RESEARCH ARTICLE

An effective value swapping method for privacy preserving data publishing

A S M Touhidul Hasan1,2, Qingshan Jiang1*, Jun Luo1, Chengming Li3 and Lifei Chen3
1 Shenzhen Key Laboratory of High Performance Data Mining, Shenzhen Institute of Advanced Technology, China, 518055
2 University of Chinese Academy of Sciences, Beijing 100049, China
3 School of Mathematics and Computer Science, Fujian Normal University, Fuzhou 350117, China

ABSTRACT

Privacy is an important concern in the society, and it has been a fundamental issue when to analyze and publish data involving human individual’s sensitive information. Recently, the slicing method has been popularly used for privacy preservation in data publishing, because of its potential for preserving more data utility than others such as the generalization and bucketization approaches. However, in this paper, we show that the slicing method has disclosure risks for some absolute facts, which would help the adversary to find invalid records in the sliced microdata table, resulting in breach of individual privacy. To increase the privacy of published data in the sliced tables, a new method called value swapping is proposed in this work, aimed at decreasing the attribute disclosure risk for the absolute facts and ensuring the l-diverse slicing. By value swapping, the published table contains no invalid information such that the adversary cannot breach the individual privacy. Experimental results also show that the NEW method is able to keep more data utility than the existing slicing methods in a published microdata table. Copyright © 2016 John Wiley & Sons, Ltd.

KEYWORDS
privacy preservation; data security; data anonymization; data publishing

*Correspondence
Qingshan Jiang, Shenzhen Key Laboratory of High Performance Data Mining, Shenzhen Institute of Advanced Technology, China, 518055.
E-mail: qs.jiang@siat.ac.cn

1. INTRODUCTION

At the age of explosive growth of the Internet have increased the dependence of sharing information, in both organization and individuals universally. This trend of data publishing has led an overgrowing demand for keeping the privacy of published data. Privacy preservation has become a primary concern in many data mining applications [1]. Table I is an example of the microdata table, which would be published for data mining or any other research purpose.

In the microdata Table I, the attributes of the table are partitioned into three categories. Some attributes are the identifiers, which can identify a person, such as the social security number or name. In general, the identifier has been removed from the microdata table during the data publishing. Some attributes are quasi-identifiers (QIs) such as birth date, sex, and zip code. An adversary may know these three QI attributes from other public data sources such as voter registration and driver registration list, and when taken these QI attributes together can identify a person. In the USA, 87% of individuals can be uniquely identified using only these three QIs [1]. Some attributes are called sensitive attributes (SAs), which are unknown to the adversary and are considered as sensitive such as disease and salary.

If a microdata table is published to the other parties for data mining, privacy preserving techniques are often mandatory to reduce the possibilities of identifying the sensitive information of an individual person [2]. For data publishing, fundamental and significant concerns are modification techniques, keeping the privacy of individual and modification percentage of the data. These issues are suggested some data anonymization techniques [1,3–5], to anonymize microdata for the data publishing. To limit disclosure risks, several anonymization techniques have been proposed recently in [1,3,4,6,7].

In this paper, we are interested in slicing [1] as it is a popular method for data anonymization with keeping more
anonymity if the size of the QI group is at least \( k \). The above example shows that slicing cannot make sure \( l \)-diverse slicing for some special set of sensitive values. In this paper, an efficient method called value swapping is proposed to improve the slicing mechanism for the special set of sensitive values. On the other hand, background knowledge helps the adversary to find the invalid records, and these invalid records support to breach the individual privacy. To identify such invalid records, we use negative association rules [5], in the sliced microdata table. The contributions to solving the slicing problems are summarized as follows.

We propose value swapping method to swap the invalid records, which also prevents the published microdata table from being created invalid records. With the negative association rules, value swapping method swaps the invalid records in the bucket. In this way, the published microdata table satisfies \( l \)-diverse slicing, and the adversary cannot breach the individual privacy. A series of experiments on synthesis and real-world data are conducted on the sliced table to support the effectiveness of the value swapping method.

The remainder of this paper is organized as follows. Section 2 presents background and related work. Section 3 describes the value swapping method. Section 4 discusses the experimental analysis. Section 5 gives the conclusion.

### 2. BACKGROUND AND RELATED WORK

In this section, we present existing anonymization techniques and discuss the data utility and problems of slicing approach for the published microdata table.

#### 2.1. Anonymization techniques

There are extensively work performed to address the information disclosure for statistical databases. Some popular anonymization techniques [1,3–5,8–13] have been proposed to protect the published microdata from information disclosure risks. In [3,9–11], the authors proposed \( k \)-anonymity approach where a QI group is said to satisfy \( k \)-anonymity if the size of the QI group is at least \( k \). The \( k \)-anonymity property ensures the identity disclosure, but it does not provide sufficient protection against attribute and
membership disclosure [5]; also, $k$-anonymity reduces the utility of published microdata table.

To address the drawbacks of $k$-anonymity, $l$-diversity is proposed [4,14]. A QI group is $l$-diverse if the probability that any tuple in this group is associated with a sensitive value is at most $1/l$. Both $k$-anonymity and $l$-diversity use generalization techniques to anonymize the microdata. For the generalization, microdata table loses an enormous amount of information, particularly for higher dimensional data [1]. Generalization breaks the correlation between attributes, and it assumes that any possible combinations of attribute values are equally possible.

To increase the data utility and the privacy, recently, the slicing method [1] and its variants [15–19] have been proposed for anonymizing the microdata. There is another slicing [13] technique that is related to sensor data publishing but not being related to relational data publishing. In [20], a location privacy preservation method was proposed for cognitive radio networks to protect the location privacy with maximizing the data utility, while [21] suggests a trade-off between individual privacy and system performance for cyber-physical systems.

For anonymizing the data, slicing partitions the data set both vertically and horizontally. Vertically partitioning methods group the highly correlated attributes, while horizontal methods group tuples into buckets, and the values are randomly permuted to break the linking between different columns. Slicing permuted attribute values randomly in each bucket, and it has the probability to create some invalid records. Invalid records will decrease the utility of the published microdata table. By analyzing invalid records in the published table, an adversary will understand the anonymization mechanism, and this knowledge will help to breach the published data privacy. The injector method is proposed in [5], to make use of negative association rules to find the invalid records in sliced microdata table. Invalid records would help the adversary to breach the privacy of published microdata.

Adversary’s background knowledge assists in learning relevant sensitive information from published microdata table. A general framework in [12] was presented to formalize the background knowledge. The background knowledge contributes to find invalid records, and it breaches individual privacy in the sliced [1,15–19] microdata table.

### 2.2. Data utility and information loss

The fundamental issue for privacy preserving data publishing is anonymization of the published data, and also, we have to consider the data utility. This problem is interesting because there is a pair of contradictory goals for data publishing, and these are data utility and privacy. If we want to increase the utility of published data, we have to provide some useful information, which may result in privacy breach. Data utility tells that how much information retain in the published microdata table, and those data are practically useful. During the generalization process of microdata table, it loses valuable information. Table III is anonymized by generalization, and it loses some specific values. A large amount of information are lost during the generalization process, and it reduces the data utility. Therefore, the researcher cannot be able to draw valuable information pattern from the published data.

There are many methods [2] to calculate the information loss in the published data. In this paper, we apply a simple metric to calculate the data utility. Invalid records reduce the data utility, and information loss happened in the published microdata table. The experimental work shows that slicing creates invalid records in the published table and reduces the data utility.

#### 2.3. Problems of slicing

In slicing techniques, authors showed how to prevent disclosure risks without using generalization mechanism. Hence, slicing keeps more data utilities and privacy in the published microdata table. Slicing is a right approach for anonymizing high dimensional data, and it keeps more data utility than $k$-anonymity and $l$-diversity because it does not generalize the attribute values. Slicing randomly permuted the values in the bucket to break the column correlation, which ensures the privacy of individuals in the published data such as attribute disclosure and membership disclosure risks. Although slicing does not generalize the values so that the data miners can obtain useful information and patterns from the published microdata table. To keep protection against membership disclosure, slicing creates fake tuples, and it does not have any effect on the data utility.

A record is said to be invalid if the sensitive values $SA$ is not compatible with the QI values. In slicing, authors have introduced $l$-diverse slicing. However, for some absolute facts, it could not satisfy $l$-diverse slicing. For example, in Table II, tuple $t_1$ has only one matching bucket, and it is linked with two sensitive values for Zipcode 47906. With $l$-diverse slicing, any person would be linked with sensitive values with a probability not greater than $1/l$. It is shown that slicing satisfies $l$-diverse slicing because it is linked with the sensitive values by $1/2$, which satisfies $l$-diverse slicing. However, if background knowledge is used for the special set of sensitive values, then tuple $t_1$ is linked with only one sensitive value because it is identified tuple $t_1$ is

### Table III. Generalized table.

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>Zipcode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>21–25</td>
<td>M</td>
<td>479*</td>
<td>Gastritis</td>
</tr>
<tr>
<td>21–25</td>
<td>F</td>
<td>479*</td>
<td>Ovarian cancer</td>
</tr>
<tr>
<td>31–35</td>
<td>F</td>
<td>479*</td>
<td>Breast cancer</td>
</tr>
<tr>
<td>51–55</td>
<td>M</td>
<td>479*</td>
<td>Flu</td>
</tr>
<tr>
<td>51–55</td>
<td>M</td>
<td>479*</td>
<td>Dyspepsia</td>
</tr>
<tr>
<td>56–60</td>
<td>F</td>
<td>479*</td>
<td>Ovarian cancer</td>
</tr>
<tr>
<td>56–60</td>
<td>F</td>
<td>473*</td>
<td>Breast cancer</td>
</tr>
<tr>
<td>61–65</td>
<td>F</td>
<td>473*</td>
<td>Fever</td>
</tr>
</tbody>
</table>
a male person, and a male person cannot suffer from ovarian cancer. For the special set of sensitive values, slicing techniques can not satisfy l-diverse slicing, and it leads to disclose the personal privacy of the published microdata table. Moreover, invalid records reduce the data utility, and it has the higher relative query error.

To eliminate these problems, value swapping mechanism is used in each bucket, and it will help to protect the published microdata table to create such invalid records so that the published microdata table satisfies l-diverse slicing.

### 3. METHODOLOGY

In this section, we present the value swapping algorithm for the sliced microdata table. Also, we provide the discussion on the value swapping method.

#### 3.1. Value swapping

In this paper, the value swapping method is presented to improve slicing approach based on negative association rules. Randomly permuted values in a bucket do not always guarantee that no membership or attribute disclosure will happen. It has a major drawback for some special set of sensitive values at SA columns. Moreover, permuting those values will not increase the privacy rather increase the risks of attribute disclosure. With negative association rules, some SA values are not compatible with QI values. If negative association rules are used such as $\text{man} \rightarrow \hat{O}$ ($O$ indicates ovarian cancer) will help to disclose the attribute disclosure.

Negative association rules and background knowledge have used to propose the value swapping method for improving the slicing technique. Value swapping method ensures the protection of creating invalid records as well as protection of attribute disclosure in the published microdata table. Table IV is a sliced table with value swapping, and it satisfies l-diverse slicing, and it does not contain any invalid records. Value swapping method will increase the data utility as well as individual privacy in the published microdata table.

<table>
<thead>
<tr>
<th>Age, sex</th>
<th>Zipcode, disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>22, M</td>
<td>47901, flu</td>
</tr>
<tr>
<td>22, F</td>
<td>47906, gastritis</td>
</tr>
<tr>
<td>33, F</td>
<td>47901, breast cancer</td>
</tr>
<tr>
<td>52, M</td>
<td>47901, fever</td>
</tr>
<tr>
<td>54, M</td>
<td>47901, gastritis</td>
</tr>
<tr>
<td>60, F</td>
<td>47906, ovarian cancer</td>
</tr>
<tr>
<td>60, F</td>
<td>47906, dyspepsia</td>
</tr>
<tr>
<td>64, F</td>
<td>47906, breast cancer</td>
</tr>
</tbody>
</table>

Table IV. Sliced table with value swapping.

The objective of this research is to find the negative association rules in every bucket $B$. It can be said that a tuple $t$ cannot take a sensitive value if there exists a negative association rule. A negative association rule is denoted as $X \rightarrow \hat{Z}$, where $X$ is a predicate involving related to QI, and $Z$ is the SAs.

Let a bucket $B$ containing $N$ tuples and $N$ sensitive values. Formally, given a tuple $t$ and a sensitive value $s$ in the bucket $B$, sensitive value $s$ is said to be valid for tuple $t$ if there exists a possible assignment between tuples and sensitive values such as the link between $t$ and $s$. Consider two tuples $t$ and $i$ where $t$ belong to $s$ and $i$ belong to $\hat{s}$. Tuple $t$ is said to be incompatible with tuple $i$ if at least one of the three conditions satisfy: (i) $s = \hat{s}$; (ii) $t$ cannot take value $\hat{s}$; and (iii) $t$ cannot take value $s$.

Let $T$ be a microdata table that has been published by slicing [1]. Table $T$ contains $b$ buckets $B_i (1 \leq i \leq b)$. Every bucket $B_i$ contains $n$ tuples $t_j (1 \leq j \leq n)$ where every tuple $t$ contains $q$ QI attributes: $\{a_1, a_2, \ldots, a_q\}$ and an SAs.

An incompatible tuple set $Y$ is the set where tuple’s QIs $\{a_1, a_2, \ldots, a_q\}$ are not compatible with the sensitive value $s$. A compatible tuple set $C$ is the set where tuple’s QIs $\{a_1, a_2, \ldots, a_q\}$ are compatible with the sensitive value $s$.

The value swapping method checks the compatibility of every tuple in the bucket $B$ and creates the incompatible and compatible tuple sets $Y$ and $C$, respectively. Incompatible tuple’s sensitive value $s$ is swapped with the compatible tuple’s sensitive value $\hat{s}$. After swapping checks the compatibility of the new tuple and if the new tuple satisfies the compatibility, it will append to the compatible tuple set $C$. This process is running in each bucket $B$ until it satisfies $Y \cap C = \emptyset$.

#### 3.2. Value swapping algorithms

In this subsection, we present three algorithms to perform value swapping for sliced microdata table. The value swapping algorithm is presented based on Injector [5], and the algorithm is modified for “incompatible tuple check” and “incompatible tuple value swapping”. Given a microdata table $T$ and a parameter $l$, the main objective is to obtain the anonymized table $T^*$, which would not have any invalid records and will satisfy $l$-diverse slicing. In Algorithm 1, it has four steps to obtain the anonymized table $T^*$. The first step is to do the anonymization on microdata table $T$ as in slicing [1] and obtain the table $T^3$.

To swap the invalid sensitive values, Algorithm 2 and Algorithm 3 are used. Algorithm 2 finds the invalid tuples in the sliced microdata, and Algorithm 3 swaps the invalid sensitive values to make sure no invalid tuples exist in...
Algorithm 1 Value Swapping
Input: Microdata Table T
Output: Anonymized Table T*
1: For a given table T generates an anonymized table T*, privacy requirement R of l-diversity.
2: Step 1: Anonymize the Table T as in slicing [1] and return Table T*.
3: Step 2: Check the incompatible tuple in each bucket B_i of Table T* by Algorithm 2: Incompatible Tuple Check.
4: Step 3: Swap the incompatible tuple values with the appropriate sensitive attributes by the Algorithm 3: Incompatible Tuple Value Swapping and return anonymized table Tsw.
5: Step 4: Take anonymized table Tsw for diversity check as in slicing [1] and publish the final anonymized table T*

Algorithm 2 Incompatible tuple check
Input: Bucket B_i
Output: Incompatible tuple set Y in Bucket B_i Incompatible (B_i)
1: each tuple t_i =< q_i, s_i > in B_i
2: Set Y = Ø
3: C = count (number of rows in B)
4: for i <= C do
5: Check the tuple t_i’s incompatibility
6: if q_i is not compatible with s_i then
7: Y = Y \{ t_i \}
8: end if
9: end for
10: if |Y| = 0 then
11: return F
12: else
13: return Y
14: end if

the published microdata table. Finally, at step four, the diversity of the published table is checked as in slicing [1].

3.2.1. Algorithm: incompatible tuple check.
Algorithm 2 checks the tuple incompatibility in each bucket B_i. In slicing, every bucket has t_i tuples, and each tuple consist of QI attributes q_i and sensitive attribute s_i, such that t_i =< q_i, s_i >. QI q_i is checked with the SA s_i. If the quasi-identifier is not compatible with the SA, then the tuple is added with the set Y. Tuple incompatibility is checked based on absolute facts. Incompatible tuple check phase finds the incompatible tuples in each bucket B and maintains the data structure Y to keep the incompatible tuples, and initially Y is empty. In each iteration (line 4 to 9), Algorithm 2 checks the compatibility of QI attributes with the SA (line 5) and update Y if it is an incompatible tuple (line 7).

3.2.2. Algorithm: incompatible tuple value swapping.
In Algorithm 3, the return value of Algorithm 2 is used for the value swapping. By Algorithm 3, incompatible tuple’s sensitive value is swapped. Table T* is used for swapping operation in each bucket B_i, based on the incompatible tuple set Y. Algorithm 2 maintains two data structures: (i) a list of bucket B_i and (ii) the value swapping table Tsw. Initially, value swapping table Tsw is empty, and bucket B_i starts from the first bucket of sliced table T*.
In each iteration (line 1 to 12), incompatibility of tuples are checked (line 2). If it returns true, the algorithm starts to swap incompatible tuples (line 5). Finally, the value swapping table Tsw (line 11) is published.

Algorithm 3 Incompatible tuple value swapping
Input: Sliced table T*
Output: Value swapping table Tsw
1: for each bucket B_i \in T^S (T* is sliced table) do
2: if algorithm Incompatible(B_i) then
3: Set C = B_i \ Y (C is the available tuples for value swapping)
4: for each tuple t in Y do
5: take the tuple t from Y and swap the sensitive values s with the tuple t from C, discard the tuple from Y and append to C.
6: Y = Y \{ t \}
7: C = C \{ t \}
8: end for
9: B_i = C
10: end if
11: Tsw = Tsw \cup \{B_i\}
12: end for
13: return Tsw

3.3. Discussion on the value swapping method
The objective of the value swapping method is to protect the sliced table from creating invalid records and maintaining the l-diverse slicing. Table V is a published microdata table anonymized by slicing. Algorithm 1 describes the steps for value swapping. In Table V applying Algorithm 2 finds the incompatible tuple set $Y = \{t_1, t_4, t_5\}$ in the bucket $B_1$. By using Algorithm 3, incompatible tuple’s sensitive value is swapped to make sure no invalid records exist in the bucket $B_1$ and the incompatible tuple set $Y = \emptyset$.
Table IV is the published microdata table by value swapping, and it increases the data utility of the published microdata because it contains no invalid records. It is not sure Table IV satisfies l-diverse slicing or not. It has been discussed earlier that the sliced microdata table could not satisfy l-diverse slicing for the special set of sensitive...
values. The following example will show how the value swapping method satisfies $l$-diverse slicing.

Let a tuple $t_1$ with QI values $(22, M, 47906)$ and searching for the QI values in both Tables IV and V. To determine $t'_1$ sensitive values $t'_1$, matching bucket is examined. By analyzing the first column (Age, sex) in both tables, $t_1$ exist in both buckets. However, by examining the Zipcode attribute of the second column (Zipcode, disease) in both buckets, Zipcode 47906 exist only in the first bucket in both Tables IV and V. For tuple $t_1$ sensitive values from Table IV, $(47906, gastritis)$, $(47906, ovarian cancer)$, $(47906, dyspepsia)$, and $(47906, breast cancer)$, and from Table V, $(47906, gastritis)$, $(47906, ovarian cancer)$, $(47906, breast cancer)$, and $(47906, gastritis)$ are found. Based on negative association rules, a male cannot suffer from ovarian and breast cancer. Because tuple $t_1$ is a male according to the QI values.

After excluding the invalid records in Table IV, sensitive values exist $(47906, gastritis)$ and $(47906, dyspepsia)$, and in Table V, sensitive values exist $(47906, gastritis)$ and $(47906, gastritis)$. Now, for Table IV, tuple $t_1$ is linked with two sensitive values, and Table IV satisfies $l$-diverse slicing, but for Table V, the tuple $t_1$ is linked with one sensitive value because both values have the same disease, which is gastritis, and it cannot satisfy $l$-diverse slicing.

It can be said that sliced microdata table increases the utility of the published data after value swapping because it does not contain any invalid records, and it satisfies $l$-diverse slicing. Slicing with value swapping is a good choice for multiple SAs because slicing with value swapping technique breaks the columns correlation by permutating the attribute values and restricts the table to create such invalid records.

### 4. EXPERIMENTAL ANALYSIS

In this section, we conduct the experiments on synthetic and real-world data sets. Experiments are divided into two parts: the first part is designed for examining the error rate in the sliced table and the second part is to measure the data utility on the published anonymized table. Algorithm 1 is used to do the anonymization process, and the value swapping method makes sure after value swapping the sliced microdata table satisfies $l$-diverse slicing.

#### 4.1. Data sets

Synthetic and adult data sets [22] are used to see how many invalid records create in the slicing process [1]. In the synthetic data set, there are 10,000 tuples, the negative association rules [5] mined records are increased from 5% to 30%, which is used to see the error rates in sliced microdata table. There are total 45,222 valid tuples in adult data set [22]. In the synthetic data set, there are four attributes, among them three (age, sex, zipcode) are used as QI values and one (disease) is used as sensitive value. In the adult data set, eight attributes are kept. Data sets are described in Tables VI and VII, respectively.

In the adult data set, error rates are shown for the slicing process [1]. By analyzing the adult data sets, negative association rules are mined and recorded in Table VIII. With the negative association rules, invalid records are found in the adult data sets after slicing. The experiment shows that slicing on adult data set has created invalid records, which are recorded as the invalid record percentage in Table IX. The value swapping method improves the final result and increases the utility of the published microdata table because after value swapping published microdata table contains no invalid records.

#### 4.2. Error rate

The goal of error rate experiment is to find the invalid records in the sliced microdata table. The slicing process creates invalid records during the anonymization. In the experiment, we check every tuple's QI values with the

<table>
<thead>
<tr>
<th>Table V. Sliced table with fake data.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age, sex</td>
</tr>
<tr>
<td>22, M</td>
</tr>
<tr>
<td>22, F</td>
</tr>
<tr>
<td>33, F</td>
</tr>
<tr>
<td>52, M</td>
</tr>
<tr>
<td>64, M</td>
</tr>
<tr>
<td>60, F</td>
</tr>
<tr>
<td>60, F</td>
</tr>
<tr>
<td>64, F</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table VI. Description of adult data set.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>1 Age</td>
</tr>
<tr>
<td>2 Work-class</td>
</tr>
<tr>
<td>3 Education</td>
</tr>
<tr>
<td>4 Marital status</td>
</tr>
<tr>
<td>5 Relationship</td>
</tr>
<tr>
<td>6 Occupation</td>
</tr>
<tr>
<td>7 Sex</td>
</tr>
<tr>
<td>8 Salary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table VII. Description of synthetic data set.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>1 Age</td>
</tr>
<tr>
<td>2 Sex</td>
</tr>
<tr>
<td>3 Zipcode</td>
</tr>
<tr>
<td>4 Disease</td>
</tr>
</tbody>
</table>
corresponding sensitive value. If the sensitive value is not compatible with the QI values, then we call the tuple is an invalid record.

### Table VIII. Negative association rule mines from adult data set.

<table>
<thead>
<tr>
<th>Conflicting column values</th>
<th>Reason with example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age, Education</td>
<td>Doctorate, masters, bachelor degree certificate cannot obtain before a certain age. (\text{Age} &lt; 25) ⇒ (\neg (\text{Education} = \text{Doctorate}))</td>
</tr>
<tr>
<td>Age, Relationship</td>
<td>Some relation could not happen under certain age. (\text{Age} &lt; 24) ⇒ (\neg (\text{Relationship}=\text{Husband or Wife}))</td>
</tr>
<tr>
<td>Work-class, Occupation</td>
<td>A government work class cannot be with private-house-serv. (\text{Work-class} = \text{Government}) ⇒ (\neg (\text{Occupation}=\text{Priv-house-serv}))</td>
</tr>
<tr>
<td>Marital status, Relationship</td>
<td>Never-married person cannot be a Husband or Wife. (\text{Marital-status} = \text{Unmarried}) ⇒ (\neg (\text{Relationship}=\text{Husband or Wife}))</td>
</tr>
<tr>
<td>Relationship, Sex</td>
<td>Husband could not be Female and vice versa. (\text{Relationship}=\text{Husband}) ⇒ (\neg (\text{Sex}=\text{Wife}))</td>
</tr>
</tbody>
</table>

### Table IX. Invalid record after slicing.

<table>
<thead>
<tr>
<th>Slicing</th>
<th>Column</th>
<th>Invalid records in percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>One Column</td>
<td>(Age), (Work-class), (Education), (Marital status), (Occupation), (Relationship), (Sex), (Salary)</td>
<td>47</td>
</tr>
<tr>
<td>Two Column</td>
<td>(Age, Education), (Work-Class, Occupation), (Marital status, Relationship), (Sex, Salary)</td>
<td>40</td>
</tr>
<tr>
<td>Three Column</td>
<td>(Age, Marital status, Relationship), (Education, Work-Class, Occupation), (Sex, Salary)</td>
<td>33</td>
</tr>
<tr>
<td>Four Column</td>
<td>(Age, Sex, Marital status, Relationship), (Education, Work-Class, Occupation), (Sex, Salary)</td>
<td>24</td>
</tr>
</tbody>
</table>

Figure 1 is the experimental result on synthetic data sets, and it shows the error rates on sliced [1] microdata table. The experiment result shows, when negative association rules mined records are increased, the error rate also increased in the sliced table. Slicing performs random permutation in each bucket, and random permutation creates invalid records. For example, a random permutation might create a record where a man can suffer from ovarian cancer. In the experiment, it is recorded as an invalid record.

### 4.3. Data utility

Data utility experiment is divided into two parts. In the first part, the data utility is measuring how many invalid records are created during the anonymization process for slicing and value swapping, and we exclude the invalid records to calculate the data utility. In the second part, the data utility is measured by the relative error in the aggregate query.

In Figure 2, the columnar bar denotes the data utility on the different anonymization techniques in different negative association rules mined records, and it is increased from 5% to 30% by the interval of 5%, and data utility is plotted on the y-axis. In Figure 2, data utility is shown in the range of 0% to 100%. On the y-axis, data utility are plotted for slicing [1,15–19], generalization for k-anonymity [3] and value swapping method with the corresponding negative association rules mined records on the
Data utility is calculated for generalization using a simple data utility metric [2], and for slicing and value swapping method based on the following formula:

$$\text{Data utility} = \frac{\text{Total records} - \text{Invalid records}}{\text{Total records}} \times 100\%$$

Figure 2 is the experimental result of the data utility based on information loss on the published synthetic data sets. Increasing of negative association rules mined records, decreases the data utility in the sliced microdata table. We observe that value swapping method has more data utility than slicing and others anonymization techniques. Value swapping method finds the invalid records before data publishing and performs swapping operation to make sure the publish data has not any invalid records. During anonymization, slicing techniques creates invalid records on the published microdata. For example, a data miner wants to know how many patients are suffered from ovarian or breast cancer. From Table II (anonymized by slicing), data miner can see four patients are suffered from ovarian or breast cancer. Now, if he wants to know how many female patients are suffered from ovarian or breast cancer. Then from Table II, data miner can see only two patients are suffered from ovarian or breast cancer, but it should be four patients. Moreover, we can conclude that slicing techniques are reducing the data utility in the published microdata table.

In the experimental analysis, the accuracy of aggregate query [24] is also evaluated for measuring data utility. It is possible to compute aggregate query operator, such as “COUNT,” “MAX,” “AVERAGE,” and so on. Only “COUNT” operator is evaluated, where the query predicate involves the sensitive values. The query is considered in the following form:

SELECT COUNT(*) FROM Table
WHERE $v_i \in V_i$ AND $\ldots v_{dim} \in V_{dim}$ AND $s \in V_s$

where $v_j(1 \leq j \leq dim)$ is the QI value for attribute $A_j$, $V_i \subseteq D_i$ and $D_i$ is the domain for attribute $A_j$, $s$ is the SA value, and $V_s \subseteq D_s$ and $D_s$ are the domains for the SA S. A query predicate is characterized by predicate dimension $dim$ and query selectivity $sel$. Dimension $dim$ indicating the number of QIs are in the predicate and selectivity $sel$ indicating number of values in each $V_j(1 \leq j \leq dim)$. Specifically, the size of $V_j(1 \leq j \leq dim)$ is randomly chosen from $0, 1, \ldots, sel \times |D_j|$. Each query is executed on the four tables: the original table, generalized table, sliced table, and the proposed value swapping table. Count is indicated from the original and anonymized tables. Original count indicates as $\text{orgcount}$ and anonymized count as $\text{anzcount}$, where $\text{anzcount}$ is generalized, slicing and swapping, respectively. Then average relative error is computed over all queries as [24]

$$\text{Relative error} = \frac{|\text{anzcount} - \text{orgcount}|}{\text{orgcount}} \times 100\%$$

In Figure 3, relative query error is plotted on the y-axis based on the QI selection. In the experiment, we have selected one, two, and three attributes as quasi-identifier, respectively, and calculated the relative query error on the anonymized table by slicing, generalization for $k$-anonymity and value swapping. For example, if we want to calculate the relative query error in slicing for Table 2, and corresponding query is

SELECT COUNT(*) FROM Table II
WHERE sex='F' AND (Disease='Ovarian Cancer')

From the query answer, there is only one female person is suffered from ovarian cancer. However, from the original table query answer, it will be two persons. Using relative error formula, it could be shown that slicing has 50% of relative query error for one attribute selection. For the experiment, all possible combinations of the query are made and executed through all anonymization tables and calculated the average relative query error. The relative query error is calculated and shown in Figure 3, where the value on y-axis denotes relative error percentage, and those on x-axis stand for different QI selection. The experimental result shows that value swapping has small relative errors compared with slicing and generalization techniques.
4.4. Attribute disclosure risks

We computed the attribute disclosure risks for the value swapping method with the slicing approach on the synthetic dataset with 30% negative association rules mined records. Attribute disclosure is computed on the 2-diversity, 3-diversity, and 4-diversity of the anonymized table. Figure 4 shows the experimental results for attribute disclosure on the anonymized table. In the figure, the value on y-axis shows the attribute disclosure in percentage, and the x-axis indicates the different diversity of the anonymized table. The experimental results show that value swapping method for 4-diversity has no attribute disclosure risk, and it is effective in decreasing the attribute disclosure risks, as discussed in Section 3.3.

4.5. Computational efficiency

To measure the computational efficiency, we compare the value swapping method with slicing and generalization for k-anonymity. We measure the computational efficiency on the synthetic dataset. In the experiment, we fix the table in 4-diversity and vary the number of records to calculate the efficiency. Figure 5 shows the computational times as a function of number of records. The experimental result shows that value swapping takes more time than generalization and slicing because we have to make sure the published table has no invalid records and satisfy l-diverse slicing by applying tuple check and value swapping algorithm.

5. CONCLUSIONS

In this paper, we started by observing the problems in the sliced microdata table, because of the invalid records created by the slicing, which reduce the data utility of the published table. With negative association rules, we proposed a new method called value swapping to increase the data utility of the sliced microdata table, by finding the invalid records in the sliced microdata table and swapping those values so that the published microdata table will not contain any invalid records. Besides, the value swapping method makes sure that published data will not breach the individual privacy and ensures the l-diverse slicing. Experiments were conducted to show the effectiveness of the value swapping method on sliced microdata tables, and the results show that it has more data utility and small relative query error compared with the existing slicing methods.

In the future research, we could improve the swapping algorithm such that it could consume less time. Slicing methods cannot protect identity disclosure; therefore, if we want to keep privacy for the identity disclosure risk in the sliced table, we need to generalize the attributes that will reduce the data utilities. In the future research, we can work on the identity disclosure risk that would ensure more privacy and would not reduce the data utilities.

ACKNOWLEDGEMENTS

This research work is supported by Guangdong-National Natural Science Foundation of China under Grant No. U1401258, Guangdong under Grant No. 2013B091300019, and Shenzhen Fundamental Research Foundation under Grant No.s CXZZ20150813155917544, JCYJ20150630114942277; and sponsored by CAS-TWAS President’s Fellowship for International Ph.D. students. The authors are grateful to the anonymous reviewers for their invaluable comments.

REFERENCES

5. Li T, Li N. Injector: mining background knowledge for data anonymization. In IEEE 24th International Conference on Data Engineering (ICDE 2008), IEEE, Cancun, Mexico, 2008; 446–455.
12. Li T, Li N, Zhang J. Modeling and integrating background knowledge in data anonymization. In IEEE 25th International Conference on Data Engineering (ICDE’09), IEEE, Shanghai, China, 2009; 6–17.