American Journal of Data Mining and Knowledge Discovery
2017; 2(2): 54-61
http://www.sciencepublishinggroup.com/j/ajdmkd
doi: 10.11648/j.ajdmkd.20170202.13

Efficient Anonymization Algorithms to Prevent Generalized Losses and Membership Disclosure in Microdata

Shivani Rohilla[1, *], Manish Bhardwaj[2]

[1]Department of Computer Science and Engineering, HRIT, Ghaziabad, India
[2]Department of Computer Science and Engineering, SRM University, Modinagar, India

Email address:
shivani.engineer@gmail.com (S. Rohilla), aapkaapna13@gmail.com (M. Bhardwaj)
[*]Corresponding author

To cite this article:
Shivani Rohilla, Manish Bhardwaj. Efficient Anonymization Algorithms to Prevent Generalized Losses and Membership Disclosure in Microdata. *American Journal of Data Mining and Knowledge Discovery*. Vol. 2, No. 2, 2017, pp. 54-61. doi: 10.11648/j.ajdmkd.20170202.13

Received: January 11, 2017; **Accepted**: January 25, 2017; **Published**: February 22, 2017

Abstract: Nowadays, data and knowledge extracted by data mining techniques represent a key asset driving research, innovation, and policy-making activities. Many agencies and organizations have recognized the need of accelerating such trends and are therefore willing to release the data they collected to other parties, for purposes such as research and the formulation of public policies. However, the data publication processes are today still very difficult. Data often contains personally identifiable information and therefore releasing such data may result privacy breaches, this is the case for the examples of micro-data, e.g., census data and medical data. This thesis studies how we can publish and share micro data in privacy-preserving manner. This present a next ensive study of this problem along three dimensions: Designing a simple, intuitive, and robust privacy model, designing an effective anonymization technique that works on sparse and high-dimensional data and developing a methodology for evaluating privacy and utility tradeoffs. Here, we present a novel technique called slicing which partitions the data both horizontally and vertically. It preserves better data utility than generalization and is more effective than bucketization in terms of sensitive attribute.

Keywords: Data Anonymization, Micro Data, PPDP, Slicing

1. Introduction

Data Anonymization is a technology that converts clear text into a non-human readable form. Data anonymization technique for privacy-preserving data publishing has received a lot of attention in recent years. Detailed data (alsocalledas micro-data) contains information about a person, a household or an organization. Most popular anonymization techniques are Generalization and Bucketization. There are number of attributes in each record which canbecategorized as 1) Identifiers such as Name or Social Security Number are the attributes that can be uniquely identify the individuals. 2) some attributes may be Sensitive Attributes (SAs) such as diseasend salary and 3) some may be Quasi-Identifiers(QI) such as pin code, age, and sex whose values, when taken together, can potentially identify an individual. Data anonymization enables the transfer of information acrossa boundary, such as between two departments with in an agency or between two agencies, while reducing the risk of unintended disclosure, and in certain environment inamanner that enables evaluation and analytics post-anonymization.

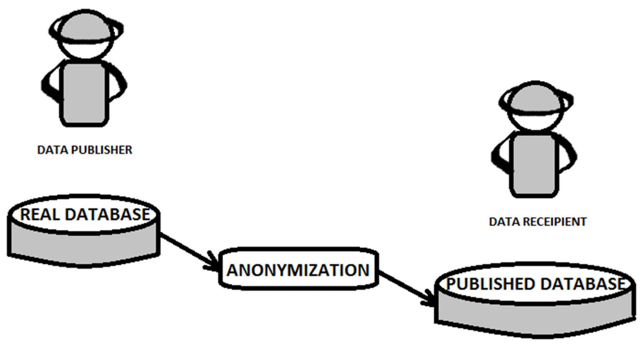


Figure 1. *A Simple Model of PPDP.*

2. Various Anonymization Techniques

Two widely studied data anonymization techniques are generalization and bucketization. The main difference between the two anonymization techniques lie in that bucketization does not generalize the QI attributes.

2.1. Generalization

Generalization is one of the commonly anonymized approaches, which replaces quasi-identifier values with values that are less-specific but semantically consistent. Then, all quasi-identifier values in a group would be generalized to the entire group extent in the QID space. If at least two transactions in a group have distinct values in a certain column (i.e. one contains an item and the other does not), then all information about that item in the current group is lost. The QID used in this process includes all possible items in the log. Due to the high-dimensionality of the quasi-identifier, with the number of possible items in the order of thousands, it is likely that any generalization method would incur extremely high information loss, rendering the data useless. In order for generalization to be effective, records in the same bucket must be close to each other so that generalizing the records would not lose too much information. However, in high-dimensional data, most data points have similar distances with each other. To perform data analysis or data miningtasks on the generalized table, the data analyst has to make the uniform distribution assumption that every value inageneralized interval/set is equally possible, as no other distribution assumption can be justified. This significantly reduces the data utility of the generalized data. And also because each attribute is generalized separately, correlations between different attributes are lost.

2.2. Bucketization

Bucketization is used to partition the tuples in T into buckets, and then to separate the sensitive attributes from then on-sensitive ones by randomly permuting the sensitive attribute values within each bucket. The sanitized data then consists of the buckets with permuted sensitive values. In this paper, we use bucketization as a method of constructing the published data from the original table T, although all our results hold for full-domain generalization as well. We now specify our notion of bucketization more formally. Partitioning the tuples into buckets (i.e., horizontally partition the tablet according to some scheme), and with in each bucket, we apply an independent an domper mutation to the column containing S values. The resulting e to buckets, denoted by B, is the published. For example, if the underlying tableis T, then the publisher might publish bucketization B. Of course, foradded privacy, the publisher can completely mask the identifying attribute (Name) and may partially mask some of the othernon-sensitive attributes (Age, Sex, Zipcode). While bucketization has better data utility than generalization, it has several limitations. First, bucketization does not prevent membership disclosure. Because bucketization publishes the QI values in the iroriginal forms, an adversary can findout whether an individual has are cord in the published data ornot. 87 percent of the individuals in the United States can be uniquely identified using only three attributes (Birthdate, Sex, and Zipcode). A micro-data (e.g., census data) usually contains many other attributes besides those three attributes. This means that he membership information of most individual scan be inferred from the bucketized table. Second, bucketization requires a clear separation between QIs and SAs. However, in many data sets, it is unclear which attributes are QIs and which are SAs. Third, by separating the sensitive attribute from the QI attributes, bucketization breaks the attribute correlations between the QIs and the SAs. Bucketization first partition stuples in the table in to buckets and then separates the quasi identifiers with the sensitive attribute by randomly permuting the sensitive attribute values in each bucket. The anonymized data consist of asset of buckets with permuted sensitive attribute values. There are some principles of privacy preserving as follows:-

3. K-Anonymity

Samarati and Sweeney introduced k-anonymity as the property that each record is indistinguishable with at least k-1 other records with respect to the quasi-identifier. In other words, k-anonymity requires that each QI group contains at least k records. k-anonymity is one of the most classic models, which prevents joining attacks by generalizing or suppressing portions of the released micro data so that no individual can be uniquely distinguished from a group of size k. k-Anonymity attributes are suppressed or generalized until each row is identical with at least k-1 other rows.

3.1. K-Anonymity Using Generalization

The generalization hierarchy transforms the k-anonymity problem into a partitioning problem. Specifically, this approach consists of two steps. The first step is to find a partitioning of the dimensional space, where n is the number of attributes in the quasi identifier, such that each partition contains at least k records. Then the records in each partition are generalized so that they all share the same quasi-identifier value. The generalization method substitutes the values of a given attribute with more general values. Generalization can be applied at the following levels:-

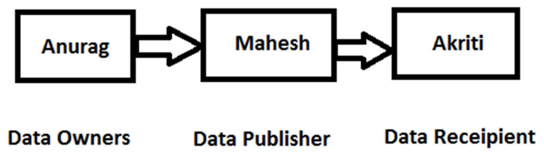


Figure 2. *Privacy preserving model for microdata.*

K-anonymity model for multiple sensitive attributes mentioned that there are three kinds of information disclosures:-

1) Identity Disclosure: When an individual is linked to a particular record in the published data called as identity disclosure.
2) Attribute Disclosure: When sensitive information regarding individual is disclosed called as attribute

disclosure.

3) Membership Disclosure: When information regarding individual's information belongs from data set is present or not is disclosed is said to be membership disclosure.

3.2. Attacks on K-Anonymity

Here, we will study two attacks on k-anonymity: the homogeneity attack and the background knowledge attack:-

1) Homogeneity Attack:

Sensitive information may be revealed based on the known information if the non sensitive information of an individual is known to the attacker. If there is no diversity in the sensitive attributes for a particular block then it occurs. To get sensitive information, this method is also known as positive disclosure.

2) Background Knowledge Attack:

If the user has some extra demographic information which can be linked to the released data which helps in neglecting some of the sensitive attributes, then some sensitive information about an individual might be revealing information. Such a method of revealing information is known as negative disclosure.

3.3. Limitations of K-Anonymity

1) K-anonymity cannot hide whether a given individual is in the database,
2) K-anonymity reveals individuals' sensitive attributes,
3) K-anonymity cannot protect against attacks based on background knowledge,
4) Mere knowledge of the k-anonymization algorithm can be violated by the privacy,
5) K-anonymity can notbe applied to high-dimensional data without complete loss of utility.

4. ℓ-Diverse Slicing

In the example given below, tuple t1 has only one matching bucket. In general, a tuple t can have multiple matching buckets. We now extend the above analysis to the general case and introduce the notion of ℓ-diverse slicing. Consider an adversary who knows all the QI values of t and attempts to infer t's sensitive value from the sliced table. He first needs to determine which buckets t may reside in, i.e., the set of matching buckets of t. Tuple t can be in any one of its matching buckets. Let p(t,B) is the probability that t is in bucket B (the procedure for computing p(t,B) will be described later in this section). For example, in the above example, p(t1,B1) = 1 and p(t1,B2) = 0. In the second step, the adversary computes p(t, s), the probability that t takes a sensitive value s. p(t, s) is calculated using the law of total probability. Specifically, let p(s|t,B) be the probability that t takes sensitive value s given that t is in bucket B, then according to the law of total probability, the probability p(t, s) is:

$$p(t, s) =\sum b\ p(t,B)p(s|t,B) \qquad (1)$$

In the rest of this section, we will show how to compute the two probabilities: p(t,B) an d p(s|t,B). Computing p(t,B):Given a tuple t and a sliced bucket B, the probability that t is in B depends on the fraction of t's column values that match the column values in B. If some column value of t does not appear in the corresponding column of B, it is certain that t is not in B. In general, bucket B can potentially match |B|c tuples, where |B| is the number of tuples in B. Without additional knowledge, one has to assume that the column values are independent; therefore each of the |B|c tuples is equally likely to be an original tuple. The probability that t is in B depends on the fraction of the |B|c tuples that match t.

We formalize the above analysis. We consider the match between t's column values {t[C 1], t[C2], · · ·, t[Cc]} and B's column values {B[C1],B[C2], · · ·,B[Cc]}. Let fi(t,B) (1 ≤ i ≤ c − 1) be the fraction of occurrences of t[Ci] in B[Ci] and let fc(t,B) be the fraction of occurrences of t[Cc −{S}] in B[Cc − {S}]). Note that, Cc − {S} is the set of QI attributes in the sensitive column. For example, in Table, f1(t1,B1) = 1/4 = 0.25 and f2(t1,B1) = 2/4 = 0.5. Similarly, f1(t1,B2) = 0 and f2(t1,B2) = 0. Intuitively, fi(t,B) measures the matching degree on column Ci, between tuple t and bucket B. Because each possible candidate tuple is equally likely to be an original tuple, the matching degree between t and B is the product of the matching degree on each column, i.e., f(t,B) = Q1_i_c fi(t,B). Note that Ptf(t,B) = 1 and when B is not a matching bucket of t, f(t,B) = 0. Tuple t may have multiple matching buckets, t's total matching degree in the whole data is f(t) = PB f(t,B).

The probability that t is in bucket B is: p(t,B) =f(t,B)/f(t)Computing p(s|t,B). Suppose that t is in bucket B, to determine t's sensitive value, one needs to examine the sensitive column of bucket B. Since the sensitive column contains the QI attributes, not all sensitive values can be t's sensitive value. Only those sensitive values whose QI values match t's QI values are t'scandidate sensitive values. Without additional knowledge, all candidate sensitive values (including duplicates) in a bucket are equally possible. Let D(t,B) be the distribution of t's candidate sensitive values in bucket B.

Definition of (D(t,B)). Any sensitive value that is associated with t[Cc − {S}] in B is a candidate sensitive value for t (there are fc(t,B) candidate sensitive values for t in B, including duplicates). Let D(t,B) be the distribution of the candidate sensitive values in B and D(t,B)[s] be the probability of the sensitive attributes in the distribution.

For example, in Table D (t1, B1) = (dyspepsia: 0.5, flu: 0.5) and therefore D (t1, B1) [dyspepsia] = 0.5. The probability p(s|t,B) is exactly D(t,B)[s], i.e., p(s|t,B) = D(t,B)[s]. ℓ-Diverse Slicing. Once we have computed p (t,B) a nd p(s|t,B), we are able to compute the probability p(t, s) based on the Equation(1). We can show when t is in the data, the probabilities that t takes a sensitive value sum up to 1. For any tuple t ∈ D, Ps p(t, s) = 1. ℓ-Diverse slicing is defined based on the probability p(t,s). Definition for ℓ-diverse slicing: A tuple t satisfies ℓ-diversity iff for any sensitive value s,p(t, s) ≤ 1/ℓ. A sliced table satisfies ℓ-diversity iff every tuple in it satisfies ℓ-diversity.

In the above example, tuple t1 has only one matching bucket. In general, a tuple t can have multiple matching buckets. We now extend the above analysis to the general case and introduce the notion of l-diverse slicing. Consider an adversary who knows all the QI values of t and attempts to infer t's sensitive value from the sliced table. He first needs to determine which buckets t may reside in, i.e., the set of matching buckets of t. Tuple t can be in any one of its matching buckets. Let p(t,B) be the probability that t is in bucket B. For example, in the above example, p(t1,B1)=1 and p(t1,B1)=0. In the second step, the adversary computes p(t,s), the probability that t takes a sensitive values. The probability for p(t,s) is calculated using the law of total probability. Let p(s|t,B) be the probability that t takes sensitive value s given that t is in bucket B according to the law of total probability, the probability p(t,s)is

$$P(t, s) = p(t,B)p(s|t,B)$$

Attacks on L-Diversity: In this section we will study two attacks on l-diversity: the Skewness attack and the Similarity attack:

1) Skewness Attack: l-diversity cannot prevent attribute disclosure whenever the overall distribution is skewed and satisfied.
2) Similarity Attack: When the sensitive attribute values are distinct but also semantically similar, an adversary can learn important information.

5. Slicing

Generally in privacy preserving, there is loss of security due to the presence of the adversary's background knowledge in real life application. Data contains sensitive information about individuals. These data when published violate the privacy. The current practice in data publishing relies mainly on policies and guidelines as to what types of data can be published and on agreements on the use of published data. The approach alone may lead to excessive data distortion or insufficient protection. Privacy-preserving data publishing (PPDP) provides methods and tools for publishing useful information while preserving data privacy. Many algorithms like bucketization, generalization have tried to preserve privacy however they exhibit attribute disclosure. So to overcome this problem an algorithm called slicing is used. Slicing partitions the dataset both vertically and horizontally. Slicing preserves better data utility than generalization and can be used for membership disclosure protection. Here we are using the following sub modules:

- Attribute partition and Columns
- Tuple Partition and Buckets
- Slicing
- Column Generalization
- Matching Buckets

a. Slicing Formalization and Analysis

Table 1 shows an example microdata table and its anonymized versions using various anonymization techniques. The original table is shown in Table 1(a). The three QI attributes are {Age, Sex, Zipcode}, and the sensitive attribute SA is Disease. A generalized table that satisfies 3 -anonymity is shown in Table 1(b), a bucketized table that satisfies 3-diversity is shown in Table 1(c), and sliced table is shown in Table 1(d). First the attributes are partitioned into columns. The column contains subset of attributes to vertically partition the table. Example, the sliced table in Table 1(d) contains 2 columns: the first column contains {Age, Sex} and the second column contains {Zipcode, Disease}.

Slicing partitions the tuples into buckets. Each bucket contains a subset of tuples to horizontally partition the table. Sliced table in Table 1(d) contain 2 buckets, each containing 3 tuples. Within each bucket, values in each column are randomly permutated to break the linking between different columns. Example in the first bucket of the sliced table shown in Table 1(d), the values {(25, M), (32, F), (40, F)} are randomly permutated and the values {(600016, ulcer), (6000116, cholera), (47905, cancer)} are randomly permutated so that the linking between the two columns within one bucket is hidden. Overlapped sliced table in Table 1 contains 2 buckets. Horizontal partitioning is done by duplicating the attributes in more than column so that the cross correlation between each column is break. In the first bucket of overlapped sliced table the original attribute in first column contains original values. The duplicate of the same attribute in the next column contains randomly permutated value. For example the first bucket in table 1.e values of the attribute sex {(25,F), (40,M), (32,F)} contains original values. In the nextcolumn duplicate attribute sex contains values {(F, 600017), (F, 600016), (M, 600017)} are randomly permutated.

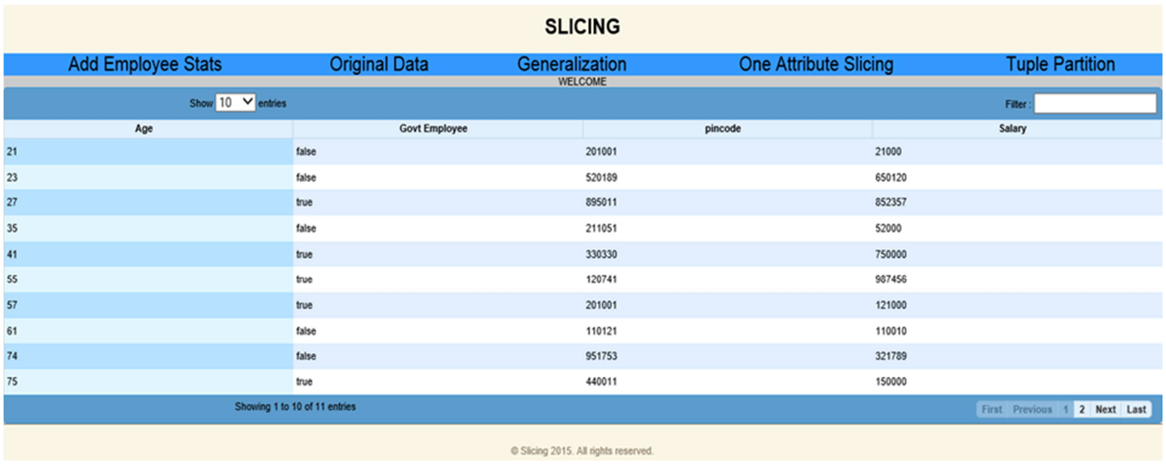
b. Methodology

The key intuition that slicing provides privacy protection is that the slicing process ensures that for any tuple, there are generally multiple matching buckets. Slicing first partitions attributes into columns. Each column contains a subset of attributes. Slicing also partition tuples into buckets. Each bucket contains a subset of tuples. This horizontally partitions the table. Within each bucket, values in each column are randomly permutated to break the linking between different columns. This algorithm consists of three phases: attribute partitioning, column generalization, and tuple partitioning.

6. Attribute Disclosure Protection

Based on the privacy requirement of ℓ-diversity slicing prevent attribute disclosure. We first give an example illustrating how slicing satisfies ℓ-diversity where the sensitive attribute is "Disease".

Real table on database:-



Age	Govt Employee	pincode	Salary
21	false	201001	21000
23	false	520189	650120
27	true	895011	852357
35	false	211051	52000
41	true	330330	750000
55	true	120741	987456
57	true	201001	121000
61	false	110121	110010
74	false	951753	321789
75	true	440011	150000

Showing 1 to 10 of 11 entries

© Slicing 2015. All rights reserved.

Figure 3. *Real Table image on Database.*

Table 1. *Original/Anonymous Tables (Example of Generalization/Bucketization/Slicing).*

a. Original Table

AGE	SEX	ZIPCODE	DISEASE
25	F	600016	Ulcer
32	F	600016	Cholera
40	F	600017	Cancer
49	M	600108	Cholera
57	M	600108	Flu
64	F	600093	Cancer

b. Generalized Table

AGE	SEX	ZIPCODE	DISEASE
[25-40]	*	60001*	Ulcer
[25-40]	*	60001*	Cholera
[25-40]	*	60001*	Cancer
[45-64]	*	60010*	Cholera
[45-64]	*	60010*	Flu
[45-64]	*	60009*	Cancer

c. Bucketized Table

AGE	SEX	ZIPCODE	DISEASE
25	F	600016	Cancer
32	F	600016	Ulcer
40	F	600017	Cholera
49	M	600108	Cancer
57	M	600108	Cholera
64	F	600093	Flu

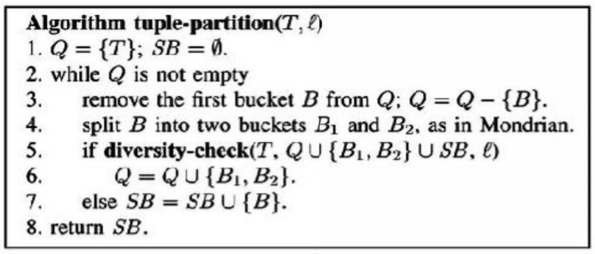
d. Sliced Table

(AGE,SEX)	(ZIPCODE,DISEASE)
(25,M)	(600016,cholera)
(32,F)	(600016,cancer)
(40,F)	(600017,ulcer)
(49,M)	(600093,cancer)
(57,M)	(600108,flu)
(64,M)	(600108,cholera)

e. Overlapped Sliced table

(AGE,SEX)	(SEX,ZIPCODE)	(ZIPCODE,DISEASE)
(25,M)	(F,600017)	(600016,cholera)
(40,M)	(F,600016)	(600017,ulcer)
(32,F)	(M,600016)	(600016,cancer)
(57,F)	(M,600093)	(600108,flu)
(64,M)	(F,600108)	(600093,cancer)
(49,M)	(M,600108)	(600108,cholera)

The sliced table shown in Table 1(d) satisfies 2-diversity. Consider tuple t1 with QI values (22, M, 60016). In order to determine t1's sensitive value, one has to examine t1's matching buckets. By examining the first column (Age, Sex) in Table 1(d), we know that t1 must be in the first bucket B1 because there are no matches of (22, M) in bucket B2. Therefore, one can conclude that t1 cannot be in bucket B2 and t1must be in bucket B1. Then, by examining the Zipcode attribute of the second column (Zipcode, Disease) in bucket B1. we know that the column value for t1 must be either (600016, cancer) or (600016, cholera) because they are the only values that match t1's zipcode 600017. Note that the other two column values have zipcode 600016. Without additional knowledge, both cholera and flu are equally possible to be the sensitive value of t1. Therefore, the probability of learning the correct sensitive value of t1 is bounded by 0.5. Similarly, we can verify that 2-diversity is satisfied for all other tuples in Table 1(d).



```
Algorithm tuple-partition($T, \ell$)
1. $Q = \{T\}$; $SB = \emptyset$.
2. while $Q$ is not empty
3.     remove the first bucket $B$ from $Q$; $Q = Q - \{B\}$.
4.     split $B$ into two buckets $B_1$ and $B_2$, as in Mondrian.
5.     if diversity-check($T$, $Q \cup \{B_1, B_2\} \cup SB$, $\ell$)
6.         $Q = Q \cup \{B_1, B_2\}$.
7.     else $SB = SB \cup \{B\}$.
8. return $SB$.
```

Figure 4. *Tuple-partition algorithm.*

The algorithm maintains two data structures:

1) a queue of buckets Q and
2) a set of sliced buckets SB. In the starting Q contains only one bucket which includes all tuples and SB is empty (line 1).

In each iteration (lines 2 to7), the algorithm removes a bucket from Q and splits the bucket into two buckets. If the sliced table after the split satisfies l-diversity (line 5), then the algorithm puts the two buckets at the end of the queue Q (for more splits, line 6). Otherwise, we cannot split the bucket anymore and the algorithm puts the bucket into SB (line 7). When Q becomes empty, we have computed the sliced table. The set of sliced buckets is SB (line 8). The main part of the tuple-partition algorithm is to check whether a sliced table satisfies l-diversity (line 5). Figure. 2 gives a description of the diversity-check algorithm. For each tuple t the algorithm maintains a list of statistics L[t] about t's matching buckets.

7. Attribute Partitioning

Highly correlated attributes are grouped together into one column in this attribute partitioning technique. There are three steps:

- *Equal Width Partitioning*

There are two types of attribute: continuous and categorical. So, in this step, continuous attribute are converted into categorical attribute. In equal width partitioning, we first divide the range into N intervals of equal size: uniform grid if A and B are the lowest and highest values of the attribute. Width of intervals will be W=(B-A)/N

- *Measures of Correlation*

Here, we calculate relation between two attributes. Let two attributes A_1 and A_2 with domains $\{V_{11},V_{12},\ldots\ldots V_1n_1\}$ and $\{V_{21},V_{22},\ldots\ldots V_2n_2\}$ respectively. Their domain sizes are thus n_1 and n_2. Therefore, Mean square contingency coefficient formula is used.

- *Attribute Clustering*

In this step, k-medoid clustering algorithm is used to partition attribute into columns as follows:-

The most common realization of k-medoid clustering is the Partitioning Around Medoids (PAM) algorithm:

```
Algorithm 1.1
1. Initialize: randomly select (without replacement) k of
   the n data points as the medoids
2. Associate each data point to the closest medoid.
   ("closest" here is defined using any valid distance
   metric, most commonly Euclidean distance, Manhattan
   distance or Minkowski distance)
3. For each medoid m
   For each non-medoid data point o
   Swap m and o and compute the total cost of the
   configuration
4. Select the configuration with the lowest cost.
5. Repeat steps 2 to 4 until there is no change in the
   medoid.
```

There can be a cluster based attribute slicing algorithm also as in existing systems, equal width discretization is used so it cannot handle skew data properly. So, to solve this problem, we proposed a new algorithm in proposed method, we use cluster based attribute algorithm for converting the continuous attribute into categorical attribute. This algorithm shows:

Input: Vector of real valued data $a=(a_1,a_2\ldots\ldots a_{11})$ and number of clusters to be determined k.

Goal: To find partition of data in k distinct clusters. Output: The set of cut points t_o, t_1……..tk with $t_o<t_1<\ldots\ldots t_n$ that defines discretization of adom(A).

```
Algorithm 1.2
1. Compute amax=max{$a_1,a_2,\ldots\ldots$an} and
   amin=min{$a_1,a_2\ldots\ldots\ldots$an}
2. Choose the centres as the first k distinct values of the
   attribute A.
3. Arrange them in increasing order i.e.
   c[1]<c[2]<………c[k].
4. Define boundary points bo=amin,
      bj = (c[j]+c[j+1]) /2 for j=1 to k-1, bk=amax
5. Find the closest cluster to ai.
6. Recompute the centres of the cluster as the average of
   the values in each cluster.
7. Find the closest cluster to ai from the possible clusters
   {j-1,j,j+1}
8. Determination of cut points:-$t_o$ = amin
   fori= 1to k-1 do
   ti=(c[i]+c[i+1]) /2
9. end for
10. tk=amax
11. Apply formula of measures of correlation
12. Apply attribute clustering algorithm
13. Apply attribute partitioning algorithm
```

```
Algorithm 1.3
Data slicing (QI, SA, B)
1. Add the Database T
2. Q={T};DSB=¢;
3. B, S={T*};QI={T-T*-key}
4. While Q is not empty
   Split Q into buckets B
   If total no. of records are <=100 Add fake tuples Else No
   need to add fake tuples
5. Q=Q- {B}
6. Sanitization of tuples by rule based id
7. Return DSB
```

- *Comparison with Bucketization*

To compare slicing with bucketization, we first note that bucketization can be viewed as a special case of slicing, where there are exactly two columns: one column contains only the SA, and the other contains all the QIs. The advantages of slicing over bucketization can be understood as follows. First, by partitioning attributes into more than two columns, slicing can be used to prevent membership disclosure. Our empirical evaluation on a real dataset shows that bucketization does not prevent membership disclosure. Second, unlike bucketization, which requires a clear separation of QI attributes and the sensitive attribute, slicingcan be used without such a separation. For dataset such as the census data, one often cannot clearly separate QIs from SAs because there is no single external public database that one can use to determine which attributes the adversary already knows. Slicing can be useful for such data. Finally, by allowing a column to contain both some QI attributes and the sensitive attribute, attribute correlations between the sensitive attribute and the QI attributes are preserved. For example, in table, Zipcode and Disease form one column, enabling inferences about their correlations. Attribute correlations are important utility in data publishing. For workloads that consider attributes in isolation, one can simply publish two tables, one containing all QI attributes and one containing the sensitive attribute.

8. Experimental Results Membership Disclosure Protection

Slicing protects against membership disclosure. We introduce a novel technique called overlapping slicing. Overlapping slicing duplicates an attribute in more than one column. This releases more attribute correlations within each column. Overlapped sliced table in Table 1.e contains 2 buckets. Horizontal partitioning is done by duplicating the attributes in more than column so that the cross correlation between each column is broke down. In the first bucket of overlapped sliced table the original attribute in first column contains original values. The duplicate of the same attribute in the next column contains randomly permutated value. Random permutation is implemented using Top-down refinement algorithm. For example the first bucket in Table 1.e values of the attribute sex {(25, F), (40, M), (32, F)} contains original values. In the next column duplicate attribute sex contains values {(F, 600017), (F, 600016), (M, 600017)} are randomly permutated. Let D be the set of tuples in the original data and let D_1 be the set of tuples that are in the duplicate attribute. Example consider the tuples in the attribute (Age, Sex) as D the original attribute and tuples in the attribute (Sex, Zipcode) are fake tuple because the tuples in the attribute Sex are duplicate of the original attribute. Let Ds be the sliced data∈. Goal∈of membership disclosure is to determine whether t D or t D_1. In order to distinguish tuples ∈ in D from tuples in D_1, we examine their differences. If t D, t must have at least one matching buckets in Ds. To protect membership information, we must ensure that at least some tuples in D should also have matching buckets ∈. Otherwise, ∈ the adversary can differentiate between t D and t D_1 by examining the number of matching buckets. We call a tuple an original tuple if it is in D. We call a tuple a fake tuple if it is in D_1 and it matches at least one bucket in the overlapped sliced data.

When the number of fake tuples is 0, the membership information of every tuple can be determined. Membership information is protected because the adversary cannot distinguish original tuples from fake tuples. Slicing is an effective technique for membership disclosure protection. A sliced bucket of size k can potentially match kc tuples. The existence of such tuples in D_1 hides the membership information of tuples in D because when the adversary finds a matching bucket, she or he is not certain whether this tuple is in D or not.

Our results show that, even when we do random grouping, many fake tuples have a large number of matching buckets. For example, for the OCC-7 dataset, for a small p = 100 and c = 2, there are 5325 fake tuples that have more than 20 matching buckets; the number is 31452 for original tuples. The numbers are even closer for larger p and c values. This means that a larger bucket size and more columns provide better protection against membership disclosure. Although many fake tuples have a large number of matching buckets, in general, original tuples have more matching buckets than fake tuples. As we can see from the figures, a large fraction of original tuples have more than 20 matching buckets while only a small fraction of fake tuples have more than 20 tuples. This is mainly due to the fact that we use random grouping in the experiments. The results of random grouping are that the number of fake tuples is very large but most fake tuples have very few matching buckets. When we aim at protecting membership information, we can design more effective grouping algorithms to ensure better protection against membership disclosure. The design of tuple grouping algorithms is left to future work.

9. Conclusion and Future Scope

Slicing overcomes the limitations of generalization and bucketization and preserves better utility while protecting against privacy threats. We illustrated how to use slicing to prevent attribute disclosure and membership disclosure. Protection against membership disclosure also helps to protect against identity disclosure and attribute disclosure. It is in general hard to learn sensitive information about an individual if you don't even know whether this individual's record is in the data or not. The general methodology proposed by this work is that: before anonymizing the data, one can analyze the data characteristics and use these characteristics in data anonymization. The rationale is that one can design better data anonymization techniques when we know the data better. We show that attribute correlations can be used for privacy attacks. We have also shown that cluster based attribute slicing can also be done to achieve attribute partitioning.

This work motivates several directions for future research. First, in this paper, we consider slicing where each attribute is in exactly one column. An extension is the notion of overlapping slicing, which duplicates an attribute in more than one columns. This releases more attribute correlations. For example, in Table 1(f), one could choose to include the Disease attribute also in the first column. That is, the two columns are {Age, Sex, Disease} and {Zipcode, Disease}. This could provide better data utility, but the privacy implications need to be carefully studied and understood. It is interesting to study the tradeoff between privacy and utility.

References

[1] C. Aggarwal, "On k -Anonymity and the Curse of Dimensionality," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 901-909, 2005.

[2] A. Blum, C. Dwork, F. McSherry, and K. Nissim, "Practical Privacy: The SULQ Framework," Proc. ACM Symp. Principles of Database Systems (PODS), pp. 128-138, 2005.

[3] J. Brickell and V. Shmatikov, "The Cost of Privacy: Destruction of Data-Mining Utility in Anonymized Data Publishing," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 70-78, 2008.

[4] B.-C. Chen, K. LeFevre, and R. Ramakrishnan, "Privacy Skyline: Privacy with Multidimensional Adversarial Knowledge," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 770-781, 2007.

[5] H. Cramt'er, Mathematical Methods of Statistics. Princeton Univ. Press, 1948.

[6] I. Dinur and K. Nissim, "Revealing Information while Preserving Privacy," Proc. ACM Symp. Principles of Database Systems (PODS), pp. 202-210, 2003.

[7] C. Dwork, "Differential Privacy," Proc. Int'l Colloquium Automata, Languages and Programming (ICALP), pp. 1-12, 2006.

[8] C. Dwork, "Differential Privacy: A Survey of Results," Proc. Fifth Int'l Conf. Theory and Applications of Models of Computation (TAMC), pp. 1-19, 2008.

[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis," Proc. Theory of Cryptography Conf. (TCC), pp. 265-284, 2006.

[10] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time," ACM Trans. Math. Software, vol. 3, no. 3, pp. 209-226, 1977.

[11] B. C. M. Fung, K. Wang, and P. S. Yu, "Top-Down Specialization for Information and Privacy Preservation," Proc. Int'l Conf. Data Eng. (ICDE), pp. 205-216, 2005.

[12] G. Ghinita, Y. Tao, and P. Kalnis, "On the Anonymization of Sparse High-Dimensional Data," Proc. IEEE 24th Int'l Conf. Data Eng. (ICDE), pp. 715-724, 2008.

[13] Y. He and J. Naughton, "Anonymization of Set-Valued Data via Top-Down, Local Generalization," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 934-945, 2009.

[14] A. Inan, M. Kantarcioglu, and E. Bertino, "Using Anonymized Data for Classification," Proc. IEEE 25th Int'l Conf. Data Eng. (ICDE), pp. 429-440, 2009.

[15] L. Kaufman and P. Rousueeuw, "Finding Groups in Data: An Introduction to Cluster Analysis," John Wiley & Sons, 1990.

[16] D. Kifer and J. Gehrke, "Injecting Utility into Anonymized Data Sets," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), pp. 217-228, 2006.

[17] N. Koudas, D. Srivastava, T. Yu, and Q. Zhang, "Aggregate Query Answering on Anonymized Tables," Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE), pp. 116-125, 2007.

[18] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Incognito: Efficient Full-Domain k -Anonymity," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), pp. 49-60, 2005.

[19] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional k -Anonymity," Proc. Int'l Conf. Data Eng. (ICDE), p. 25, 2006.

[20] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Workload-Aware Anonymization," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 277-286, 2006.

[21] N. Li, T. Li, and S. Venkatasubramanian, "t -Closeness: Privacy Beyond k -Anonymity and ℓ -Diversity," Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE), pp. 106-115, 2007.

[22] T. Li and N. Li, "Injector: Mining Background Knowledge for Data Anonymization," Proc. IEEE 24th Int'l Conf. Data Eng. (ICDE), pp. 446-455, 2008.

[23] T. Li and N. Li, "On the Tradeoff between Privacy and Utility in Data Publishing," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 517-526, 2009.

[24] T. Li, N. Li, and J. Zhang, "Modeling and Integrating Background Knowledge in Data Anonymization," Proc. IEEE 25th Int'l Conf. Data Eng. (ICDE), pp. 6-17, 2009.

[25] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "ℓ -Diversity: Privacy Beyond k -Anonymity," Proc. Int'l Conf. Data Eng. (ICDE), p. 24, 2006.

[26] D. J. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Y. Halpern, "Worst-Case Background Knowledge for Privacy-Preserving Data Publishing," Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE), pp. 126-135, 2007.

[27] M. E. Nergiz, M. Atzori, and C. Clifton, "Hiding the Presence of Individuals from Shared Databases," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), pp. 665-676, 2007.

[28] P. Samarati, "Protecting Respondent's Privacy in Microdata Release," IEEE Trans. Knowledge and Data Eng., vol. 13, no. 6, pp. 1010-1027, Nov./Dec. 2001.

[29] L. Sweeney, "Achieving k -Anonymity Privacy Protection Using Generalization and Suppression," Int'l J. Uncertainty Fuzziness and Knowledge-Based Systems, vol. 10, no. 6, pp. 571-588, 2002.

[30] L. Sweeney, "k -Anonymity: A Model for Protecting Privacy," Int'l J. Uncertainty Fuzziness and Knowledge-Based Systems, vol. 10, no. 5, pp. 557-570, 2002.

[31] M. Terrovitis, N. Mamoulis, and P. Kalnis, "Privacy-Preserving Anonymization of Set-Valued Data," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 115-125, 2008.

[32] R.C.-W. Wong, A.W.-C. Fu, K. Wang, and J. Pei, "Minimality Attack in Privacy Preserving Data Publishing," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 543-554, 2007.

[33] R.C.-W. Wong, J. Li, A.W.-C. Fu, and K. Wang, "(α, k)-Anonymity: An Enhanced k -Anonymity Model for Privacy Preserving Data Publishing," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 754-759, 2006.

[34] X. Xiao and Y. Tao, "Anatomy: Simple and Effective Privacy Preservation," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 139-150, 2006.

[35] Y. Xu, K. Wang, A.W.-C. Fu, and P. S. Yu, "Anonymizing Transaction Databases for Publication," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 767-775, 2008.