International Journal of Current Trends in Engineering & Technology
Volume: 02, Issue: 01 (JAN-FAB, 2016)

A Serve on Slicing: Method of Data Publishing While Maintaining Privacy

Devpal Yadav, Prof. Kailas Patidar, Prof. Manoj Yadav
Sri Satya Sai Institute of Science and Technology,
Sehore, Madhya Pradesh, India

***Abstract* — Latest work shows that abstraction loses amount of information for high spatial data. There are several anonymization techniques like Abstraction, Containerization for privacy preserving small data publishing. Bucketization does not avoid enrollment acknowledgment and does not give clear separation between aspects. We are presenting a technique called slicing for multiple columns multiple attributes which partitions the data both horizontally and vertically. We also show that slicing conserves better data service than generalization and Bucketization and can be used for enrollment acknowledgment conservation. Slicing can be used for aspect acknowledgment conservation and establishing an efficient algorithm for computing the sliced data that obey the l-diversity requirement Our workload confirm that this technique is used to prevent membership disclosure and it also used to increase the data utility and privacy of a sliced dataset by allowing multiple columns multiple attributes slicing while maintaining the prevention.**

Keyword: **- Data security, Data publishing, Privacy preservation, Data anonymization.**

I. INTRODUCTION

Micro data has been studied extensively in recent years for privacy preserving. Records of micro data, each of which contains information about an individual entity, such as a person, a household, or an organization there are many micro data anonymization techniques have been proposed. The most popular ones are generalization [9], [11] for k-anonymity [11] and Bucketization [12], [7], [3] for 'l-diversity [6]. In both approaches, attributes are partitioned into three categories:

1. Some attributes are identifiers that can uniquely identify an individual, such as Name or Social Security Number;
2. Some attributes are Quasi Identifiers (QI), which the adversary may already know (possibly from other publicly available databases) and which, when taken together, can potentially identify an individual, e.g., Birthdate, Sex, and Zip code;
3. Some attributes are Sensitive Attributes (SAs), which are unknown to the adversary and are considered sensitive, such as Disease and Salary.

In both generalization and Bucketization, first we remove identifiers from the data and then partitions tuples into buckets. The two techniques differ in the next step. Generalization transforms the QI-values in each bucket into "less specific but semantically consistent" values so that tuples in the same bucket cannot be distinguished by their QI values. In Bucketization, one separates the SAs from the QIs by randomly permuting the SA values in each bucket. The anonymized data consist of a set of buckets with permuted sensitive attribute values.

II. LITERATURE REVIEW

In [9], [11], [10], generalization has been proposed. Generalization replaces a value with a "less-specific

but semantically consistent" value. Three types of encoding schemes have been proposed for generalization: global recoding, regional recoding, and local recoding. In Global recoding [4] the property that multiple occurrences of the same value are always replaced by the same generalized value.

In Regional record [5] also called multidimensional recoding (the Mondrian algorithm) which partitions the domain space into non-intersect regions and data points in the same region are represented by the region they are in. Local recoding [13] does not have the above constraints and allows different occurrences of the same value to be generalized differently. The main problems with generalization are:

1. It fails on high-dimensional data due to the curse of dimensionality [1]
2. It causes too much information loss due to the uniform-distribution assumption [12].

In [12], [7], [3], Bucketization has been proposed. In Bucketization first partitions tuples in the table into 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 a set of buckets with permuted sensitive attribute values. In particular, Bucketization has been used for anonym zing high-dimensional data [2]. However, their approach assumes a clear separation between QIs and SAs. In addition, because the exact values of all QIs are released, membership information is disclosed.

III. EXISTING SYSTEM PROBLEM

Generalization for k-anonymity suffers from following three reasons because of which it losses considerable amount of information, especially for high-dimensional data. First, generalization for k-anonymity suffers from the curse of dimensionality. 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, forcing a great amount of generalization to satisfy k-anonymity even for relatively small ks. Second, in order to perform data analysis or data mining tasks on the generalized table, the data analyst has to make the uniform distribution assumption that every value in a generalized interval/set is equally possible, as no other distribution assumption can be justified. This significantly reduces the data utility of the generalized data. Third, because each attribute is generalized separately, correlations between different attributes are lost. In order to study attribute correlations on the generalized table, the data analyst has to assume that every possible combination of attribute values is equally possible. This is an inherent problem of generalization that prevents effective analysis of attribute correlations. Bucketization has better data utility than generalization, but has some limitations as follows:

First, Bucketization does not prevent membership disclosure [8]. Because Bucketization publishes the QI values in their original forms, an adversary can find out whether an individual has a record in the published data or not. As shown in [11], 87 percent of the individuals in the United States can be uniquely identified using only three attributes (Birthdate, Sex, and Zip code). A micro data (e.g., census data) usually contains many other attributes besides those three attributes. This means

that the membership information of most individuals can be inferred from the packetized 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.

IV. SOLUTION FOR THE EXISTING SYSTEM PROBLEM

We introduce a novel data anonymization technique called slicing to improve the current state of the art. Slicing partitions the data set both vertically and horizontally. Vertical partitioning is done by grouping attributes into columns based on the correlations among the attributes. Each column contains a subset of attributes that are highly correlated. Horizontal partitioning is done by grouping tuples into buckets. Finally, within each bucket, values in each column are randomly permutated (or sorted) to break the linking between different columns.

The basic idea of slicing is to break the association cross columns, but to preserve the association within each column. This reduces the dimensionality of the data and preserves better utility than generalization and Bucketization. Slicing preserves utility because it groups highly correlated attributes together, and preserves the correlations between such attributes. Slicing protects privacy because it breaks the associations between uncorrelated attributes, which are infrequent and thus identifying. Note that when the data set contains QIs and one SA, Bucketization has to break their correlation; slicing, on the other hand, can group some QI attributes with the SA, preserving attribute correlations with the sensitive attribute.

The key intuition that slicing provides privacy protection is that the slicing process ensures that for any tuple, there are generally multiple matching buckets.

V. PROPOSED SYSTEM

We first give an example to illustrate slicing. We then formalize slicing, compare it with generalization and Bucketization, and discuss privacy threats that slicing can address.

Table 1 An Micro data Table and its Anonymized versions using various Anonyms zing Technique.

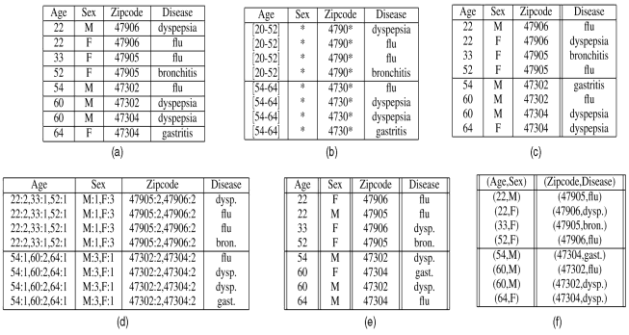

Table 1 shows an example micro data table and its anonymized versions using various anonymization techniques. The original table is shown in Table 1a. The three QI attributes are {Age: Sex; Zip code}, and the sensitive attribute SA is Disease. A generalized table that satisfies 4- anonymity is shown in Table 1b, a packetized table that satisfies 2-diversity is shown in Table 1c, a generalized table where each attribute value is replaced with the multistep of values in the bucket is shown in Table 1d, and two sliced tables are shown in Tables 1e and 1f. Slicing first partitions attributes into columns. Each column contains a subset of attributes. This vertically partitions the table. For example, the sliced table in Table 1f contains two columns: the first

column contains {Age; Sex} and the second column contains {Zip code; Disease}. The sliced table shown in Table 1e contains four columns, where each column contains exactly one attribute slicing also partition tuples into buckets. Each bucket contains a subset of tuples. This horizontally partitions the table. For example, both sliced tables in Tables 1e and 1f contain two buckets, each containing four tuples. Within each bucket, values in each column are randomly permutated to break the linking between different columns.

5.1 Formalization of Slicing

Let T be the micro data table to be published. *T* contains *d* attributes: A = {A1; A2…Ad} and their attribute domains are {D [A1]; D [A2]; . . .; D [Ad]}. A tuple t ϵ T can be represented as t = (t [A1]; t [A2]; . . . ; t [Ad]) where t [Ai] ($1 \leq i \leq d$) is the Ai value of t.

5.2 Attribute Partition and Columns

An attribute partition consists of several subsets of A, such that each attribute belongs to exactly one subset. Each subset of attributes is called a column.

5.3 Tuple Partition and Buckets

A tuple partition consists of several subsets of T, such that each tuple belongs to exactly one subset. Each subset of tuples is called a bucket.

5.4 Slicing

Given a micro data table T, a slicing of T is given by an attribute partition and a tuple partition.

For example, Tables 1e and 1f are two sliced tables. In Table 1e, the attribute partition is {{Age}, {Sex}, {Zip code}, {Disease}} and the tuple partition is {{t1; t2; t3; t4}, {t5; t6; t7; t8}}. In Table 1f, the attribute partition is {{Age, Sex}, {Zip code, Disease}} and the tuple partition is {{t1; t2; t3; t4}, {t5; t6; t7; t8}}.

5.6 Column Generalization

Column generalization ensures that one column satisfies the k-anonymity requirement. It is a multidimensional encoding [5] and can be used as an additional step in slicing. Specifically, a general slicing algorithm consists of the following three phases: attribute partition, column generalization, and tuple partition. Because each column contains much fewer attributes than the whole table, attribute partition enables slicing to handle high-dimensional data.

5.7 Matching Buckets

For example, consider the sliced table shown in Table 1f, and consider t1 = (22; M; 47906; dyspepsia). Then, the set of matching buckets for t1 is {B1}.

5.8 Comparison with Generalization

We now show that slicing preserves more information than such a local recoding approach, assuming that the same tuple partition is used. We achieve this by showing that slicing is better than the following enhancement of the local recoding approach. Rather than using a generalized value to replace more specific attribute values, one uses the multiset of exact values in each bucket. For example, Table 1b is a generalized table, and Table 1d is the result of using multisets of exact values rather than generalized values. For the Age attribute of the first bucket, we use the multiset of exact values {22, 22, 33, 52} rather than the generalized interval [22-52]. The multiset of exact values provides more information about the distribution of values in each attribute than the generalized interval. Therefore,

using multisets of exact values preserves more information than generalization. However, we observe that this multiset-based generalization is equivalent to a trivial slicing scheme where each column contains exactly one attribute, because both approaches preserve the exact values in each attribute but break the association between them within one bucket. For example, Table 1e is equivalent to Table 1d. Now comparing Table 1e with the sliced table shown in Table 1f, we observe that while one-attribute-per-column slicing preserves attribute distributional information, it does not preserve attribute correlation, because each attribute is in its own column. In slicing, one groups correlated attributes together in one column and preserves their correlation. For example, in the sliced table shown in Table 1f, correlations between Age and Sex and correlations between Zip code and Disease are preserved. In fact, the sliced table encodes the same amount of information as the original data with regard to correlations between attributes in the same column.

5.9 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 data set shows that Bucketization does not prevent membership disclosure.

Second, unlike Bucketization, which requires a clear separation of QI attributes and the sensitive attribute, slicing can be used without such a separation. For data set 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 1f, Zip code 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

5.10 Privacy Threats

For slicing, we consider protection against membership disclosure and attribute disclosure. It is a little unclear how identity disclosure should be defined for sliced data (or for data anonymized by Bucketization), since each tuple resides within a bucket and within the bucket the association across different columns are hidden. In any case, because identity disclosure leads to attribute disclosure, protection against attribute disclosure is also sufficient protection against identity disclosure.

We would like to point out a nice property of slicing that is important for privacy protection. In slicing, a tuple can potentially match multiple buckets, i.e., each tuple can have more than one matching buckets. This is different from previous work on generalization (global recoding specifically) and Bucketization, where each tuple can belong to a unique equivalence-class (or bucket).

V. CONCLUSION

Here we conduct a serve on slicing by using a method of data publishing while maintaining the privacy. Slicing overcomes the limitations of generalization and Bucketization and preserves better utility while protecting against privacy threats. We illustrate how to use slicing to prevent attribute disclosure and membership disclosure. Our experiments show that slicing preserves better data utility than generalization and is more effective than Bucketization in workloads involving the sensitive attribute. 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.

ACKNOWLEDGMENT

We would like to take this opportunity to express our profound gratitude and deep regard to my Guide Prof. Kailas Patidar, for his exemplary guidance, valuable feedback and constant encouragement throughout the duration of the project. I would like to thanks our principal and the entire staff member who directly and indirectly guide me. Never the less I also like to thanks my parents and friends to help me during the completion of this task.

REFERENCE

[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] G. Ghinita, Y. Tao, and P. Kalnis, "On the Anonymization of Sparse High-Dimensional Data," Proc. IEEE 24th International Conference Data Engineering (ICDE), pp. 715-724, 2008.

[3] N. Koudas, D. Srivastava, T. Yu, and Q. Zhang, "Aggregate Query Answering on Anonymized Tables," Proceedings IEEE 23rd International Conference Data Engineering. (ICDE), pp. 116-125, 2007.

[4] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Incognito: Efficient Full-Domain k-Anonymity," Proceedings ACM SIGMOD International Conference Management of Data (SIGMOD), pp. 49-60, 2005.

[5] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional k-Anonymity," Proceedings International Conference Data Engineering (ICDE), p. 25, 2006.

[6] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "'-Diversity: Privacy Beyond k-Anonymity," Proceedings International Conference Data Engineering (ICDE), p. 24, 2006.

[7] D.J. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J.Y. Halpern, "Worst-Case Background Knowledge for Privacy-Preserving Data Publishing," Proceedings IEEE 23rd International Conference Data Engineering (ICDE), pp. 126-135, 2007.

[8] M.E. Nergiz, M. Atzori, and C. Clifton, "Hiding the Presence of Individuals from Shared Databases," Proceedings ACM SIGMOD International Conference Management of Data (SIGMOD), pp. 665-676, 2007.

[9] P. Samarati, "Protecting Respondent's Privacy in Micro data Release," IEEE Transaction Knowledge and Data Engineering, vol. 13, no. 6, pp. 1010-1027, Nov./Dec. 2001.

[10] L. Sweeney, "Achieving k-Anonymity Privacy Protection Using Generalization and Suppression," international journal Uncertainty

Fuzziness and Knowledge-Based Systems, vol. 10, no. 6, pp. 571-588, 2002.

[11] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," international journal Uncertainty Fuzziness and Knowledge-Based Systems, vol. 10, no. 5, pp. 557-570, 2002.

[12] X. Xiao and Y. Tao, "Anatomy: Simple and Effective Privacy Preservation," Proceedings international conference Very Large Data Bases (VLDB), pp. 139-150, 2006.

[13] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A.W.-C. Fu, "Utility-Based Anonymization Using Local recoding," Proceedings 12th ACMSIGKDD international conference Knowledge Discovery and Data Mining (KDD), pp. 785-790, 2006.

AUTHOR'S PROFILE

Devpal Yadav Completed BE in Information Technology from North Maharashtra University, Jalgaon in 2011. He is a student at Sri Satya Sai Institute of Science and Technology, Sehore, MP, India. His area of Interest is networking, data mining and cryptography.