Exclusive Strategy for Generalization Algorithms in Micro-Data Disclosure

Lei Zhang[1], Lingyu Wang[2], Sushil Jajodia[1], and Alexander Brodsky[1]

[1] Center for Secure Information Systems
George Mason University
Fairfax, VA 22030, USA
{lzhang8,jajodia,brodsky}@gmu.edu
[2] Concordia Institute for Information Systems Engineering
Concordia University
Montreal, QC H3G 1M8, Canada
wang@ciise.concordia.ca

Abstract. When generalization algorithms are known to the public, an adversary can obtain a more precise estimation of the secret table than what can be deduced from the disclosed generalization result. Therefore, whether a generalization algorithm can satisfy a privacy property should be judged based on such an estimation. In this paper, we show that the computation of the estimation is inherently a recursive process that exhibits a high complexity when generalization algorithms take a straightforward *inclusive strategy*. To facilitate the design of more efficient generalization algorithms, we suggest an alternative *exclusive strategy*, which adopts a seemingly drastic approach to eliminate the need for recursion. Surprisingly, the data utility of the two strategies are actually not comparable and the exclusive strategy can provide better data utility in certain cases.

1 Introduction

The dissemination and sharing of information has become increasingly important to our society. However, such efforts may be hampered by the lack of security and privacy guarantees. For example, when a healthcare organization releases tables of diagnosis information, explicit identifiers such as names will be removed. However, an adversary may still identify a patient from the released table if, say, the combination of the patient's race, date of birth, and Zip code can be linked to a unique record in a publicly available voter list [25, 24, 20].

Existing solutions to the micro-data release problem are largely based on randomization or generalization. This paper considers generalization techniques. At an abstract level, a micro-data table can be considered as a mapping between quasi-identifiers (for example, the combination of race, date of birth, and Zip code) and sensitive values (such as diagnosis result). A generalization can be regarded as a partition on this mapping, which divides quasi-identifiers and corresponding sensitive values into disjoint groups. By hiding the detailed mapping

inside each group, each quasi-identifier is *blended* with others in the same group. The amount of privacy protection achieved through such a generalization can be measured under various privacy properties, such as l-diversity[2].

However, a major limitation of most existing solutions is to assume a disclosed table to be the only source of information available to an adversary. Unfortunately, this is not always the case. An adversary usually knows the fact that a generalization algorithm will maximize the utility function in addition to satisfying the privacy property (relying on the secrecy of such information is an example of *security by obscurity*). As recently pointed out in [30], this extra knowledge may allow the adversary to obtain a more precise estimation of the secret table, on which the privacy property may no longer be satisfied. An apparent solution is to anticipate what the adversary will do, that is to estimate the secret table based on both the disclosed table and public knowledge about the generalization algorithm. Once the estimation is obtained, the privacy property can be evaluated to decide the safety of the generalization algorithm.

In this paper, we study the computation of an adversary's estimation of the secret table, which can be modeled as a set of possible instances of the unknown secret table, namely, *disclosure set*. We show that a given sequence of generalization functions can be combined into different strategies in releasing generalized tables. We first consider generalization algorithms designed under a straightforward *inclusive strategy*. We show that the computation of disclosure sets under the inclusive strategy is inherently a recursive process and exhibits a high complexity. To facilitate the design of efficient generalization algorithms, we then suggest an alternative *exclusive strategy*, which adopts a seemingly more drastic approach to generalization in order to avoid the need for a recursive process. Surprisingly, we show that the data utility of those two strategies are actually incomparable, and the exclusive strategy can provide better data utility in certain cases. First of all, we motivate further discussions with an example.

Motivating Example Table 1 shows our running example as a table containing patient information. The table has three attributes: Name, Age, and Patient's Condition. The attribute Name is an identifier. We assume the Age attribute forms a quasi-identifier, and the Condition attribute is sensitive.

Table 2 shows three possible generalizations, G_1, G_2 and G_3, together with the original table, in an abstract way. We denote as ID the identifier (that is, Name), QI the quasi-identifier (that is, Age), and S the sensitive attribute (that is, Condition). Each generalization G_1, G_2 and G_3 includes a group quasi-identifier $QI_i (i = 1, 2, 3)$ and the sensitive attribute S (notice the identifier ID has been removed). For simplicity, we omit the details of each group quasi-identifier and sensitive value in the remainder of this paper.

Name	Age	Condition
$Alice$	21	flu
Bob	27	tracheitis
$Clark$	31	pneumonia
$Diana$	36	tracheitis
$Ellen$	43	gastritis
Fen	49	gastritis
$George$	52	cancer
$Henry$	58	enteritis
Ian	63	cancer
$Jason$	67	heart disease

Table 1. An Example of Patient Information Table

Original Table G_0			Generalization G_1		Generalization G_2		Generalization G_3	
ID	QI	S	QI_1	S	QI_2	S	QI_3	S
A	$g_0^1(21)$	c_1	g_1^1	c_1	g_2^1	c_1	g_3^1	c_1
B	$g_0^2(27)$	c_2	$(20 \sim 29)$	c_2	$(20 \sim 29)$	c_2	$(20 \sim 34)$	c_2
C	$g_0^3(31)$	c_3	g_1^2	c_3	g_2^2	c_3		c_3
D	$g_0^4(36)$	c_2	$(30 \sim 39)$	c_2	$(30 \sim 44)$	c_2	g_3^2	c_2
E	$g_0^5(43)$	c_4	g_1^3	c_4		c_4	$(35 \sim 54)$	c_4
F	$g_0^6(49)$	c_4	$(40 \sim 49)$	c_4	g_2^3	c_4		c_4
G	$g_0^7(52)$	c_6	g_1^4	c_6	$(45 \sim 59)$	c_6		c_6
H	$g_0^8(58)$	c_5	$(50 \sim 59)$	c_5		c_5	g_3^3	c_5
I	$g_0^9(63)$	c_6	g_1^5	c_6	g_2^4	c_6	$(55 \sim 69)$	c_6
J	$g_1^{10}(67)$	c_7	$(60 \sim 69)$	c_7	$(60 \sim 69)$	c_7		c_7

Table 2. An Example of Three Generalization Functions

We assume the generalization algorithm to be public knowledge. This knowledge has several aspects. First, the generalization algorithm defines a sequence of generalization functions sorted in a non-increasing order of data utility. In Table 2, the three generalizations are results of applying g_1, g_2, and g_3 to the original table G_0, respectively. We assume the three generalizations have non-increasing data utility (for example, average group size). The assumption of given aggregation functions is a common practice of most existing generalization techniques. Although the number of possible generalization functions may grow quickly, say, in the number of attributes, the issue of choosing suitable aggregation functions among all possibilities is beyond the scope of this paper. Our assumption of non-increasing utility in the sequence of functions is also a common practice, and also notice that functions with equal or incomparable utilities can be treated in the same way in our discussions. Second, the generalization algorithm defines a privacy property. In this paper, we consider a particular privacy property, namely, recursive $(2, 2)$-diversity (basically, among all possible sensitive values that any record can take, the highest ratio of any value X, denoted as $r_{DS}(X)$, should satisfy $r_{DS}(X) < 2(1 - r_{DS}(X))$, or equivalently, $r_{DS}(X) < 2/3$) [2]. Third, the generalization algorithm applies the sequence of

generalization functions to the original table, and returns the first generalization on which the privacy property evaluates to true. Clearly, this approach aims to maximize the data utility while satisfying the privacy property.

However, the above knowledge about the generalization algorithm may allow an adversary to deduce more information than what is directly disclosed in the generalization. For example, in Table 2, consider two cases. First, suppose an adversary does not know about the generalization algorithm, but only sees the second generalization G_2. In guessing the original table G_0, the adversary cannot discriminate the sensitive values in each group with respect to their association with each ID. For example, the ID A can be associated with either c_1 or c_2 in the group g_2^1. Therefore, to the adversary, all tables obtained by permuting the sensitive values within each group can potentially be the original table. Second, suppose an adversary knows about the generalization algorithm in addition to seeing G_2. The adversary can then deduce that G_1 must not satisfy the recursive $(2, 2)$-diversity because otherwise G_1 will be returned instead of G_2 due to better data utility. Although the adversary cannot see G_1 (more precisely, the sensitive values of G_1), based on the relationship between the groups in G_1 and G_2, he/she can still conclude that both E and F must be associated with c_4 in the original table. Clearly, between the above two cases, the recursive $(2, 2)$-diversity is satisfied in the first but not satisfied in the second.

The above example shows that it is insufficient to evaluate a privacy property based on a generalization itself when the generalization algorithm is publicly known. Unfortunately, this is indeed the approach adopted by most existing generalization algorithms. Those algorithms may thus produce results that actually violate the given privacy property (we say such algorithms are *unsafe*). To develop *safe* generalization algorithms, a critical question is: *What exactly can an adversary deduce about the original table, when he/she knows about the generalization algorithm?* In this paper, we first show how to exactly compute an adversary's knowledge about the original table, namely, the *disclosure set*. Second, as a consequent, we obtain a safe version of the traditional approach to generalization by evaluating the privacy property on the disclosure set instead of the generalization. Later in this paper, we shall show that by applying the safe version of the generalization algorithm to the above example, we would reach the counter-intuitive conclusion that neither G_2 nor G_3 can be safely disclosed.

Organization The remainder of the paper is organized as follows. Section 2 shows how to compute a disclosure set and reveals the inherent complexity of such a process. Section 3 introduces the *exclusive strategy* and studies the complexity and data utility of the corresponding generalization algorithms. Section 4 reviews related work. Section 5 finally concludes the paper.

2 Computing Disclosure Sets under the Inclusive Strategy

Section 2.1 first introduces the concept of *disclosure set*. Section 2.2 then studies the computation of disclosure sets under the *inclusive strategy*.

2.1 Disclosure Set

We consider the following micro-data disclosure problem. An *original table* $G_0(ID, QI, S)$ is given where ID, QI, and S denote the identifier attribute, quasi-identifier attribute(s), and sensitive attribute, respectively. A *generalization algorithm* $\mathbb{G}$ is given, which defines a sequence of *generalization functions* $g_1, g_2, \ldots, g_n$. The algorithm $\mathbb{G}$ applies each g_i in the given order to G_0 to obtain a *generalization* $G_i(QI_i, S)$ where QI_i is the group quasi-identifier attribute. We assume the last generalization function g_n always yields an empty set, indicating that nothing should be disclosed. The algorithm $\mathbb{G}$ always returns a generalization G_i that satisfies a given *privacy property* CHK.

The above discussion, however, does not address a critical issue, that is how the given privacy property CHK should be evaluated when a generalization G_i is to be disclosed. Generally, CHK should be evaluated based on an adversary's knowledge about the original table G_0. Such knowledge can be characterized as follows. The adversary attempts to guess G_0 based on the disclosed generalization G_i and the public information about the generalization algorithm $\mathbb{G}$. Any table that contradicts the information available to the adversary will be eliminated. The adversary will end up with a set of *possible instances*, which represents the best guess the adversary can make about G_0, namely, his/her knowledge about G_0. We call such a set the *disclosure set* corresponding to the generalization function g_i, denoted as DS_i. Clearly, the privacy property CHK should be evaluated on DS_i, when the generalization G_i is to be disclosed.

If a disclosed generalization is the only source of information available to the adversary, then the disclosure set DS_i is simply the collection of tables to which applying g_i will yield the generalization G_i. Such a collection of tables can be obtained by fixing an order on the attribute ID and QI while permuting the sensitive values within each group of G_i. We denote the collection of tables as $PER(G_i)$. For example, in Table 2, $PER(G_1)$ includes $2 \times 2 \times 1 \times 2 = 16$ tables since every group except g_1^3 has two permutations.

As illustrated in Section 1, we cannot simply evaluate CHK on $PER(G_i)$ when the generalization algorithm $\mathbb{G}$ is publicly known. The reason is that an adversary may eliminate some possible instances from $PER(G_i)$ due to a conflict with the fact that G_i is disclosed by $\mathbb{G}$. More precisely, the adversary can apply $\mathbb{G}$ to each possible instance in $PER(G_i)$. If $\mathbb{G}$ returns any $G_j (j < i)$, then the adversary knows this possible instance cannot be the original table G_0,

and hence it should not be included in DS_i. In this way, the adversary can derive DS_i as a subset of $PER(G_i)$.

Example 1. In Table 2, DS_1 is simply $PER(G_1)$ because generally the original table will not satisfy CHK, so the adversary cannot eliminate any instance from $PER(G_1)$. Although we shall delay the computation of DS_2, we can see that any possible instance in $PER(G_2)$ that has two different sensitive values associated with the ID E and F will cause $\mathbb{G}$ to return G_1 instead of G_2, and thus will not be included in DS_2.

We formalize the concept of disclosure set in Definition 1.

Definition 1. *The disclosure set DS_i corresponding to a generalization G_i is a set of possible instances that satisfy*

- $DS_i \subseteq PER(G_i)$
- $\forall X \in DS_i$ *the generalization algorithm* $\mathbb{G}$ *will not return* G_j *for any* $j < i$.

2.2 The Computation of Disclosure Set

Table 3 shows two algorithms. $\mathbb{G}$ on the left-hand side is a generalization algorithm and $\mathbb{DS}$ on the right-hand side is an algorithm for computing the disclosure set of a given generalization. $\mathbb{G}$ simply returns the first generalization G_i whose disclosure set (computed by the other algorithm $\mathbb{DS}$) satisfies a given privacy property CHK. On the other hand, $\mathbb{DS}$ computes the disclosure set of G_i by eliminating from $PER(G_i)$ any instance X for which the algorithm $\mathbb{G}$ returns a generalization that appears before G_i.

Algorithm $\mathbb{G}$
Input: An original table G_0,
generalization functions $g_1, g_2, \ldots, g_n$,
and a privacy property CHK
Output: A generalization $G_i (1 \leq i \leq n)$ or ϕ
Method:
1. **For** $i = 1$ to n
2. **If** $\mathbb{DS}(g_i(G_0))$ satisfies CHK
3. **Return** $g_i(G_0)$
4. **Return** ϕ

Algorithm $\mathbb{DS}$
Input: A generalization G_i
Output: The disclosure set DS_i
Method:
1. **Let** $DS_i = PER(G_i)$
2. **For** each $X \in DS_i$
3. **If** $\mathbb{G}(X) = G_j$ for some $j < i$
4. **Let** $DS_i = DS_i \setminus \{X\}$
5. **Return** DS_i

Table 3. Algorithms $\mathbb{G}$ and $\mathbb{DS}$

The algorithms in Table 3 show that the computation of disclosure sets is inherently a recursive process. In the algorithm $\mathbb{DS}$, to compute the disclosure

set of a generalization G_i, we must test every possible instance X in $PER(G_i)$ to determine whether X should be included in DS_i. More specifically, we first assume X to be the original table, then we apply the generalization algorithm $\mathbb{G}$. Each call to $\mathbb{G}$ will then involve $i-1$ calls to the algorithm $\mathbb{DS}$ for computing the disclosure set of the generalizations $g_j(X)(j = 1, 2, \ldots, i-1)$ (each such computation will again involve multiple calls to the generalization algorithm). One subtlety here is that we are actually using a modified version of $\mathbb{G}$ since it only uses the first $i-1$ generalization functions. This is in accordance with Definition 1, and it also guarantees the recursive process to always terminate.

Example 2. In Table 2, to compute DS_2, the algorithm $\mathbb{DS}$ will call the algorithm $\mathbb{G}$ with each of the possible instances as the input. In this simple case, only g_1 is applied to each instance, and DS_1 is simply equal to $EXP(G_1)$. Clearly, for any instance in which E and F are not both associated with c_4, the disclosure set DS_1 will satisfy the $(2, 2)$-diversity, and hence the instance is not included in DS_2. On the other hand, all instances in which both E and F are associated with c_4 (such as the original table G_0) form the disclosure set DS_2, which clearly does not satisfy $(2, 2)$-diversity, either.

The computation of disclosure sets has another complication as follows. Recall that to compute DS_i, we apply the algorithm $\mathbb{G}$ to each $X \in PER(G_i)$. The algorithm $\mathbb{G}$ will then compute a disclosure set for each generalization $g_j(X)(1 \leq j \leq i-1)$. It may seem that we can then reuse previous results since the disclosure sets $g_j(X)(1 \leq j \leq i-1)$ should normally have been computed before we compute DS_i (refer to the algorithm $\mathbb{G}$). However, this is not the case. The two sets $PER(G_{i-1})$ and $PER(G_i)$ are generally not comparable. Some instance X may appear in $PER(G_i)$ (for example, $g_2(X) = g_2(G_0)$) but not in $PER(G_{i-1})$ (for example, $g_1(X) \neq g_1(G_0)$). For such an instance X, the disclosure sets for $g_j(X)(1 \leq j \leq i-1)$ must be computed from scratch.

Figure 1 illustrates this situation. The left-hand side denotes the disclosure set DS_1, which is equal to $EXP(G_1)$. In the middle is DS_2 where the shaded oval represents $EXP(G_2)$. Each of the two small circles denotes a set $PER(X)$ that satisfies CHK for some $X \in EXP(G_2)$. All the instances in $EXP(G_2) \setminus PER(X)$ should thus be excluded from DS_2. Notice that while all instances in $PER(G_2)$ yield the same generalization under g_2, they may yield different results under g_1, as indicated by the two disjoint circles (there may certainly be more than two different results under g_1). One subtlety here is that when we compute DS_2 we typically assume DS_1 does not satisfy CHK, so none of the small circles could be DS_1.

Example 3. In Table 2, any instance $X \in PER(G_2)$ in which E and F are not both associated with c_4 will not appear in $PER(G_1)$. The disclosure set DS_1

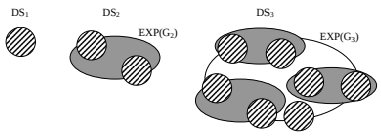

Fig. 1. Computing Disclosure Sets

must thus be re-computed for each such X while computing DS_2. On the other hand, for any such instance X, $PER(X)$ will satisfy the $(2, 2)$-diversity. If we represent those instances as small circles to be subtracted from $EXP(G_2)$, as in Figure 1, there would be $3 \times 3 - 1 = 8$ such circles (in $PER(G_2)$, E and F can each be associated with three different values so totally nine different generalizations are possible under g_1 among which only G_1 does not satisfy the $(2, 2)$-diversity).

The situation of computing DS_3 is similar but more complicated, as illustrated in the right-hand side of Figure 1. The ellipse depicts $PER(G_3)$. We first consider how the algorithm $\mathbb{DS}$ will compute DS_3. For each $X \in PER(G_3)$, the algorithm $\mathbb{G}$ may return g_1 if CHK is satisfied on the disclosure set of $g_1(X)$ (that is, $PER(g_1(X))$), as represented by the small circle. If CHK is not satisfied, the algorithm $\mathbb{G}$ will continue to compute the disclosure set for $g_2(X)$, which again involves computing a disclosure set for the generalization under g_1 on each instance in $PER(g_2(X))$. If CHK is satisfied on the disclosure set of $g_2(X)$, then the algorithm $\mathbb{G}$ returns g_2. When $\mathbb{G}$ returns either g_1 or g_2, the algorithm $\mathbb{DS}$ will exclude the instance X from DS_3. As illustrated in the right-hand side of Figure 1, an instance X in $PER(G_3)$ can satisfy one (and only one) of the following conditions.

1. CHK holds on the disclosure set of $g_1(X)$ (illustrated as small circles in Figure 1)
2. CHK holds on the disclosure set of $g_2(X)$ (illustrated as shaded areas)
3. CHK does not hold on the disclosure set of $g_1(X)$ or $g_2(X)$ (illustrated as unfilled areas)

Example 4. In addition to the original table G_0 in Table 2, Table 4 shows two other possible instances in $PER(G_3)$. The left-hand side table G_a is an example of instances that satisfy the first condition since $PER(g_1(G_1))$ clearly satisfies $(2, 2)$-diversity. Both the original table G_0 in Table 2 and the right-hand side table G_b in Table 4 are examples of instances that satisfy the third condition.

In Example 4, although both G_0 in Table 2 and G_b in Table 4 satisfy the third condition, they clearly do so in different ways. More specifically, CHK does not hold on $PER(g_1(G_b))$ or $PER(g_2(G_b))$; CHK does not hold on $PER(G_1)$ but it does hold on $PER(G_2)$. The reason that G_0 does not satisfy the second condition but the third is that CHK does not hold on DS_2. Referring to Figure 1, $PER(g_2(G_b))$ will be a shaded oval; $DS(G_2)$ will be a shaded oval (that is, $PER(G_2)$) subtracted by some small circles (that is, $PER(g_1(X))(X \in PER(G_2)$) on which CHK holds.

Table G_a

ID	QI	S
A	g_0^1	c_1
B	g_0^2	c_2
C	g_0^3	c_3
D	g_0^4	c_4
E	g_0^5	c_2
F	g_0^6	c_6
G	g_0^7	c_4
H	g_0^8	c_5
I	g_0^9	c_6
J	g_0^{10}	c_7

Table G_b

ID	QI	S
A	g_0^1	c_1
B	g_0^2	c_3
C	g_0^3	c_2
D	g_0^4	c_2
E	g_0^5	c_4
F	g_0^6	c_4
G	g_0^7	c_6
H	g_0^8	c_6
I	g_0^9	c_5
J	g_0^{10}	c_7

Table 4. Two Possible Instances in $PER(G_3)$

We are now ready to consider which instances in $PER(G_3)$ should be included in DS_3. Clearly, according to Definition 1, any instance that satisfies the first two conditions should be excluded, whereas instances satisfying the last condition should be included. Although the third condition can be satisfied in two different ways, we do not need to treat the two cases differently with the generalization algorithm $\mathbb{G}$ (however, we shall see the need for doing so in next section). In Figure 1, DS_3 corresponds to the unfilled area formed as the complement of all the small circles and shaded ovals.

Example 5. Both G_0 in Table 2 and G_b in Table 4 will be included in DS_3, although they fail $(2, 2)$-diversity in different ways (we shall see another case in next section).

In this special case, DS_3 can actually be computed more easily since there does not exist any $X \in PER(G_3)$ that can satisfy the above second condition (that is, $(2, 2)$-diversity is satisfied on the disclosure set of $g_2(X)$). Informally, any such X must first allow the $(2, 2)$-diversity to satisfy on $PER(g_2(X))$ but not on $PER(g_1(X))$ (for example, G_0 meets this requirement). However, we have that $g_1^1 = g_2^1$, $g_1^5 = g_2^4$, and g_2^2 and g_2^3 can satisfy $(2, 2)$-diversity only if they each includes three different values. Therefore, the only possibility is that

g_1^3 has two identical values, such as in the case with G_0. However, we already know that in this case the disclosure set of $g_2(X)$ will not satisfy $(2,2)$-diversity since both E and F must be associated with the same value. We conclude that the second condition cannot be satisfied by any instance in $PER(G_3)$, and DS_3 can thus be computed by excluding from $PER(G_3)$ any instance X with $(2,2)$-diversity satisfied on $PER(g_1(X))$.

In Figure 1, a confusion may arise about the instances in $PER(G_{i-1}) \setminus PER(G_i)$, such as those inside the small circles but outside the shaded ovals. When we compute the disclosure set for G_2, for any instance $X \in PER(G_2)$, we evaluate CHK on the disclosure set of $g_1(X)$. It seems those instances in $PER(g_1(X)) \setminus PER(G_2)$ should be excluded during such an evaluation because we know those instances are not possible. However, this is not the case. The algorithm $\mathbb{DS}$ simulates what an adversary will do to eliminate an instance X from $PER(G_2)$, he/she aims to prove that X cannot be the original table. For this purpose, the adversary will first assume that X is the original table and then attempt to show that CHK is already satisfied on $PER(g_1(X))$. If this is indeed the case, then $g_1(X)$ would have been released, and thus the adversary would not have any knowledge about $g_2(X)$ at all.

3 Exclusive Strategy

The generalization algorithm $\mathbb{G}$ in Table 3 adopts a straightforward strategy in using the sequence of generalization functions $g_1, g_2, \ldots, g_n$. That is, each function is applied in the given order, and the first generalization whose disclosure set satisfies the privacy property will be returned. Although this strategy is a natural choice and has been adopted by most existing generalization algorithms, it is not necessarily the only choice, neither is it an optimal choice in terms of data utility or computational complexity. By adopting different strategies, we may develop different generalization algorithms from the same sequence of generalization functions. In this paper, we do not intend to give a comprehensive study of possible strategies. Instead, we only present one strategy that is more efficient and may lead to more data utility in some cases.

Recall that in Example 4, G_0 in Table 2 and G_b in Table 4 are both included in DS_3. However, the difference lies in that CHK does not hold on $PER(g_2(G_b))$ but it does on $PER(G_2)$. An important observation is that we know G_b should be included in DS_3 without computing any disclosure sets, whereas we do not know whether G_2 should be included in DS_3 until we compute DS_2 (and know it does not satisfy CHK). Such a recursive computation of DS_2 within that of DS_3 brings high complexity, and should be avoided if possible. We thus propose a different strategy in handling instances like G_0. That is,

we simply do not include it in DS_3, regardless whether DS_2 satisfies CHK (notice that if DS_2 does satisfy CHK then G_2 will also be excluded from DS_3). If we were to represent this situation using Figure 1, then the shaded oval will correspond to any $PER(g_2(X))$ that satisfies CHK (and the small circles remain to have the same meaning), regardless whether the corresponding disclosure set satisfies CHK. More generally, we exclude any instance $X \in EXP(G_i)$ from DS_i, if only $EXP(G_j)$ satisfies CHK. We present this *exclusive* strategy as Algorithm $\mathbb{G}_e$ in Table 5. On the other hand, we shall refer to Algorithm $\mathbb{G}$ in section 2 as the *inclusive* strategy from now on.

Algorithm $\mathbb{G}_e$
Input: An original table G_0,
generalization functions $g_1, g_2, \ldots, g_n$,
and a privacy property CHK
Output: A generalization $G_i (1 \leq i \leq n)$ or ϕ
Method:
1. **For** $i = 1$ to n
2. **If** $PER(g_i(G_0))$ satisfies CHK
3. **If** $\mathbb{DS}(g_i(G_0))$ satisfies CHK
4. **Return** $g_i(G_0)$
5. **Else**
6. **Return** ϕ
7. **Return** ϕ

Algorithm $\mathbb{DS}_e$
Input: A generalization G_i;
Output: The disclosure set DS_i;
Method:
1. **Let** $DS_i = PER(G_i)$;
2. **For**each $X \in DS_i$
3. **For**$j = 1$ to $i - 1$
4. **If**$PER(g_j(X))$ satisfies CHK
5. **Let**$DS_i = DS_i \setminus \{X\}$
6. **Return** DS_i

Table 5. Algorithms $\mathbb{G}_e$ and $\mathbb{DS}_e$

In Table 5, it can be noticed that both the algorithm for generalization and that for computing disclosure sets in the exclusive strategy are different from those in the inclusive strategy. This fact is a reflection of the inter-dependency between the two algorithms, or equivalently, the inter-dependency between the approach to generalization and adversary's knowledge. More specifically, the generalization algorithm $\mathbb{G}_e$ simply refuse to disclose anything, if the given original table yield a generalization G_i for which $PER(G_i)$ satisfies CHK but DS_i does not. An adversary also knows this fact since the algorithms are publicly known. In guessing the original table after seeing G_i released, the adversary will test each instance $X \in PER(G_i)$ to see whether X can be the original table. However, different from the inclusive strategy, the exclusive strategy makes such a testing fairly simple. That is, any instance X for which $PER(g_j(X))$ satisfies CHK for some $j < i$ can be immediately eliminated from further consideration, because if X were indeed the original table, then the algorithm $\mathbb{G}_e$ would have either returned $g_j(X)$ (if its disclosure set satisfies CHK) or nothing (if the disclosure set does not satisfy CHK) instead of releasing G_i.

Example 6. Consider applying the exclusive strategy to G_0 in Table 2. Clearly, the three generalizations G_1, G_2, and G_3 do not change, because we are still using the same generalization functions as before (but in a different way). The disclosure sets DS_1 and DS_2 also remain the same (note that $PER(G_1) = DS_1$). When the algorithm $\mathbb{G}_e$ sees that $PER(G_1)$ does not satisfy CHK, it continues to the next generalization function g_2 as with the inclusive strategy. However, when $\mathbb{G}_e$ sees $PER(G_2)$ satisfies CHK but DS_2 does not, it simply returns ϕ indicating that nothing can be disclosed (recall that with the inclusive strategy, $\mathbb{G}$ will continue to g_3).

In contrast to the inclusive strategy, the exclusive strategy may seem to be a more drastic approach that may result in less data utility. Example 6 may seem to support this statement. However, this is in fact not the case. Due to space limitation, we cannot show DS_3 computed from Table 2 under the inclusive strategy, but we calculate the ratio of the association between E and c_4 in Example 7.

Example 7. As mentioned in Section 2.2, for this special case, DS_3 can be computed by excluding any instance X for which $EXP(g_1(X))$ satisfies CHK. The instances in DS_3 must thus fall into following three sets. First, both E and F have c_4. Second, both C and D have c_2, and only one of E and F may have c_4 (the other will have c_6). Third, both G and H have c_6, and only one of E and F may have c_4 (the other will have c_2). These three sets are clearly disjoint. Moreover, by counting the number of permutations, we can see that the cardinality of the first set is $6 \times 2 \times 6 = 72$ (A, B, and C can have 6 different permutations; D and G can have 2, etc.) among which all have E associated with c_4. Similarly, the second and third set each has $2 \times 6 = 12$ instances in which E is associated with c_4, and the other 12 instances in which E is associated with c_6 and c_2, respectively. We can thus conclude that the ratio of E associated with c_4 is $(72 + 12 + 12)/(72 + 24 + 24) = 0.8$.

By applying the inclusive strategy, the $(2, 2)$-diversity is not satisfied on DS_3. Therefore, nothing can be disclosed under the inclusive strategy, either. That is, for the given original table G_0 (and also the substantialized table in Table 1), the two strategies yield the same data utility. Besides, there also exist other cases where the exclusive strategy will provide more data utility. Suppose now G_b in Table 4 is given as the original table. Clearly, the inclusive strategy will disclose nothing because none of the generalizations through G_1, G_2 and G_3 can satisfy $(2, 2)$-diversity. For exclusive strategy, neither $PER(g_1(G_b))$ nor $PER(g_2(G_b))$ can satisfy $(2, 2)$-diversity. For $PER(g_3(G_b))$, again we calculate the ratio that c_4 is associated with E among all conditions in Example 8.

Example 8. Following Example 7, DS_3 under the exclusive strategy can be obtained by eliminating any instance X for which $PER(g_2(X))$ satisfy $(2,2)$-diversity from the previous result of DS_3 under the inclusive strategy. For the first set, D and G must now have c_2 and c_6, respectively, so we are left with 36 instances. Moreover, C and H must have 2 and 6, respectively, leaving totally 20 instances all with E associated with c_4. For the second and third set, nothing need to be eliminated. The ratio of E associated with c_4 is thus now $(20+12+12)/(20+24+24) = 0.647$. And this is also the maximal ratio of a single condition among all IDs.

Surprisingly, under the exclusive strategy, we can now disclose G_3 for the original table G_b in Table 4 (a substantialized example is shown in Table 6). In another word, the exclusive strategy actually provides more data utility in this case. The reason lies in the fact that the privacy property (that is, $(2,2)$-diversity) is not set-monotonic [30], neither is the sequence of sets of possible instances $PER(G_i)$ ($i = 1, 2, \ldots, n$). Generally, the data utility of the two strategies will be incomparable. Their performances depend on specific problem settings.

NAME	AGE	Condition
$Alice$	21	flu
Bob	27	pneumonia
$Clark$	31	tracheitis
$Diana$	36	tracheitis
$Ellen$	43	gastritis
Fen	49	gastritis
$George$	52	cancer
$Henry$	58	cancer
Ian	63	enteritis
$Jason$	67	heart disease

Table 6. Another Example of Patient Information Table

However, the exclusive strategy has an important advantage over the inclusive strategy, that is, a significantly lower complexity. In Table 5, unlike under the inclusive strategy, the algorithms under the exclusive strategy are not recursive because we do not call $\mathbb{G}_e$ within $\mathbb{DS}_e$. Denote x_i the complexity for computing the disclosure set DS_i under the inclusive strategy, and y_i the cardinality of $PER(G_i)$. We have that $x_i = (\sum_{j=1}^{i-1} x_j) \cdot y_i$ and $x_1 =\mid G_0 \mid$. By solving this recursive function, we can estimate the worst case complexity of the inclusive strategy to be $O((\mid PER(G_{max}) \mid)^n)$ where G_{max} is a generalization with the maximum cardinality of possible instances. In contrast, the complexity of the exclusive strategy is $O(n^2 \cdot \mid PER(G_{max}) \mid)$. By avoiding a recursive process, the exclusive strategy reduces the complexity from exponential to polynomial.

Other strategies are certainly possible, although their discussion is out of the scope of this paper. One complication is that the definition of disclosure sets given in Definition 1 should be generalized to accommodate the fact that the given sequence of generalization functions is not necessarily evaluated in the given order. The evaluation of those functions may actually happen in any order as defined in a strategy, and may vary depending on the given original table. For example, the exclusive strategy may directly jump to the last function (that returns ϕ) from any step. One way to keep the Definition 1 valid in this particular case is to have multiple copies of the last function and place a copy in front of each generalization function in the given sequence. In each step, if the algorithm chooses to either return the current generalization or to use the copy of the last function to return ϕ, then the current instance will be eliminated from the next disclosure set, which is in accordance with Definition 1.

4 Related Work

Micro-data disclosure has been extensively studied [1, 3, 10, 16, 17] where the security issue discussed in this paper is largely ignored. In particular, data swapping [9, 23, 28] and cell suppression [18] both aim to protect micro-data released in census tables. However, the amount of privacy is usually not measured in those earlier work. Miklau et. al presents an interesting measurement of information disclosed through tables based on the perfect secrecy notion by Shannon [8]. The important notion of k-anonymity is a model of privacy requirement [25] that received extensive studies in recent years. To achieve optimal k-anonymity (with the most utility) is shown to be computationally infeasible [21].

A model based on the intuition of blending individuals in a crowd is recently proposed in [27]. Personalized requirement for anonymity is studied in [29]. In [11], the authors approach the issue from a different perspective where the privacy property is based on generalization of the protected data and can be customized by users. Much efforts have been made around developing efficient k-anonymity algorithms [7, 24, 25, 20, 26, 15, 5], whereas the security of the k-anonymity model is assumed. Two exceptions are the l-diversity notion proposed in [2] and the t-closeness notion proposed in [19], which address the deficiency of k-anonymity of allowing insecure groups with a small number of sensitive values. Algorithms developed for k-anonymity can be extended to l-diversity and t-closeness, but they still do not take into account an adversary's knowledge about generalization algorithms. In [30], the authors pointed out the above problem and proposed a model for the adversary's knowledge, but did not give any efficient solution for the general micro-data disclosure problem.

In contrast to micro-data disclosure, aggregation queries are the main concern in statistical databases [22, 10, 13]. The main challenge is to answer aggregation queries without allowing an adversary to deduce secret individual values. The auditing methods in [6, 4] address this problem by determining whether each new query can be safely answered based on previously answered queries. The authors of [6, 12, 14] consider the same problem in more specific settings of off-line auditing and online auditing, respectively. Closest to our work, the authors of [14] consider knowledge about the decision algorithm itself. However, it only applies to a limited case of aggregation queries and does not consider the current state of the database in determining the safety of a query.

5 Conclusion

Armed with knowledge about a generalization algorithm used for computing disclosed data, an adversary may deduce more information to violate a desired privacy property. We have studied this issue in the context of generalization-based micro-data disclosure algorithms. We showed that a naive solution to address this issue demands prohibitive computational cost. We then introduced an alternative *exclusive strategy* for generalization algorithms. Compare to the naive exponential algorithms based on the traditional *inclusive strategy*, algorithms based on *exclusive strategy* have much better efficiency (polynomial in the size of the table), and also, provide even better data utility in certain cases.

Acknowledgements This material is partially supported by the National Science Foundation under grants CT-0716567, CT-0627493, IIS-0242237, and IIS-0430402; by the Army Research Office under the grant W911NF-07-1-0383; by the MITRE Technology Program; by the Natural Sciences and Engineering Research Council of Canada under Discovery Grant N01035; and by Fonds de recherche sur la nature et les technologies. The authors thank the anonymous reviewers for their valuable comments.

References

1. A.Dobra and S.E.Feinberg. Bounding entries in multi-way contingency tables given a set of marginal totals. In *Foundations of Statistical Inference: Proceedings of the Shoresh Conference 2000*. Springer Verlag, 2003.
2. A.Machanavajjhala, J.Gehrke, D.Kifer, and M.Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the 22nd IEEE International Conference on Data Engineering (ICDE 2006)*, 2006.
3. A.Slavkovic and S.E.Feinberg. Bounds for cell entries in two-way tables given conditional relative frequencies. *Privacy in Statistical Databases*, 2004.
4. D.P.Dobkin, A.K.Jones, and R.J.Lipton. Secure databases: Protection against user influence. *ACM TODS*, 4(1):76–96, 1979.

5. Y. Du, T. Xia, Y. Tao, D. Zhang, and F. Zhu. On multidimensional k-anonymity with local recoding generalization.
6. F.Chin. Security problems on inference control for sum, max, and min queries. *J.ACM*, 33(3):451–464, 1986.
7. G.Aggarwal, T.Feder, K.Kenthapadi, R.Motwani, R.Panigrahy, D.Thomas, and A.Zhu. k-anonymity: Algorithms and hardness. *Technical report, Stanford University*, 2004.
8. G.Miklau and D.Suciu. A formal analysis of information disclosure in data exchange. In *SIGMOD*, 2004.
9. G.T.Duncan and S.E.Feinberg. Obtaining information while preserving privacy: A markov perturbation method for tabular data. In *Joint Statistical Meetings*. Anaheim,CA, 1997.
10. I.P.Fellegi. On the question of statistical confidentiality. *Journal of the American Statistical Association*, 67(337):7–18, 1993.
11. J.Byun and E.Bertino. Micro-views, or on how to protect privacy while enhancing data usability: concepts and challenges. *SIGMOD Record*, 35(1):9–13, 2006.
12. J.Kleinberg, C.Papadimitriou, and P.Raghavan. Auditing boolean attributes. In *PODS*, 2000.
13. J.Schlorer. Identification and retrieval of personal records from a statistical bank. In *Methods Info. Med.*, 1975.
14. K.Kenthapadi, N.Mishra, and K.Nissim. Simulatable auditing. In *PODS*, 2005.
15. K.LeFevre, D.DeWitt, and R.Ramakrishnan. Incognito: Efficient fulldomain k-anonymity. In *SIGMOD*, 2005.
16. L.H.Cox. Solving confidentiality protection problems in tabulations using network optimization: A network model for cell suppression in the u.s. economic censuses. In *Proceedings of the Internatinal Seminar on Statistical Confidentiality*, 1982.
17. L.H.Cox. New results in disclosure avoidance for tabulations. In *International Statistical Institute Proceedings*, 1987.
18. L.H.Cox. Suppression, methodology and statistical disclosure control. *J. of the American Statistical Association*, 1995.
19. N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, 2007.
20. L.Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.
21. A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *ACM PODS*, 2004.
22. N.R.Adam and J.C.Wortmann. Security-control methods for statistical databases: A comparative study. *ACM Comput. Surv.*, 21(4):515–556, 1989.
23. P.Diaconis and B.Sturmfels. Algebraic algorithms for sampling from conditional distributions. *Annals of Statistics*, 1998.
24. P.Samarati. Protecting respondents' identities in microdata release. In *IEEE TKDE*, pages 1010–1027, 2001.
25. P.Samarati and L.Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. *Technical report, CMU, SRI*, 1998.
26. R.J.Bayardo and R.Agrawal. Data privacy through optimal k-anonymization. In *ICDE*, 2005.
27. S.Chawla, C.Dwork, F.McSherry, A.Smith, and H.Wee. Toward privacy in public databases. In *Theory of Cryptography Conference*, 2005.
28. T.Dalenius and S.Reiss. Data swapping: A technique for disclosure control. *Journal of Statistical Planning and Inference*, 6:73–85, 1982.
29. X.Xiao and Y.Tao. Personalized privacy preservation. In *SIGMOD*, 2006.
30. L. Zhang, S. Jajodia, and A. Brodsky. Information disclosure under realistic assumptions: Privacy versus optimality. In *ACM Conference on Computer and Communications Security (CCS) 2007*.