SEQUENTIAL CLUSTERING ALGORITHM WITH SLICING FOR ANONYMIZING SOCIAL NETWORKS

E.Mariammal¹, S.Selvakumar²

¹Pg Scholar, Department of CSE, Jayalakshmi Institute of Technology, India
²UGC Project Fellow, Department of Computer science, Periyar University, India

mariammalmecse@gmail.com, info_selva@yahoo.co.in

ABSTRACT:
We study the problem of privacy-preservation in social networks. The goal is to arrive at an anonymized view of the unified network without revealing to any of the data holders information about links between nodes that are controlled by other data holders. This paper produces anonymizations by means of clustering with better utility than sequential clustering algorithm. Sequential clustering is a local search algorithm. This may be repeated several times with different random partitions in order to find the best local minimum among those repeated partition. Several anonymization techniques, such as generalization and bucketization, have been designed for privacy preserving in social network. This paper, present a novel technique called slicing, which partitions the data both horizontally and vertically. Slicing preserves better data utility than generalization and can be used for attribute disclosure protection and it can have a clear separation between quasi-identifying attributes and sensitive attributes also handle high-dimensional data. Slicing preserves better utility than generalization and is more effective than bucketization also show that slicing can be used to prevent membership or attribute disclosure.

Keywords: Anonymization techniques, Generalization, social networks

1 INTRODUCTION
Networks are structures that describe a set of entities and the relations between them. A social network, for example, provides information on individuals in some population and the links between them, which may describe relations of friendship, collaboration, correspondence and so forth. An information network, as another example, may describe scientific publications and their citation links. In their most basic form, networks are modeled by a graph, where the nodes of the graph correspond to the entities, while edges denote relations between them. Real social networks may be more complex or contain additional information. For example, in networks where the described interaction is asymmetric (e.g., a financial transaction network), the graph would be directed; if the interaction involves more than two parties (e.g., a social network that describes co-membership in social clubs) then the network would be modeled as a hyper-graph; in case where there are several types of interaction, the edges would be labeled; or the nodes in the graph could be accompanied by attributes that provide demographic information such as age, gender, location or occupation which could enrich and shed light on the structure of the network.

Such social networks are of interest to researchers from many disciplines, be it sociology, psychology, market research. However, the data in such social network cannot be released as is, since
it might contain sensitive information. Therefore, it is needed to anonymize the data prior to its publication in order to address the need to respect the privacy of the individuals whose sensitive information is included in the data. Data anonymization typically trades off with utility.

Hence, it is required to find a golden path in which the released anonymized data still holds enough utility, on one hand, and preserves privacy to some accepted degree on the other hand. A naïve anonymization of the network, in the sense of removing identifying attributes like names or social security numbers from the data, is insufficient. The structure of the released graph may reveal the identity of the individuals behind some of the nodes. The idea behind the attack is to inject a group of nodes with a distinctive pattern of edges among them into the network. The adversary then may link this distinctive structure to some set of targeted individuals.

When the naïvely anonymized network is published, the adversary traces his injected subgraph in the graph; if successful (namely, there is only one such subgraph in the graph, an event of probability that can be made sufficiently high), the targets who are connected to this subgraph are re-identified and the edges between them are disclosed. Even less sophisticated adversaries may use prior knowledge of some property of their target nodes in order to identify them in the published graph and then extract additional information on them. Hence, one needs to apply a more substantial procedure of anonymization on the network before its release.

The methods of privacy preservation in networks fall into three main categories. The methods of the first category, provide k-anonymity via a deterministic procedure of edge additions or deletions. In those methods it is assumed that the adversary has a background knowledge regarding some property of its target node, and then those methods modify the graph so that it becomes k-anonymous with respect to that assumed property. The methods of the second category add noise to the data, in the form of random additions, deletions or switching of edges, in order to prevent adversaries from identifying their target in the network, or inferring the existence of links between nodes. The methods of the third category do not alter the graph data like the methods of the two previous categories; instead, they cluster together nodes into super-nodes of size at least k, where k is the required anonymity parameter, and then publish the graph data in that coarse resolution.

The study of anonymizing social networks with slicing, i.e., networks that are held by one data holder. However, in some settings, the network data is split between several data holders, or players and partition the data into horizontally and vertically. 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.

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 across columns, but to preserve the association within each column. This reduces the dimensionality of the data and preserves better utility than generalization and bucketization.

**Organization of the paper:** We begin by formal definitions in Section 2 and a survey of related work in Section 3. In Section 4 we stay in the realm of anonymizing of social network and propose the
slicing algorithm which is based on sequential clustering and comparing with generalization and bucketization. We conclude in Section 4 and 5 by outlining future work directions and the conclusion of this paper.

2. THE MODEL

2.1 The data

View of the social network as a simple undirected graph, \( G = (V,E) \), where \( V = \{v_1, \ldots, v_N\} \) is the set of nodes \( E = \binom{V}{2} \) is the set of edges1. Each node corresponds to an individual in the underlying group, while an edge that denotes the set of all unordered pairs of elements from \( V \) connects two nodes describes a relationship between the two corresponding individuals. In addition to the structural data that is given by \( E \), each node is described by a set of non-identifying attributes, such as age or zipcode that are called quasi-identifiers. Combinations of such attributes could be used for unique identification by means of linking attacks, whence, they should be generalized in order to thwart such attacks.

Let \( A_1, \ldots, A_I \) denote the quasi-identifiers, as well as the set of values that they may attain (e.g., if \( A_1 = \text{gender} \) then \( A_1 = \{M,F\} \)). Then each node \( v_n \), \( 1 \leq n \leq N \), is described by a quasi identifier record, \( R_n = (R_n(1), \ldots, R_n(I)) \in A_1 \times \cdots \times A_I \) . (an integral index is denoted by a lower-case letter, while the upper limit of that index is denoted by the corresponding upper-case letter.)

To summarize, a (naïvely anonymized) social network is defined as follows:

**Definition:** Let \( A_1, \ldots, A_I \) be a collection of quasiidentifier attributes. A social network over \( V = \{v_1, \ldots, v_N\} \) is \( SN = (V,E,R) \) where \( E = \binom{V}{2} \) is the structural data (edges), describing relationships between individuals in \( V \), and \( R = \{R_1, \ldots, R_N\} \), where \( R_n \in A_1 \times \cdots \times A_I \), \( 1 \leq n \leq N \), are the descriptive data of the individuals in \( v \).

An example of a network of seven nodes, with two dimensional quasi-identifier records (age and gender), and a corresponding partition network with three super-nodes, is given in Figure 1. (The two numbers in each super-node are its size and the number of intra-cluster edges.)

![Figure 1: A network and the corresponding partition](image)

Given a social network \( SN = (V,E,R) \), a corresponding clustered social network is called \( k \)-anonymous (or a \( k \)anonymization of \( SN \)) if the size of all of its clusters is at least \( k \). Our goal is to find a \( k \)-anonymization in which the loss of information is minimal (or, in other words, the utility is maximal). To that end, we need to define measures of information loss.

3. ANONYMIZATION BY SLICING

Data mining is the extracting the meaningful information from the large data sets such as data warehouse data contains records each of which contains information about an individual entity.
Several social network data anonymization techniques have been introduced. The most popular that generalization for k-anonymity and bucketization for l-diversity. In both approaches attributes are partitioned into three categories:

1) some attributes are identifiers that can uniquely identify an individual like Name or Social Security Number, zip code, phone number, age etc.,

2) some attributes are Quasi Identifiers (QI), which the adversary may already know and which, when taken together, can potentially identify an individual, that tuples of the same bucket cannot be distinguished by their QI values. In bucketization separates the SAs from the QIs but randomly permuting the SA values in each bucket, e.g., Birth date, Sex, and Zip code;

3) some attributes are Sensitive Attributes (SAs), which are unknown to the adversary and are considered sensitive. Generally when the data publishing the various attacks occurred like record linkage model attack and attribute linkage model attack. So avoid these attacks the various anonymization techniques was introduced.

In both generalization and bucketization removes the identifiers from the data and also partitions tuples into buckets. Buckets contain the subset of tuples. Generalization transforms the QI values in each bucket into “less specific but semantically consistent” values. So that tuples of the same bucket cannot be distinguished by their QI values. In bucketization separates the SAs from the QIs but randomly permuting the SA values in each bucket.

Generalization for k-anonymity losses considerable amount of information, particularly for high-dimensional data. This is due to the following reasons. First, generalization for k-anonymity undergo from the curse of dimensionality. In order to generalization for effective records in the similar bucket must be close up to each other so that generalizing the records would not lose too much information. on the other hand, in high-dimensional data, most of data points have similar distances with each other, forcing a large amount of generalization to assure k-anonymity even for relative small k’s.

In order to study attribute correlations on the generalized table, the data analyst has to assume that each possible combination of attribute values is equally possible. This is a natural problem of generalization that avoid effective analysis of attribute correlations. While bucketization has better data utility than generalization, also it has several limitations. First, bucketization gives attention to membership disclosure. Because bucketization issues the QI values in their original forms, an opposition can find out whether an individual has a record in the published data or not. As shown in following table, some percents of the individuals in the United States can be uniquely identified using only three attributes (Birth_date, Sex, and Zip_code). A data (e.g., census data) usually includes many other attributes besides those three attributes. That means the membership information of most persons can be inferred from the bucketized table. Second, bucketization needed a clear separation between QIs and SAs. However, in many datasets, it is ambiguous which attributes are QIs and which are SAs. Third, separate the sensitive attribute from the QI attributes, bucketization split the attribute correlations between the QIs and the SAs.

In this paper, we introduce a novel data anonymization method called slicing to improve the current state of the art. In that method called slicing the dataset partitions into both vertically and horizontally. In that vertical partitioning is done by grouping attributes into columns based on the correlations among the attributes. Each of the columns contains a subset of attributes that are highly correlated. And the horizontal partitioning is done by grouping tuples into the buckets. Finally, within
each bucket, values in each column are randomly sorted to break the linking between different columns. The vital idea of slicing is to break the relationship cross columns, but to preserve the relationship within each column. Many existing clustering algorithms for e.g., sequential clustering algorithm require the calculation of the “centroids”. But there is no perception of “centroids” in our setting where each attribute forms a data point in the clustering space.

3.1 RELATED WORK:

Although there has been a notion of Anonymization in the existing algorithms, and various technique would possibly suit the real time databases. Here we discuss about the some methods/algorithms for Privacy preserving of micro data publishing.

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. There are several types of recodings for generalization.

The recoding that preserves the most information is local recoding. In local recoding, one first groups tuples into buckets and then for each bucket, one replaces all values of one attribute with a generalized value. Such a recoding is local because the same attribute value may be generalized differently when they appear in different buckets. The example table is follows:

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>Zipcode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>[20-52]</td>
<td>*</td>
<td>4790</td>
<td>Cancer</td>
</tr>
<tr>
<td>[20-52]</td>
<td>*</td>
<td>4790</td>
<td>Thyroid</td>
</tr>
<tr>
<td>[20-52]</td>
<td>*</td>
<td>4790</td>
<td>Thyroid</td>
</tr>
<tr>
<td>[20-52]</td>
<td>*</td>
<td>4790</td>
<td>Diabetes</td>
</tr>
<tr>
<td>[54-64]</td>
<td>*</td>
<td>4790</td>
<td>Cancer</td>
</tr>
<tr>
<td>[54-64]</td>
<td>*</td>
<td>4790</td>
<td>Nausea</td>
</tr>
<tr>
<td>[54-64]</td>
<td>*</td>
<td>4790</td>
<td>Cancer</td>
</tr>
<tr>
<td>[54-64]</td>
<td>*</td>
<td>4790</td>
<td>Thyroid</td>
</tr>
</tbody>
</table>

Figure 2: Generalized data

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 leak. 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.
3.2. DATA SLICING

Basic Idea of Data Slicing

This paper introduces a new method, called DATA SLICING. This method partitions the data both horizontally and vertically. 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. At last, within each bucket, values in each column are randomly permuted to break the association between different columns. The core idea of data slicing is to break the association across columns, but to preserve the association within each column. This reduces the dimensionality of the data and preserves better data utility than bucketization and generalization. Data analysis methods such as query answering can be easily viewed on sliced data. Data slicing method consists of four stages. They are

1. Partitioning attributes and columns
2. Partitioning tuples and buckets.
3. Generalization of buckets
4. Matching the buckets

In the first stage, an attribute partition consists of several subsets of $A$, where each attribute belongs to exactly one subset. A column is nothing but a subset of attributes. Consider only one sensitive attribute $S$, if the data contains multiple sensitive attributes, one can either consider them separately or consider their joint distribution. The column that contains sensitive attribute is called as the sensitive column. Remaining column contains only quasi identifying attributes.

In the second stage, partitioning of tuples is taken place, each tuple belongs to exactly one subset and the subset of tuples is called a bucket. In the third stage, column generalization is done. A column generalization maps each value to the region in which the value is contained. In the last stage we have to check whether the buckets are matching.

Slicing Algorithms:

Our algorithm consists of some phases: that are tuple partitioning and Diversity check. Now we describe these phases.
Functional and Slicing Architecture:

**Functional procedure**

Step 1: Extract the data set from the database.
Step 2: Removes the queue of buckets and splits the Bucket into two.
Step 3: computes the sliced table. Step 4: Diversity maintains the multiple matching Buckets.
Step 3: Random tuples are computed.
Step 5: Attributes are combined and secure data Displayed.

**1. Attribute Partitioning**

This algorithm partitions attributes so that highly correlated attributes are in the same column.
This is good for both utility and privacy. In terms of data utility, grouping highly correlated attributes preserves the correlations among those attributes. In terms of privacy, the association of uncorrelated attributes presents higher identification risks than the association of highly correlated attributes because the associations of uncorrelated attribute values is much less frequent and thus more identifiable.

**2. Column Generalization**

First, column generalization may be required for identity/membership disclosure protection. If a column value is unique in a column, a tuple with this unique column value can only have one matching bucket. This is not good for privacy protection, as in the case of generalization/bucketization where each tuple can belong to only one equivalence-class/bucket.

**3. Tuple Partitioning**

The algorithm maintains two data structures: 1) a queue of buckets Q and 2) a set of sliced buckets SB. Initially, Q contains only one bucket which includes all tuples and SB is empty. For each iteration, the algorithm removes a bucket from Q and splits the bucket into two buckets. If the sliced table after the split satisfies l-diversity, then the algorithm puts the two buckets at the end of the queue Q. Otherwise, we cannot split the bucket anymore and the algorithm puts the bucket into SB. When Q becomes empty, we have computed the sliced table. The set of sliced buckets is SB. Example for sliced data.
1. Diversity:

The main part of the tuple-partition algorithm is to check whether a sliced table satisfies ℓ-diversity 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. Each element in the list L(t) contains statistics about one matching bucket b, the matching probability p(t,B) and the distribution of candidate sensitive values d(t,B). The algorithm first takes one scan of each bucket b to record the frequency f(v) of each column value v in bucket b.

Then, the algorithm takes one scan of each tuple t in the table t to find out all tuples that match b and record their matching probability p(t,B) and the distribution of candidate sensitive values d(t,B) which are added to the list l(t). We have obtained, for each tuple t, the list of statistics L(t) about its matching buckets. A final scan of the tuples in t will compute the p(t,b) values based on the law of total probability.

4. Future Work

In this paper, we considered slicing where each attribute is in exactly one column. The extension is the notion of overlapping slicing which releases more attribute correlations. One could choose to include the one attribute in the first column also and the privacy implications need to be carefully understood. This could provide better data utility. The tradeoff between utility of data and data privacy is very interesting. Finally, even though we are having many number of anonymization technique, it remains a problem how to use the anonymized data. In our work randomly generated the associations between column values of a bucket.

5. Conclusion

In this paper, we present a new anonymization method that is data slicing for privacy preserving and data publishing. Data Slicing overcomes the limitations of generalization and bucketization and preserves better utility while protecting against privacy threats. We illustrate that how slicing is used to prevent attribute disclosures. The general methodology of this work is before data anonymization one can analyze the data characteristics in data anonymization. The basic idea is one can easily design better anonymization techniques when we know the data perfectly. Finally, we have showed some advantages of data slicing comparing with generalization and bucketization. Data slicing is a promising technique for handling high dimensional data. By partitioning attributes into columns, privacy is protected.

5. References:

