.

It remains to clarify the decision of β. Initially, we set it to the smallest possible value m, and attempt to solve α from the above three inequalities. If α exists, then the formulation of α and β is completed. Otherwise, β is increased by 1, and we again try to solve α. This process is repeated, until β reaches the first value that yields a solution of α. As proved in Lemma 5, the process always

```
Algorithm Assign ($S_-$)
1.  $\lambda$ = the number of distinct sensitive values in $S_-$
2.  while $S_-$ is not empty //start an interation
3.      $\gamma = |S_-|$
4.      obtain $n_1, n_2, ..., n_\lambda$ where $n_i$ ($1 \le i \le \lambda$) is the number of tuples
        having the $i$-th most frequent sensitive value $v_i$ in the current $S_-$
5.      $\beta = m$
6.      $\alpha$ = the largest positive integer $\alpha$ satisfying Inequalities 5-7
7.      if $\alpha$ does not exist
8.          $\beta = \beta + 1$; go to Line 6
9.      $BUC$ = a bucket whose signature is $\{v_1, ..., v_\beta\}$ (create $BUC$ if it
        does not exist yet)
10.     for $i$ = 1 to $\beta$
11.         randomly move $\alpha$ tuples with value $v_i$ from $S_-$ to $BUC$
```

Figure 3: The assignment algorithm

terminates with a pair of appropriate α and β. Figure 3 formally presents the assignment algorithm.

LEMMA 5. *If S_- is m-eligible, the algorithm in Figure 3 assigns all tuples in S_- to balanced buckets.*

4.3 The Split Phase

Let BUC be a balanced bucket output by the assignment phase, whose signature has $s \ge m$ sensitive values $v_1, v_2, ..., v_s$. The split phase starts by initiating a set $S_{buc} = \{BUC\}$. If BUC includes more than s tuples, we remove it from S_{buc}, and split (the tuples in) BUC into two balanced buckets BUC_1 and BUC_2 with the same signature as BUC. BUC_1 and BUC_2 are then added to S_{buc}.

If any bucket in S_{buc} still has a size over s, we set BUC to that bucket, and repeat the above procedures. The phase terminates when all the buckets in S_{buc} contain exactly s tuples. They are returned as the QI groups for generalization, as discussed in Section 4.1. Totally $\frac{|BUC|}{s} - 1$ bucket splits are performed (notice that the cardinality of each bucket in S_{buc} is always a multiple of s).

Next we clarify the details of splitting BUC. We organize BUC into s groups, such that the j-th ($1 \le j \le s$) group contains only the tuples with the sensitive value v_j. Clearly, every group has size $|BUC|/s$. Then, the tuples in each group are sorted in ascending order of their A_1^{qi} values ($\emptyset$ precedes all non-empty values in sorting). Let us deploy L_j to denote the sorted list of the j-th ($1 \le j \le s$) group.

Suppose that we include the first tuple of each L_j ($1 \le j \le s$) into BUC_1, and the other tuples into BUC_2. This determines a *split scheme*. Similarly, we can obtain an alternative scheme by placing the first 2 (or 3, 4, ..., $|BUC|/s - 1$) tuples of each L_j into BUC_1, and the rest into BUC_2. This way, $s - 1$ different schemes have been defined. Recall that L_1, ..., L_s were computed based on attribute A_1^{qi}. By generating these sorted lists with respect to each other A_i^{qi} ($2 \le i \le d$), we derive another $s - 1$ schemes in the same manner.

Among all the $d \cdot (|BUC|/s - 1)$ schemes, we pick the "best" one as the final split, which minimizes the perimeter sum of BUC_1 and BUC_2. Specifically, the *perimeter* of BUC_1 equals $|BUC_1| \cdot \sum_{i=1}^{d} l_i$, where l_i ($1 \le i \le d$) is the length of the minimum interval, which encloses the A_i^{qi} values of all tuples in BUC_1. The perimeter of BUC_2 is defined symmetrically. To make the lengths of minimum intervals along different A_i^{qi} comparable, we normalize the domain of each A_i^{qi} to $[0, 1]$.

4.4 Discussion

The QI-groups output by our algorithm conform to the *local-recoding* generalization scheme [12], in contrast to the *global-recoding* scheme often adopted in the literature. We employ the local-recoding scheme because it provides higher flexibility in

attribute	Age	Gender	Education	Birthplace	Occupation	Income
domain size	79	2	17	57	50	50

Table 5: Attribute domain sizes

forming QI-groups. Apparently, the concept of m-invariance can also be implemented under global-recoding, although it demands the development of another generalization algorithm. In any case, the choice of a scheme is not important, as long as the published information has good utility. In the next section, we will show that the anonymized data produced by our technique permit accurate aggregate analysis.

5. EXPERIMENTS

All the experiments are performed on a machine running a 3Ghz CPU with 1 Giga-byte memory. We deploy two real databases OCC and SAL downloadable from *http://ipums.org*. Each database contains 600k tuples, each storing the information of an American adult. OCC includes four QI attributes, *Age*, *Gender*, *Education*, and *Birthplace*, and a sensitive attribute *Occupation*. SAL contains the same QI attributes, but a different sensitive attribute *Income*. All columns are discrete, and the sizes of their domains are given in Table 5.

A dynamic microdata table T_{occ} (T_{sal}) is created from OCC (SAL). It suffices to clarify the generation of T_{occ}, since the same method is used for T_{sal}. The first version $T_{occ}(1)$ contains 200k tuples randomly sampled from OCC. We initiate a *pool* that contains the other 400k tuples in OCC. At the j-th ($j \ge 2$) timestamp, $T_{occ}(j)$ is obtained by arbitrarily deleting r tuples from $T_{occ}(j-1)$, and then inserting the same number of tuples randomly removed from *pool*. Here, r is a parameter, called *update volume*, controling the update rate. We repeat this process up to timestamp $H = 1 + 400\text{k}/r$ (e.g., for r = 20k, the history has totally H = 21 timestamps).

We refer to the sequence $\{T_{occ}(1), T_{occ}(2), ..., T_{occ}(H)\}$ as a T_{occ}*-series* (the concept of T_{sal}*-series* is defined similarly). Such a series is characterized by the parameter r.

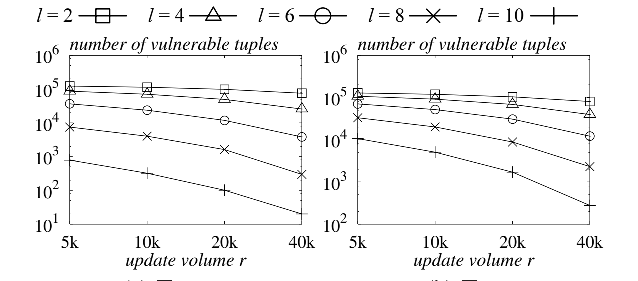
5.1 Failure of Conventional Generalization

In the first set of experiments, we aim at establishing our conjecture that the existing generalization principles may lead to severe privacy disclosure in re-publication. We use l-diversity as the representative principle, since it is widely adopted and offers stronger protection than k-anonymity.

Given a T_{occ}-series, we adopt the algorithm in [13] to compute an l-diverse version $T^*_{occ}(j)$ of each $T_{occ}(j)$ for all $j \in [1, H]$. Then, we capture all the tuples (that ever appear in T_{occ}) whose sensitive values will definitely be revealed, i.e., their privacy receives no protection at all. These *vulnerable tuples* are extracted using Lemma 1. Note that, a generalization principle can be used for re-publication, only if no vulnerable tuple can ever be found.

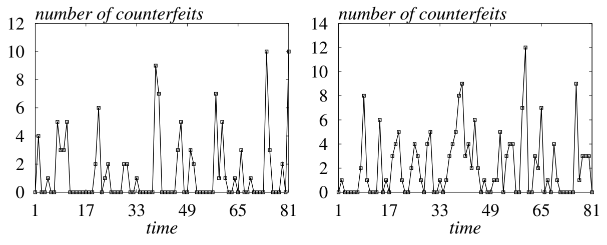
In Figure 4a, we plot the number of vulnerable tuples as a function of r, as this parameter changes from 5k to 40k. Each curve corresponds to the result obtained with a different l (varied from 2 to 10).

Obviously, l-diversity fails to support re-publication, because it results in a large number of vulnerable tuples. For example, for $l = 2$, there are nearly 100k tuples whose privacy is not preserved at all, regardless of r. Although fewer vulnerable tuples exist as l increases, they still cannot be completely prevented even with the largest $l = 10$. The number of vulnerable tuples decreases as r grows. This is because, for a larger r, each tuple in the microdata is published fewer times, and thus, has a lower chance of becoming vulnerable (in the extreme case where each tuple is released



(a) T_{occ} (b) T_{sal}

Figure 4: Number of vulnerable tuples vs. the update volume r



(a) T_{occ} (b) T_{sal}

Figure 5: Number of counterfeits vs. time (r = 5k, m = 10)

only once, no tuple is vulnerable). We repeat the experiment on T_{sal}. The results are illustrated in Figure 4b, confirming the same observations.

5.2 m-Invariance Evaluation

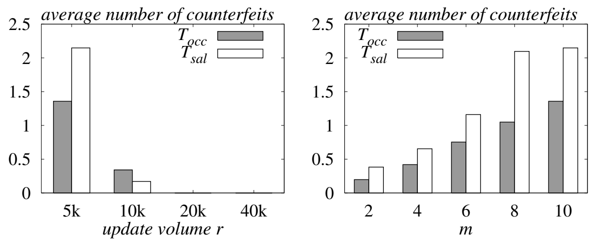
In the sequel, we examine the effectiveness and efficiency of m-invariant generalization. Given a T_x-series (where x = *occ* or *sal*), the counterfeited generalization (CG) algorithm in Section 4 is invoked to compute the generalized relations $T_x^*(1)$, $T_x^*(2)$, ..., $T_x^*(H)$ for m-invariant publication. We call the sequence $\{T_x^*(1)$, ..., $T_x^*(H)\}$ a T_x^**-series*, which is characterized by parameters r and m.

Number of Counterfeits. We start by demonstrating that only a small number of counterfeits are needed to enforce m-invariance. We first deploy the T_{occ}^*-series with r = 5k and m = 10. The series includes 81 releases. Figure 5a demonstrates the numbers of counterfeits in those releases, in ascending order of the publication timestamps. Figure 5b presents the results of a similar experiment with respect to T_{sal}. For T_{occ} (T_{sal}), the maximum number of counterfeits at a timestamp is only 10 (12). Furthermore, at 51 (33) timestamps, no counterfeit is necessary at all.

Next, we focus on the average number of counterfeits per timestamp in a T_x^**-series*. Fixing m to 10, Figure 6a shows the averages for both T_{occ}^*- and T_{sal}^*-series as a function of r. The average decreases quickly as r increases, such that CG does not generate any counterfeit, for $r \geq$ 20k. This is expected, because critical absence is less likely when a larger number of tuples are inserted at each timestamp. In Figure 6b, we set r to 5k, and measure the average by varying m from 2 to 10. The average number never exceeds 2.5.

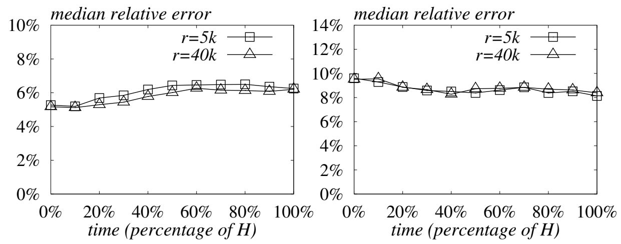
Utility of the Published Data. In the following set of experiments, we will use T_x^*-series (where x = occ or sal) to answer queries about the original microdata, and demonstrate the accuracy of query results. We concentrate on *aggregate queries*, since they are the basic operation for numerous mining tasks (e.g., decision tree learning, association rule mining, etc.). Specifically, each query has the form:

SELECT COUNT(*) FROM $T_x(j)$
WHERE $pred(A_1^{qi})$ AND ... AND $pred(A_4^{qi})$ AND $pred(A^s)$



(a) Vs. r (m = 10) (b) Vs. m (r = 5k)

Figure 6: Average number of counterfeits per timestamp



(a) T_{occ} (b) T_{sal}

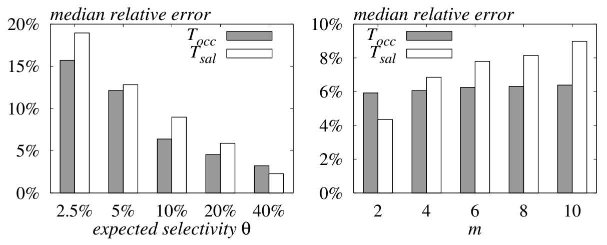
Figure 7: Query error vs. time (m = 10)

where j is a timestamp in the history of the deployed T_x^*-series, A_1^{qi}, ..., A_4^{qi} denote the four QI attributes in T_x, and A^s the sensitive attribute. For each attribute A, the condition $pred(A)$ has the form $A \in R$. Here, R is a random range in the domain of A, and has length $|A| \cdot \theta^{1/5}$, where $|A|$ is the domain size of A (see Table 5), and θ a query parameter called the *expected selectivity*. A larger result is returned with a higher θ. A workload consists of 10000 queries with the same j and s.

Given a query, we obtain its actual result act from the microdata table $T_x(j)$, and compute an estimated answer est from $T_x^*(j)$. The computation of est follows the algorithm in [23], except that here the table $R(j)$ auxiliary to $T_x^*(j)$ is also taken into account. Specifically, from each QI-group QI^* in $T_x^*(j)$, a *partial answer* is calculated as follows. We define a 4-dimensional rectangle z from QI^*, using $QI^*[A_1^{qi}]$, ..., $QI^*[A_4^{qi}]$ as the extents of z. Similarly, a 4-dimensional rectangle z' can be defined from the query's $pred(A_1^{qi})$, ..., $pred(A_4^{qi})$. We set c_1 to the ratio between the areas of $z' \cap z$ and z, c_2 to the percentage of the tuples in QI^* whose sensitive values qualify $pred(A^s)$, and c_3 to the number of counterfeits in QI^* (c_3 is available from $R(j)$). Then, the partial answer of QI^* equals $(|QI^*| - c_3) \cdot c_1 \cdot c_2$. Our estimate est equals the sum of the partial answers of all QI-groups.

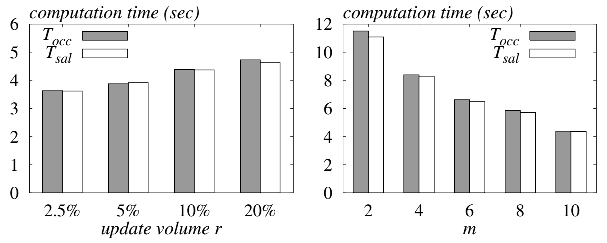
The relative error of a query equals $|act - est|/act$. We measure the *workload error* as the median relative error of all the queries. Adopting m = 10, Figure 7a (7b) plots the workload error as a function of time, for T_{occ}^*- (T_{sal}^*-) series with r = 5k and 40k (i.e., the two extreme values tested in Figure 6a), respectively. At all timestamps, the error is at most 10%, indicating high utility of the generalized tables. Furthermore, the error does not vary significantly with time, and is not sensitive to the update volume r.

In the experiments of Figure 8, we focus on T_{occ}^*- (T_{sal}^*-) series with r = 5k. We measure the average workload error of all workloads performed at each timestamp in the history of the employed series. Figure 8a plots the average error as a function of θ, for T_{occ}^*- and T_{sal}^*- series with m = 10. The accuracy improves as θ increases. This is expected because a higher θ leads to larger query results, whereas in general, aggregate analysis is effective for sizable queries. Fixing θ at 10%, Figure 8b illustrates the average error with respect to m. A smaller m requires less generalization, and hence, permits even more accurate analysis.



(a) Vs. θ (m = 10) (b) Vs. m (θ = 10%)

Figure 8: Average query error per timestamp (r = 5k)



(a) Vs. r (m = 10) (b) Vs. m (r = 5k)

Figure 9: Cost of counterfeited generalization

Computation Overhead. The last experiment evaluates the efficiency of our CG algorithm. First, we set m to 10, and measure the average time of computing a generalized relation in T_x^*-series (where x = occ or sal) of different r. Figure 9a demonstrates the time as a function of r. The cost is more expensive when r is higher, because the algorithm needs to process more newly inserted tuples at each timestamp.

Then, we fix r to 5k, and plot the cost as a function of m in Figure 9b. The overhead decreases as m increases, since a larger m necessitates fewer buckets, and requires a smaller number of bucket splits. In all cases, the algorithm terminates within 12 seconds.

6. RELATED WORK

The literature of centralized publication can be classified into three main categories. The first one aims at devising generalization principles, which serve as the criteria for judging whether a published relation provides sufficient privacy protection. k-anonymity [18, 20] and l-diversity [15] are the two most widely accepted principles, and hence, are used as the representatives in our discussion. Li and Li [14] propose t-*closeness*, which requires the distribution of sensitive values in each QI group to be analogous to the distribution of the entire dataset. Aggarwal et al. [2] suggest two principles based on clustering. Xiao and Tao [23] adopt a personalized approach, where each individual may request a tailored degree of privacy preservation.

The second category includes algorithms for computing a generalized table under a generalization principle, which minimizes some quality metric. With certain constraints [12] on the resulting generalization, it is often feasible to enumerate all the possible generalized relations. In that case, the optimal relation can be found efficiently by using several heuristics [5, 12] to prune the search space. Greedy solutions have also been proposed [7, 8, 11, 13] to obtain a suboptimal solution much faster. Several hardness results have been derived. In particular, it is shown [3, 13, 17] that computing the optimal generalized table is NP hard, even when a simple quality metric is deployed. Aggarwal [1] proves that, when the number of QI attributes is high, enforcing k-anonymity necessarily results in severe information loss, even for $k = 2$.

The third category concerns improving the utility of the published dataset, without compromising privacy protection. Kifer and Gehrke [9] advocate releasing the *marginals*, each of which anonymizes the projection of the microdata table on a subset of the (QI and sensitive) attributes. Xiao and Tao [22] present the *anatomy technique*, which publishes QI and sensitive values in two separate tables. In this way, QI values do not need to be generalized, since the separation already prevents privacy breaches. Koudas et al. [10] exploit a similar idea for improving the accuracy of aggregate search.

The above works focus on one-time publication. Wang and Fung [21] consider a re-publication scenario different from ours. They assume a (again, static) microdata table that contains a large number of QI attributes. In the first publication, the publisher releases a subset of those attributes, together with the sensitive attribute. Later, the publisher is requested to release a different subset of the QI attributes (but not the sensitive attribute). The objective is to anonymize the second publication, so that no adversary can infer sensitive data by combining both releases. Unlike [21], we aim at re-publication of microdata after its contents have been updated.

The work closest to ours is due to Byun et al. [6]. They propose an interesting technique that enables privacy-preserving re-publication of a dataset, after new tuples have been inserted. Despite its pioneering role in tackling re-publication, the technique has three shortcomings. First, it is inapplicable when deletions are allowed in the microdata, unlike the proposed solution that supports both insertions and deletions. Second, it enforces a simple version of l-diversity with weak privacy guarantees. Specifically, let S be the set of sensitive values that an adversary thinks may be the real sensitive value v of an individual. The technique of [6] ensures $|S| \geq l$, but imposes no limit on the adversary's confidence about v taking a specific element in S. For instance, if l = 2, an adversary may derive $S = \{HIV, flu\}$, but have exceedingly high (e.g., 99%) confidence about v = *HIV*. On the other hand, under the same assumption of adversaries' prior knowledge, m-invariance guarantees that such confidence is bounded by $1/m$, as proved in Lemma 3. Third, the technique of [6] requires consideration of all the releases in history, in assessing the risk of privacy disclosure of a new release. As a result, the space consumption (at the publisher) increases continuously with time, due to the need of retaining every past release. Furthermore, the computation cost of preparing a new release also grows monotonically, because more information must be examined each time. Our solution does not have this defect, since it demands storing only the last release, as established in Lemma 4.

7. CONCLUSIONS

The existing centralized-publication methods do not support re-publication of microdata in the presence of both insertions and deletions. This paper remedies the problem by developing m-invariance, a novel concept that prevents an adversary from using multiple releases to infer sensitive information. We present a formal analytical study that elaborates the theoretical foundation of m-invariance, and proves its effectiveness of limiting privacy disclosure. As a second step, we provide an efficient algorithm for computing anonymized versions of the microdata, which adequately protect privacy and yet support effective data analysis.

This work also initiates several promising directions for future work. First, it would be exciting to extend the proposed technique to tackle alternative forms of background knowledge. Research towards this direction may lead to the discovery of alternative generalization principles. Second, it may be worthwhile to study the possibility of releasing marginal tables [9], in order to further improve the utility of m-invariant publication. Another idea towards enhancing utility may be to combine m-invariance with *anatomy*

[22]. Third, it would be interesting to explore how m-invariant generalization can be adapted to optimize a given workload [11].

ACKNOWLEDGEMENTS

This work was supported by CERG Grant CUHK 1202/06 from the RGC of the HKSAR government. The authors would like to thank the anonymous reviewers for their insightful comments.

REFERENCES

[1] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB*, pages 901–909, 2005.

[2] G. Aggarwal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu. Achieving anonymity via clustering. In *PODS*, pages 153–162, 2006.

[3] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Anonymizing tables. In *ICDT*, pages 246–258, 2005.

[4] F. Bacchus, A. J. Grove, J. Y. Halpern, and D. Koller. From statistical knowledge bases to degrees of belief. *Artif. Intell.*, 87(1-2):75–143, 1996.

[5] R. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE*, pages 217–228, 2005.

[6] J.-W. Byun, Y. Sohn, E. Bertino, and N. Li. Secure anonymization for incremental datasets. In *SDM*, pages 48–63, 2006.

[7] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *ICDE*, pages 205–216, 2005.

[8] V. Iyengar. Transforming data to satisfy privacy constraints. In *SIGKDD*, pages 279–288, 2002.

[9] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *SIGMOD*, pages 217–228, 2006.

[10] N. Koudas, D. Srivastava, T. Yu, and Q. Zhang. Aggregate query answering on anonymized tables. In *ICDE*, 2007.

[11] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Workload-aware anonymization. In *SIGKDD*, 2006.

[12] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *SIGMOD*, pages 49–60, 2005.

[13] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *ICDE*, 2006.

[14] N. Li and T. Li. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, 2007.

[15] A. Machanavajjhala, J. Gehrke, and D. Kifer. l-diversity: Privacy beyond k-anonymity. In *ICDE*, 2006.

[16] D. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Halpern. Worst-case background knowledge in privacy. In *ICDE*, 2007.

[17] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *PODS*, pages 223–228, 2004.

[18] P. Samarati. Protecting respondents' identities in microdata release. *TKDE*, 13(6):1010–1027, 2001.

[19] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *PODS*, page 188, 1998.

[20] 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.

[21] K. Wang and B. C. M. Fung. Anonymizing sequential releases. In *SIGKDD*, pages 414–423, 2006.

[22] X. Xiao and Y. Tao. Anatomy: Simple and effective privacy preservation. In *VLDB*, pages 139–150, 2006.

[23] X. Xiao and Y. Tao. Personalized privacy preservation. In *SIGMOD*, pages 229–240, 2006.

APPENDIX

Proof of Lemmas 1 and 2. Let b be the row in $B(n)$ generated by t, and S_j $(x \le j \le y)$ be the set of tuples in $T^*(j)$ whose values along A_i^{qi} cover $b[A_i^{qi}]$, for each $i \in [1, d]$. Consider any reasonable surjection $f : U^*(n) \to B(n)$. For every $j \in [x, y]$, there exists exactly one tuple $t_j^* \in S_j$ satisfying $f(t_j^*) = b$ (Condition 1.1 of Definition 13). Furthermore, $t_x^*, t_{x+1}^*, ..., t_y^*$ have the same sensitive value (Condition 1.2), which is the sensitive value v of t reconstructed by f. By Definition 15, $v = t_j^*[A^s] \in t.CSS(j)$ $(x \le j \le y)$, which is equivalent to $v \in \cap_{k=x}^{y} t.CSS(k)$, and thus establishes Lemma 2.

$\cap_{k=x}^{y} t.CSS(k)$ definitely contains the real sensitive value $t[A^s]$ of t. Hence, if $\cap_{k=x}^{y} t.CSS(k)$ contains only one value, this value must equal $t[A^s]$. In this case, any reasonable surjection f reconstructs $t[A^s]$ correctly, rendering $n_{breach}(t) = n_{total}$. Hence, Lemma 1 is proved. ■

Proof of Lemma 3. Let t be an arbitrary tuple in $U(n)$, and b be the row in $B(n)$ generated by t. Given any reasonable surjective function $f : U^*(n) \to B(n)$, we define $AQ(b, f)$ as the set of QI-groups in $T^*(1)$, ..., $T^*(n)$ that contain at least a tuple in $f^{-1}(b)$.

Consider the n_{total} reasonable surjections from $U^*(n)$ to $B(n)$. We divide them into *batches* such that, for any surjections f and f' in the same batch, $AQ(b, f) = AQ(b, f')$. Let n_{bat} be the total number of resulting batches. For the i-th $(1 \le i \le n_{bat})$ batch F_i, use $cnt(F_i, v)$ to denote the number of surjections in F_i that reconstruct the sensitive value of t as v. In the sequel, we will show $cnt(F_i, t[A^s]) \le \frac{|F_i|}{m}$, which will establish the lemma because

$$risk(t) = \frac{\sum_{i=1}^{n_{bat}} cnt(F_i, t[A^s])}{n_{total}} \le \frac{\sum_{i=1}^{n_{bat}} |F_i|}{m \cdot n_{total}} = \frac{1}{m}.$$

Given any surjection $f \in F_i$, all QI-groups in $AQ(b, f)$ have an identical signature (Definition 16), due to m-invariance. Without loss of generality, assume that each QI-group in $AQ(b, f)$ has x sensitive values $v_1, v_2, ..., v_x$. The value of x is at least m as required by m-invariance.

Let f_1 be any surjection in F_i that reconstructs the sensitive value of t as v_1. Let us design another surjection $f_2 : U^*(n) \to B(n)$ as follows. Initially, $f_2(t^*)$ is undefined for any tuple $t^* \in U^*(n)$. Then, we inspect the QI-groups in each $T^*(j)$ $(1 \le j \le n)$. If a QI-group has two tuples t_1^* and t_2^* satisfying $t_1^*[A^s] = v_1$ and $t_2^*[A^s] = v_2$, we set $f_2(t_1^*) = f_1(t_2^*)$ and $f_2(t_2^*) = f_1(t_1^*)$. After examining all QI groups, for any tuple $t^* \in U^*(n)$ such that $f_2(t^*)$ remains undefined, we set $f_2(t^*) = f_1(t^*)$. Subjection f_2 is a reasonable surjection, which satisfies $AQ(b, f_1) = AQ(b, f_2)$. Hence, f_2 belongs to F_i.

f_2 reconstructs the sensitive value of t as v_2. The existence of f_2 for any f_1 implies $cnt(F_i, v_1) \le cnt(F_i, v_2)$. By symmetry, we can also derive $cnt(F_i, v_2) \le cnt(F_i, v_1)$, and thus, $cnt(F_i, v_2) = cnt(F_i, v_1)$. Extending the analysis to all the elements of $\{v_1, ..., v_x\}$, we have $cnt(F_i, v_1) = cnt(F_i, v_2) = ... = cnt(F_i, v_x) = |F_i|/x$. Given $x \ge m$ and $t[A^s] \in \{v_1, ..., v_x\}$, it holds that $cnt(F_i, t[A^s]) \le |F_i|/x \le |F_i|/m$. ■

Proof of Lemma 4. By Definition 17, if $\{T^*(1), ..., T^*(n)\}$ is m-invariant, then the two conditions in Lemma 4 hold. Conversely, assume that $T^*(n)$ satisfies both conditions in Lemma 4, we will show that $\{T^*(1), ..., T^*(n)\}$ is an m-invariant sequence.

Since all T_j^* $(1 \le j \le n)$ are m-unique, they satisfy the first requirement in Definition 17. Next we show that the second requirement holds for any tuple $t \in U(n)$. If $t \notin T(n)$, all the generalized hosting groups of t must have the same signature, because $\{T^*(1), ..., T^*(n-1)\}$ is m-invariant. If $t \in T(n)$ but $t \notin T(n-1)$, then t has only one generalized hosting group $t.QI^*(n)$, which appears in $T^*(n)$. In that case, the second requirement in Definition 17 trivially holds for t.

Now consider the case when $t \in T(n) \cap T(n-1)$. Since $\{T(1), ..., T(n-1)\}$ is m-invariant, the generalized hosting groups of t in $T^*(1)$, ..., $T^*(n-1)$ should share the same signature. By the second condition in Lemma 4, $t.QI^*(n-1)$ and $t.QI^*(n)$ also have an identical signature. Therefore, the second requirement in Definition 17 is also fulfilled for t. ■

Proof of Lemma 5. Let us consider any iteration I of the algorithm *Assign*. $\lambda, \alpha, \beta, \gamma, n_1, n_2, ..., n_\lambda$ are as defined in Section 4.2. To prove the lemma, it suffices to show that (again, for any I) we can always find a pair of (α, β) such that $\alpha \ge 1$, $\beta \in [m, \lambda]$, and they satisfy Inequalities 5-7. Let i' be the largest subscript i satisfying $n_i = n_1$ (specially, $i = 1$, if no element in $\{n_2, ..., n_\lambda\}$ is equivalent to n_1). We will prove that the three inequalities hold, given $\alpha = 1$ and $\beta = \max\{i', m\}$.

In the sequel, S refers to the content of S_- before I starts. Since S is m-eligible, $n_m \ge 1$. Hence, $n_\beta \ge n_{\max\{i', m\}} \ge n_m \ge 1 = \alpha$. Thus, Inequality 5 holds.

By the way β is defined, we have $n_{\beta+1} \le n_\beta - 1 = n_\beta - \alpha \le n_1 - \alpha$. Consequently, next we discuss only Inequality 6, since its satisfaction automatically validates Inequality 7. For this purpose, an important observation is $n_1 \le \gamma/\beta$. This observation can be established by discussing two cases. First, if $\beta = m$, by the m-eligibility of S, we know $n_1 \le \gamma/m = \gamma/\beta$. Second, if $\beta > m$, then $n_1 \cdot \beta = \sum_{i=1}^{i'} n_i \le \gamma$, also resulting in $n_1 \le \gamma/\beta$. Hence:

$$n_1 - \alpha \le \gamma/\beta - 1 \le (\gamma - \beta)/\beta \le (\gamma - \alpha \cdot \beta)/m.$$

Namely, Inequality 6 holds. ■