Privacy Preserving Data Obfuscation for Inherently Clustered Data

Rupa Parameswaran*

School of Electrical and Computer Engineering,
Georgia Institute of Technology
Atlanta, Georgia, USA
Phone: +1 650-506-6114
E-mail: rupa@ece.gatech.edu
*Corresponding author

Douglas M. Blough

School of Electrical and Computer Engineering, Professor
Georgia Institute of Technology
Atlanta, Georgia, USA
Phone: +1 404-385-1271
E-mail: doug.blough@ece.gatech.edu

Abstract: Privacy is defined as the freedom from unauthorized intrusion. The availability of public records along with intelligent search engines and data mining tools allow easy access to useful information. They also serve as a haven for individuals with malicious intent. This paper proposes an approach that protects the privacy of individual records while retaining the information content. The techniques that have been proposed for privacy protection so far either provide insufficient privacy or trade off too much useful information on account of privacy protection. This paper proposes an attack model to analyze the different types of privacy breaches, proposes a set of properties for good privacy protection, proposes a robust data protection technique, and compares the privacy and usability properties of the new technique with some of the existing techniques.

Reference to this paper should be made as: Rupa Parameswaran and Douglas M. Blough (xxxx) 'Privacy Preserving Data Obfuscation for Inherently Clustered Data', *Int. J. Information and Computer Security*, Vol. 1, No. 2, pp.xxx–xxx.

Biographical notes: Rupa Parameswaran received her B.Eng. degree in computer science and engineering from Bangalore Universty, India, in 2000, and her M.S. and PhD degrees in electrical and computer engineering at Georgia Institute of Technology, Atlanta, Georgia, USA, in 2002 and 2006, respectively. She is working in the Core Database Security Group at Oracle. She was an active member of the Georgia Tech Information and Computer Security Colloquium (GTISC) and is also a member of the IEEE and ACM. Her research interests include issues related to practical data privacy, framework for secure online transactions, building access and privacy policies for data manipulation and privacy

preserving data mining.

Douglas M. Blough received the B.S. degree in electrical engineering and the M.S. and Ph.D. degrees in computer science from The Johns Hopkins University, Baltimore, MD, in 1984, 1986, and 1988, respectively. Since Fall 1999, he has been Professor of Electrical and Computer Engineering at the Georgia Institute of Technology, where he also holds a joint appointment in the College of Computing. From 1988 to 1999, he was on the faculty of Electrical and Computer Engineering at the University of California, Irvine. Dr. Blough was Program Co-Chair for the 2000 International Conference on Dependable Systems and Networks (DSN) and the 1995 Pacific Rim International Symposium on Fault-Tolerant Systems. He has been on the Program Committees of numerous other conferences, was Associate Editor for IEEE Transactions on Computers from 1995 through 2000, and was Associate Editor for IEEE Transactions on Parallel and Distributed Systems from 2001 through 2005. His research interests include distributed systems, dependability and security, and wireless multihop networks.

1 Introduction

The concern over privacy of personal and sensitive information has led to the implementation of several techniques for hiding, obfuscating and encrypting sensitive information in databases. The need for privacy has led to the development of several data obfuscation (DO) techniques that provide privacy preservation at the cost of information loss. Most of the techniques cater to specific domains and perform well for a limited set of applications. In the absence of a standard for classifying DO techniques, comparison and performance analysis of the different techniques is not straightforward. The domain of interest in this research is data mining. Many data mining applications involve learning through cluster analysis. The term *Usability* refers to the usefulness of the transformed data. In this paper, usability is measured in terms of preservation of the inherent clustering of the original data. The need for an obfuscation technique that preserves privacy as well as usability of the transformed data has motivated the design, development, and preliminary performance analysis of a robust cluster-retaining DO technique in this research. The paper proposes the use of the *Reversibility Property* as a measure of privacy preservation. The privacy provided by the proposed data obfuscation technique, Nearest Neighbor Data Obfuscation (*NeNDS*), is evaluated and compared with other obfuscation techniques with respect to its reversibility and usability.

This paper is an extended version of a workshop paper presented at ICDM 2005 [22]. The main contribution of this paper is the design, development, and analysis of the proposed DO technique *NeNDS* as well as a hybrid *Geometrically Transformed* version called *GT-NeNDS*. The motivation for the choice of the DO technique as well as the description of the proposed technique is provided in Section 5. The definition of the *Reversibility Property*, the classification of different transformation techniques based on reversibility, and the evaluation of existing DO techniques is provided in Section 4.1. An experimental analysis of *NeNDS* is carried

out in Section 7 to study its cluster-preserving characteristics.

2 Motivation and Related Work

The abundance of information available online has resulted in a loss of individual privacy [7]. Several methods have been proposed and implemented for privacy preservation of sensitive data sets [14]. The term *data obfuscation* [3] is used as a generalization of approaches that involve distorting the data for privacy preservation and other purposes. One of the more common techniques is cryptography, where sensitive data is encrypted with a key and is accessible only to authorized users. In several applications, it is necessary to provide different levels of precision of data, based on the type of user requesting access. The encryption of data does not provide this capability. The usability of the data is therefore restricted only to a narrow set of users. Secure multi-party encryption techniques propose to perform computations on data in the encrypted form [24].

Privacy preservation by data randomization is based on adding a noise vector to the original data, thereby desensitizing the precise information content [2]. Data randomization mainly operates on a subset of database tables, fields, and records and is designed to maintain the statistical properties of a database. Unless the noise distribution follows the distribution of the original data, information regarding dependencies among the attributes is lost.

Data anonymization [15] attempts to classify data into fixed or variable intervals. The usefulness of the obfuscated data and the privacy factor are dependent on the choice of the interval. A large interval makes the data less useful, while an interval that is too small does not provide sufficient privacy protection of the data. K-anonymity [25],proposes a generalization and suppression approach to obtaining the required anonymity level: generalization replaces a value with a less specific value, while suppression does not release a value at all. The goal here is to ensure that each record in a database is indistinguishable from at least k other records in the database. K-anonymization has been proved to be an NP-hard problem [16]. Various algorithms, such as k-optimal anonymization [25], simulated annealing [27], and condensation-based k-anonymization [1], have been proposed to produce approximate solutions to the generalization/suppression problem. Another drawback of the anonymization technique is loss of information. The generalization approach categorizes quantitative information into intervals, thus reducing the granularity of the information. Furthermore, data entries that are not possible to generalize are suppressed. This leads to a complete loss of information regarding certain fields.

The term *usability*, also referred to as data utility, pertains to the usefulness of the data that has been obfuscated. The most important characteristics that must be preserved for data mining applications are the multivariate statistical distributions as well as the clustering property of the data. So far, the emphasis on data utility has been on preserving statistical inferences [4][8][9]. While researchers have focused on preservation of statistics as the only measure of data utility, cluster preservation is an equally important data utility metric that is often ignored. An optimum data obfuscation technique is one that preserves both these properties while still providing strong privacy preservation. One of the techniques that proposes to preserve usability while preserving privacy is geometric transformation [20] [21]. In

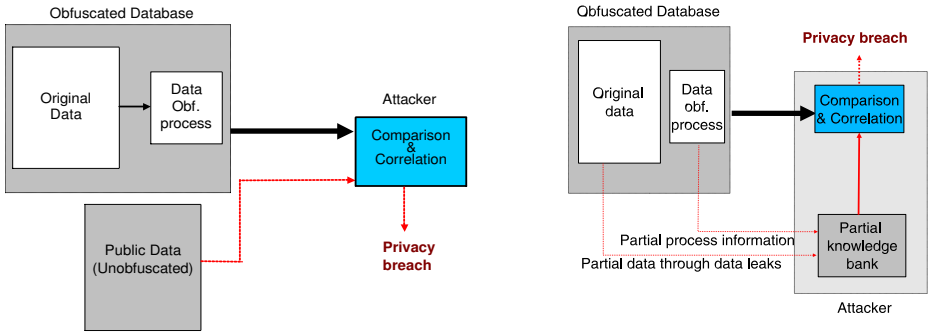
this approach, geometric transformations such as rotation, scaling, and translation are used to obfuscate the data. This type of obfuscation is proposed for preserving the inherent clustering information of data. As geometric transformations are isometric, the transformed data retains its isometric properties. While this technique does involve modifying data, the inter-relation of the data elements within the data sets and across the fields is maintained even after the obfuscation. This type of approach is useful in applications where the data needs to be disguised completely, such as the third-party mining of sensitive data. Geometric transformation-based obfuscation is weak in terms of privacy preservation and is unsuitable for use in sensitive databases. Random data perturbation, as well as anonymization, result in the modification of data. This results in slight differences of the characteristics between the original and obfuscated distributions. Such differences are likely to be very small for large data sets, but are observed to be significant for smaller data sets [19]. Owners of sensitive databases, such as the Census Bureau, look unfavorably upon such modifying data obfuscation techniques. Since the preservation of multi-variate characteristics while preserving privacy is an intractable problem, the next important statistical characteristic to preserve is the marginal distribution characteristics. One of the obfuscation techniques that has been widely adopted for sensitive data protection is data swapping [5][6][23][12]. The concept of obfuscating data sets by swapping the elements in the data set, proposed as early as 1978, intelligently swaps entries within a single field in a set of records so that the individual record entries are unmatched, but the statistics are maintained across the individual fields. Swapping can be implemented such that the swapped values are close to each other, thus approximating the information in the non-obfuscated data records [10]. As data swapping does not modify the actual values of the data, the characteristics of the marginal distributions of the variables are preserved exactly [19].

The requirement of preserving privacy as well as usability of sensitive data has led us to develop of a robust data obfuscation technique called *Nearest Neighbor Data Substitution (NeNDS)*. The underlying principle of this technique is a more generalized version of data swapping. In *NeNDS*, sets of data that are close to each other in Euclidean space are grouped together into neighborhoods. The data within each neighborhood are permuted in such a way that the original values are replaced by one of the neighbors in a non-reflective manner. The non-reflective condition is enforced to avoid the swapping of identical data items and to make it less vulnerable to reversal. This approach benefits from the advantages of data swapping, but is a more robust privacy-preserving scheme for data obfuscation.

3 Data Privacy Attack Model

The first part to developing a data obfuscation technique is to build an attack model to assess the vulnerable points that the attacker can use to compromise the database. There are ways of accessing a database. The attack models presented in this section assume that the database that the attacker queries is stored in obfuscated form. This means that the result of the database remains the same every time for the same query posed by the attacker. The discussion also assumes a query-based centralized database. However, the models are also applicable when the attacker has access to the entire obfuscated database.

The existing attack model considered for privacy evaluation is shown in Figure 1(a). Here, the attaker is assumed to have access to publicly available databases, some of which are unobfuscated. The attacker makes queries to the target database and compares the results of the query with the records in the databases to which he has access. The privacy preserving techniques proposed so far assume this type of attack model. A privacy breach results when an attacker is able to obtain previously unknown information about one or more records from the obfuscated database by comparing and correlating the database contents with other databases that he is able to access. Although this attack model is a likely scenario for compromise, it does not model all the modes of attack that lead to a breach of privacy.



(a) Existing model (b) Proposed model

Figure 1 Attack Models for Analysis

The existing model assumes that the attacker has no *a priori* knowledge of the process used for data obfuscation. It also assumes that the attacker does not know any of the entries in the target database. These are impractical assumptions. There are several reasons for assuming that the attacker may have access to some of the records in the database. In certain cases, the attacker is also likely to have *a priori* knowledge regarding the obfuscation process used for obfuscating the database.

The proposed attack model in Figure 1(b) includes side channels from the original target database and the data obfuscation process to model the partial information gained by the attacker. The attacker can then use this partial information to attempt to reverse engineer the entire data set. One useful byproduct of this model is a measure of the robustness of a data obfuscation technique, namely the percentage of the unobfuscated data set that an attacker must know in order to be able to learn the entire set. Using this new measure, we are able to demonstrate that many well-known data obfuscation techniques are highly vulnerable to reverse engineering through unintentional release of only a small percentage of the unobfuscated data set. We also propose to use the amount of information required for reverse engineering as a measure of privacy preservation for this attack model.

4 Data Obfuscation Properties

The term data privacy is broadly defined as the presence of appropriate safeguards to ensure the security and confidentiality of data records. To implement appropriate safeguards, it is necessary to first understand the nature of the application using the data as well as the security and confidentiality threats that need to be protected against. Hiding too much information results in loss of data usability, while insufficient protection poses a threat to data privacy. Although several Data Obfuscation (DO) techniques have been proposed for the protection of data privacy, no standard has been developed as yet for measuring or comparing DO techniques. This section identifies the different aspects of privacy protection that can be used to evaluate the strength of DO techniques.

4.1 Data Privacy

The definition of privacy is dependent on the type of data that needs to be protected as well as the target applications that use the data. In some privacy sensitive databases or applications, distortion of the original data such that it is similar but not exactly identical to the original data is considered as *acceptable privacy*. In other applications, any similarity between the obfuscated data and the original data is *unacceptable* and is equivalent to an invasion of privacy. Similarly, in applications such as medical databases, the breach of even a single record is *unacceptable*. However, in the case of publicly available databases, the breach of a small percentage of the total database is considered as *acceptable* privacy. Hence, any method that measures the strength of DO techniques needs to address the different requirements of privacy for different applications. The privacy measure proposed in this research measures the strength of DO techniques based on three aspects of privacy invasion.

- Approximate privacy invasion: This refers to the ability of an attacker to recover a value close to the value of an original data item being targeted.
- Absolute privacy invasion: This refers to the ability of an attacker to retrieve the original data exactly.
- Partial privacy invasion: This refers to the ability of an attacker to retrieve a portion of the original data either exactly or approximately from the obfuscated data.

Privacy against approximate invasion, absolute invasion, and partial invasion cover the basic requirements of privacy for any sensitive database or application that uses the sensitive data. The property of *reversibility* is proposed here to characterize the privacy provided by DO techniques with respect to the three aspects of privacy invasion. The term *Reversibility* is defined as the property that dictates the ease or difficulty of privacy invasion, i.e. the process of reverse engineering obfuscated data [3]. The reversibility property provides a measure of the robustness of privacy protection that is provided by a DO technique. The reversibility property exhibited by a DO technique can be measured by the *time* required for reverse engineering the data or by the amount of *a priori* information required to reverse engineer the

rest of the original data from the obfuscated data. In this research, the amount of *a priori* information that leads to a privacy breach is used as a measure of *reversibility* of DO techniques.

An obfuscation technique that can be reversed with the knowledge of the process is known as a process reversible transformation function. A model similar to the cryptanalytic attack model may be used for this category of transformations. Cryptanalysis of encryption techniques proves the weakness of algorithms to one or more of the well known attack models: plain-text attacks, chosen-plain-text attacks, and chosen-cipher-text attacks. Similarly, process reversible DO techniques can be analyzed with respect to their vulnerability to complete reversal under one or more of the following conditions: with no *a priori* information, with some *a priori* information of the DO process, and with complete *a priori* knowledge of the DO process. *Process reversibility* is sub-classified into the following categories.

1. Partial knowledge reversibility: *Partial knowledge reversibility* implies that a transformation function exhibiting this property can be reverse engineered with the knowledge of either some of the original data entries or a combination of some original entries of data and some information regarding the process used. The level of difficulty of the reversal process is dependent on the DO technique. Obfuscation techniques that involve a *one-to-one* mapping between the original and the transformed data, are vulnerable to *partial knowledge reversibility.* The *Reversibility* analyses for linear and non-linear one-to-one transformations are provided in Section 6.2.

2. Random number reversibility: This property indicates that the original data set can be reverse engineered with knowledge of the process, the *Pseudo-Random-Number Generator (PRNG)*, and the seed. Most obfuscation techniques invoke PRNGs to generate random sequences. The robustness of DO techniques exhibiting this property relies in protecting the PRNG sequence. As long as the random seed and the sequence are unknown to the attacker, the obfuscated data is robust to reversal. Once this information is revealed and the obfuscation process is known, the entire data is compromised. Transformations that fall under this category cannot be analyzed using cryptanalysis due to their non-deterministic nature.

Obfuscation techniques that result in a non-invertible data transformation exhibit *irreversibility.* A *Maximum-likelihood reversibility* estimate can be made in the case of some of the techniques, which provides an estimate of the confidence with which a guess can be made on the original data. Cryptanalysis fails to account for such transformations as well. With irreversible techniques, there is an inherent loss of information. Lossy compression techniques and data generalization techniques, which make it impossible to exactly recover the original data, fall under this category. The second category of *irreversibility* contains the set of obfuscation techniques in which a part of the obfuscated data becomes irreversible during the transformation. An example of *partial irreversibility* is substitution with repetition, where each data element is replaced by its nearest neighbor. Data elements that are not nearest neighbors of any other element are not included in the final data set and are lost completely. In such cases, the elements that are eliminated from the database cannot be exactly restored by any reversal process.

4.2 Data Usability

The term *Data Usability* refers to the ability of a DO technique to provide accurate aggregate information. An ideal DO technique is one that preserves both statistical information as well as clustering information. Data randomization techniques [2], which obfuscate data by the addition of random noise to the original data, can be tailored to preserve statistical information. However, the inherent clusters in the original data are distorted because of the addition of random noise. Data anonymization techniques [25], which categorize sets of k similar records by a process of suppression and generalization, can preserve statistical information for small values of k, but fail to preserve the original clusters because of the process of suppression and generalization. Geometric transformation techniques [20] obfuscate the data using linear transformations such as rotation, scaling, and translation. These transformations distort the statistical distributions of the data. Geometric transformation techniques preserve the original clusters by virtue of their linearity property and are suitable for data mining applications. Data swapping, which obfuscates by a process of swapping nearest neighbors, preserves statistical moments over individual datasets. NeNDS, which is the data obfuscation approach proposed in this research, obfuscates data by permuting amongst similar data items. *NeNDS* preserves all statistical moments over each dataset in the database, but fails to preserve multi-variate statistics. *NeNDS* also preserves the original clusters even after obfuscation.

5 Proposed Data Obfuscation Technique

This section provides a detailed description of the proposed DO technique called *Nearest Neighbor Data Substitution (NeNDS)*. Applications of the proposed technique lie in sensitive databases that require a data protection technique without loss of information content. Examples of such applications are medical records as well as micro-databases released by the Census Bureau, where the privacy of individuals is important as the correctness of the data provided to the end user [17]. The data substitution technique proposed here preserves privacy by permuting elements among groups of data items that are close to each other. Data substitution is performed individually for each field (dataset) in the database, and each field is permuted independently of the rest of the fields. *NeNDS* can be used for transformation of any data set that has some notion of distance among the elements. In other words, any dataset that forms a metric space can be transformed using *NeNDS*.

5.1 Nearest Neighbor Data-Substitution - NeNDS

NeNDS is a lossless DO technique that preserves privacy of individual data elements by substituting them with one of their neighbors in the metric space. A set of neighboring data elements are grouped together to form a neighborhood. The minimum number of neighbors that comprise a neighborhood is specified by the parameter c, where $1 < c < n - 1$, and n is the size of the data set. The minimum size of a neighborhood is given as $c + 1$, so that each data element in a

neighborhood has at least c neighbors. Hence, the number of neighborhoods in a data set (if all neighborhoods are at size exactly $c+1$) is given by $NH = \lfloor \frac{N}{c+1} \rfloor$. In the case where $c = 1$, each neighborhood would contain at least two neighbors, reducing the substitution technique to data swapping in some cases. The reflective nature of data swapping makes it vulnerable to privacy breaches in case of prior knowledge of some of the elements of the original data set. In order to strengthen the privacy preserving capability of *NeNDS*, c is set to be greater than 1.

Each field in the database is *NeNDS*-transformed independently of other fields in the database. Let Σ_{in} represent the original database of m attributes and n records, and Σ_{out} represent the *NeNDS* transformed database. The obfuscation technique performs substitutions on data items that lie close to each other within a single attribute field, so that the correlation of the data across the different attributes is not destroyed.

The neighborhoods created are likely to be of different sizes depending on the number of identical elements in each neighborhood. The algorithm uses a tree-traversal approach to obtain an optimum substitution pattern. The nodes of the tree correspond to the elements of a single data set with the first element as the root of the tree. The children are ordered from left to right based on their proximity to the parent node. The distance between the parent and child are given along the edge connecting them. A Depth First Search (DFS) approach is used here to traverse the tree. A maximum edge cost counter, C_{ME}, is maintained for each path being probed. An optimum substitution pattern is one that has the smallest cost C_{ME}. The substitution corresponding to the path chosen is the permutation used to replace the original data set.

Algorithm 5.1 shows the working of *NeNDS*. Σ_{in} is the input database with m attributes (fields) an n records. The minimum number of neighbors in each neighborhood, c, is the input parameter for the algorithm. Each individual dataset of the original and transformed database is denoted by ${\Sigma^i}_{in}$, ${\Sigma^i}_{out}$, respectively, where $i \in [1, m]$. Each dataset is divided into NH neighborhoods, denoted by NH_j, $j \in [1, NH]$. The recursive **CreateTree** algorithm is then invoked to build a $c - ary$ tree for each NH_j. The procedure **Ancestors**(Tree,NH_j) returns all the ancestors of a specified node, and the procedure **Identical**(Parent, NH_j) returns all the entries in NH_j that are identical to the parent of the specified node. *ChildrenTree* holds the set of valid children of the parent node in *Tree*. The populated tree is then assigned to the variable $Tree_j$ in Algorithm 5.1. All paths in $Tree_j$ that have a length equal to the size of the neighborhood are candidates for substitution. The maximum edge distance C_{ME} is determined for each candidate path. Procedure **min**(CandidateSet) identifies the path with the smallest C_{ME} as the optimum substitution pattern. This path is then assigned to NH'_j. The datasets $({\Sigma^1}_{out}, {\Sigma^2}_{out}, \ldots, {\Sigma^m}_{out})$ form the transformed database Σ_{out}, where ${\Sigma^i}_{out} = (NH'_1, NH'_2, \ldots, NH'_{NH})$. *NeNDS* can be performed on any data set in which the elements are related by some notion of distance, and can be expressed as a metric space. The algorithm is run for each field in the database that forms a metric space. An analysis of the DFS based algorithm for finding the substitution pattern indicates that the algorithm has an order of complexity that is exponential in neighborhood size. However, the branch and bound nature of the heuristic reduces the actual running time to a much smaller value, which is demonstrated later by the successful completion of *NeNDS* even for large data sets.

NeNDS(c)

1. For each $i \in [1, m]$ do
   (a) $NHsize = \lfloor \frac{N}{c+1} \rfloor$
   (b) $\Sigma^i{}_{in} = (NH_1, NH_2, \ldots NH_{NHsize})$
   (c) For each $NH_j \in \Sigma^i{}_{in}$ do
      i. $Tree_j$ = **CreateTree**(NH_j,0, NHsize)
      ii. $d_j = depth(Tree_j)$
      iii. For each $path_k$ in $Tree_j$ of length $d_j - 1$
         • CandidateSet = CandidateSet $\bigcup$ ($path_k$)
      iv. NH'_j = **min**(CandidateSet)
2. $\Sigma^i{}_{out} = (NH'_1, NH'_2, \ldots, NH'_{NHsize})$

CreateTree(NH_j, Tree, Size)

1. If $Tree = 0$ then Tree = NH[0]
2. If $NH_j = 0$ then Return Tree
3. ChildrenTree = NH_j - **Ancestors**(Tree,NH_j) - **Identical**(Parent, NH_j)
4. Child(Tree) = **Sort**(ChildrenTree)
5. Tree = Child(Tree)

The algorithm is explained with the help of an example. Consider a neighbohood [35 37 38 40 42] that has 5 elements. Figure 2 shows the tree-based permutation process for the neighborhood. The first item in the set is made the root of the tree. The remaining elements [37 38 40 42] become the children of the root node and are ordered from left to right based on their distance from the parent node. The distance between the parent and child is given along the edge connecting them. For instance, the edge distance between the root (35) and its left most child (37) is 2, which is assigned to the edge connecting the two nodes E_{35-37} and is called the cost of the edge $C_{E_{35-37}}$. Each child of the root becomes the root of a sub tree with all the data items that are not in the path from the root of the tree to the root of the sub tree becoming the children of this root. The nodes are expanded using a depth first approach from left to right, which means that the leftmost child of the root is expanded completely before the next child is expanded. In the figure, the first child (37) of the root (35) becomes the parent node for the nodes [38 40 42], which are not in its path from the root. The next node that is expanded is (38) because it is the leftmost child of the parent (37). The leftmost child is expanded each time until the leaf node is reached. The node (42) is the leaf node for the leftmost path in the figure. The root node is appended to the leaf and the distance between the two nodes is assigned as its edge distance. The edge with the largest edge cost, called C_{ME} is marked with a green box. This is the maximum cost of the specified path. The path ($35 \rightarrow 37 \rightarrow 38 \rightarrow 40 \rightarrow 42 \rightarrow 35$) with $C_{ME} = 7$ becomes a candidate for the permutation set. The next node that is expanded is the second child of the node (38), which is (42). The maximum edge cost for this path is $C_{ME} = 5$, which is less than the cost of the previous path and hence replaces the first path as the candidate permutation set. The new candidate set is

now $(35 \rightarrow 37 \rightarrow 38 \rightarrow 42 \rightarrow 40 \rightarrow 35)$ with $C_{ME} = 5$. The algorithm backtracks to the next unexpanded node, which is the second child of node (37), which is (40). The maximum edge cost for the leftmost path of this sub tree is $C_{ME} = 7$. Since the cost of the maximum edge for this path is greater than the cost of the candidate set, this path is ignored. The next path in the sub tree has an edge cost $C_{ME} = 3$, which is smaller than the maximum cost of the candidate set. The path $(35 \rightarrow 37 \rightarrow 40 \rightarrow 42 \rightarrow 38 \rightarrow 35)$ replaces the candidate set and becomes the new candidate with $C_{ME} = 3$. The next node to be expanded is the third child of (37), which is (42). The edge cost for this edge $C_{E_{37-42}} = 5$, which is larger than C_{ME}. Any path in this sub tree would have a maximum edge cost greater than or equal to 5. Hence the node is aborted and marked in yellow. The rest of the children of the root node have edge costs greater than the maximum edge cost of the candidate set $C_{ME} = 3$. As a result, none of these nodes are expanded. The final candidate set is $(35 \rightarrow 37 \rightarrow 40 \rightarrow 42 \rightarrow 38 \rightarrow 35)$, which is represented by a green arrow in the figure. The permutation set for the neighborhood [35 37 38 40 42] is obtained from the candidate $35 \rightarrow 37 \rightarrow 40 \rightarrow 42 \rightarrow 38 \rightarrow 35$ by replacing each item in the neighborhood by the item on the right of it in the candidate set. The permutation set for this neighborhood is [37 40 35 42 38].

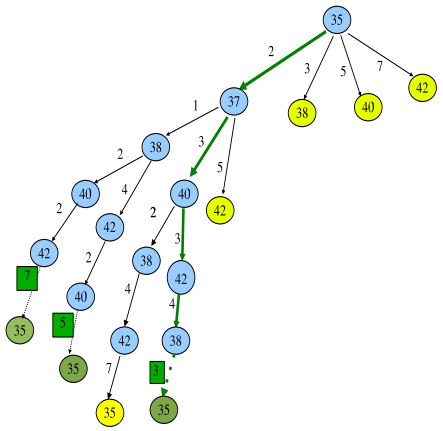


Figure 2 Permutation tree for the Age data set.

By considering only those nodes that are not yet in the path from the root of the tree to the root of the sub tree, the size of the tree is significantly reduced. This method also avoids swapping of data items. The maximum edge cost helps remove all those paths that would result in a less optimal solution, which reduces the complexity of the search process. Only those nodes that can lead to a minimum cost path are expanded. The depth first approach used here is derived from the Depth First Search tree traversal algorithm. Since the tree is finite, the DFS search will complete and will yield a solution.

NeNDS ensures a completely robust framework for data mining applications by preserving all the information content for cluster preservation and providing a secure and privacy preserving framework for drawing inferences on the data. As *NeNDS* preserves the original values of the data even after transformation, it is still vulnerable to privacy breaches as mentioned in [11]. This type of privacy breach may be unacceptable in highly sensitive databases. Section 5.3 provides a hybrid

version of *NeNDS* that preserves all the favorable characteristics of *NeNDS* and also overcomes this shortcoming of *NeNDS*.

5.2 Geometric Transformation Technique

An overview of the geometric transformation based DO proposed in [20] [21] is given here. This approach is of interest in data mining applications due to its inherent cluster preservation property. Hence, this technique will be used as a benchmark to evaluate the cluster retention capability of *NeNDS*. Transformations such as rotation, scaling and translation are used for distorting the data [13].

With geometric transformations, any pair of numerical fields in the database is interpreted as a two-dimensional space and the co-ordinates of the data items are distorted by geometric transformation. The approaches can also be scaled to three or more dimensions without loss of generality. The database is denoted by $D_{d,n}$, where d is the number of attributes and n represents the number of records or entries in the database. The transformations translation, scaling and rotation can be implemented using matrix multiplication. Each of the three transformations can be represented in terms of the equation $[X'\,Y']^T = A[X\,Y]^T + B$. In all of the transformations, A, B are the transformation matrices, (X, Y) are the original data, and (X', Y') are the results of the transformations on the original data. From the description of the transformations, it can be observed that each data set is distorted by the same amount relative to the placement of the individual elements in the set. In this way the clusters are maintained during obfuscation.

5.3 A Hybrid Data Substitution Approach

In this approach, referred to as *GT-NeNDS*, the data sets are first onfuscated using NeNDS, and then geometrically transformed. *NeNDS* provides a privacy preserving wrapper on the geometrically transformed data. The transformation functions like rotation and translation are isometric in nature, thereby preserving cluster information of the data sets and retaining the nearest neighbor information for the substitution step. NeNDS permutes similar sets of data items and is vulnerability to approximate privacy breaches. The additional geometric transformation performed on NeNDS-obfuscated results in a robust data obfuscation technique that protect data from absolute and approximate privacy breaches.

6 Analysis of Privacy for DO Techniques

Section 4.1 provides a classification of all transformation functions based on their reversibility property. Random data perturbation techniques are hard to reverse because they exhibit random number reversibility. Geometric transformations, being linear transformations, can be reversed with the knowledge of a finite number of original records. *NeNDS* involves a non-linear *one-to-one* transformation, and hence can also be reversed with the knowledge of sufficient number of original records. In this section, we derive the value for the minimum number of original records that are required to reverse engineer data that is obfuscated using

geometric transformations and *NeNDS*.

6.1 Analysis of Geometric Transformations

Geometric transformations fall under the category of linear transformation functions. These functions are the most vulnerable DO techniques that are subject to partial reversibility. A cryptanalysis of linear geometric transformations renders it weak to cipher-text only attacks. The knowledge of the type of obfuscation technique used results in an immediate reversal of the data. The linearity property of this data obfuscation technique preserves the clustered nature of the data, but also results in weak privacy protection. The assumption made here is that the attacker is aware that the DO process is a linear transformation. In this case, we prove that for a database with $d * n$ entries, where d is the number of attributes and n is the number of records, the knowledge of only $d+1$ linearly independent records in the original database, is sufficient to uniquely determine the linear transformation. Once the transformation matrix is obtained, all the original data entries for which the obfuscated values are available, are compromised. Therefore the geometric transformations of [20] [21], being instances of linear transformation functions, are compromised with the knowledge of $d+1$ linearly independent records in the original data [18].

6.2 NeNDS and Data Swapping

Data swapping and *NeNDS* fall under the category of non-linear bijective transformations. In this type of transformation, *reversibility* is dependent on the minimum number of records r that are sufficient for complete reverse engineering. In the case of data swapping, the minimum value for r is half the number of elements in the data set. For each element in the data set that is known *a priori*, the corresponding element involved in the swap is revealed.

In the case of *NeNDS*, complete reversal of the entire data set would require the knowledge of at least $r = c - 1$ distinct data elements for each neighborhood, where c is the minimum size of a neighborhood. Even partial reversal of a single neighborhood would require the knowledge of $c - 1$ of its elements. The fraction $\frac{c_i - 1}{c_i}$ determines the ease of reversal of a specific neighborhood i having exactly c_i elements. The proof for this claim is provided below. The goal of the attacker is to retrieve the original value corresponding to one of the obfuscated items in a dataset with absolute certainty. We refer to this as a targeted value attack.

Theorem 1. *Let $[X, Y]$ be the original and obfuscated datasets of size n respectively.*

$$X = x_1, x_2, \ldots, x_n \tag{1}$$

$$Y = y_1, y_2, \ldots, y_n \tag{2}$$

Let $y_t \in Y$ be the obfuscated item whose original value x_t the attacker wants to retrieve and let x_t belong to the p^{th} neighborhood. Assume that all c items in

the p^{th} neighborhood are distinct values. Assume that the attacker has complete knowledge of the NeNDS algorithm, including the value of neighborhood size c used to produce Y, but no additional knowledge except for a subset of the original data items. Then, the attacker needs to know at least $c-1$ original data items other than the targeted item to succeed in a targeted value attack.

Proof. Let $[X_p, Y_p]$ be the original and obfuscated data items in the p^{th} neighborhood.

$$X_p = x_{p1}, x_{p2}, \ldots, x_{pc} \tag{3}$$

$$Y_p = y_{p1}, y_{p2}, \ldots, y_{pc} \tag{4}$$

We evaluate what can be determined with the knowledge of at most $c-2$ original data items.

The only information known to the attacker:

$$X'_p = x_{p1}, x_{p2}, \ldots, U, \ldots, U, \ldots, x_{pc} \tag{5}$$

$$Y = y_1, y_2, \ldots, y_n \tag{6}$$

where X'_p is a set of $c-2$ original data items, and each U represents a missing value. The goal of the attacker is to identify two missing original values and determine which of these corresponds to the original value of y_t.

Case 1: There exist two items y_k, y_l in the dataset Y that fall within the interval $[min(Y_p), max(Y_p)]$. In this case, the attacker knows that y_k, y_l are the missing items in the neighborhood p. These two items can be placed in the neighborhood in two ways, both of which produce the same obfuscated neighborhood Y_p:

$$X'_p = x_{p1}, x_{p2}, \ldots, y_k, \ldots, y_l, \ldots x_{pc} \tag{7}$$

$$X''_p = x_{p1}, x_{p2}, \ldots, y_l, \ldots, y_k, \ldots x_{pc} \tag{8}$$

Since there is no additional information that enables the attacker to accurately identify which of the two sequences X'_p, X''_p is the original neighborhood, the attacker cannot determine with certainty whether y_k or y_l is equal to x_t.

Case 2: There are no items in the obfuscated data set that fall within the interval $[min(Y_p), max(Y_p)]$. In this case, the missing items are one of the three pairs: $min(Y_p)-2, min(Y_p)-1$, $max(Y_p)+1, max(Y_p)+2$ or $min(Y_p)-1, max(Y_p)+1$. For each pair, there are two permutations of the neighborhood that could be the original neighborhood. In this case, the original value corresponding to y_t can be one of 6 values, and the attacker cannot determine with certainty which of these corresponds to x_t.

Case 3: One item in the obfuscated dataset lies in $[min(Y_p), max(Y_p)]$. Let this item be denoted as y_{kl}. In this case, the missing items can be one of two pairs: $min(Y_p)-1, y_{kl}$ or $y_k l, max(Y_p)+1$. Each pair can fill up the missing positions in two ways. In this case, there are 4 candidates corresponding to the original value for y_t and again the attacker cannot know the value of x_t with certainty.

This shows that even with the knowledge of $c-2$ items in a neighborhood, the attacker cannot determine the original values of the remaining items with certainty. □

NeNDS with duplicates

In the presence of duplicate entries, the minimum size of a neighborhood with m duplicates is $c = 3m$. In this case, retrieving the original value of even a single obfuscated item requires *a priori* knowledge of at least $2m$ or $2c/3$ original items in the neighborhood containing the targeted original value. The minimum bound applies to cases where the unknown items are all duplicates. If the missing items are distinct, the minimum amount of information required is still $c-1$ items in the original neighborhood p that contains the targeted item. Even in the worst case for data with duplicate items, the attacker needs to know at least $2/3$ of the items in a neighborhood to be able to retrieve even a single targeted original value.

7 Experimental Results

The evaluation of usability is carried out using both real and synthetic data. Real data is obtained from the UCI repository that provides sample data for data mining applications. Two real databases are used in these experiments, the Diabetes database and the Thyroid database. Synthetic data is generated using IBMs Quest synthetic data generator. The inherent clustering degree C_{in} of the database to be generated can be specified as an input parameter, which enables the generation of databases with different clustering patterns. The other input specified to the data generator is the number of records required. The generator outputs a database with 9 fields and n records, where n is the number of records specified as the input.

A measure known as the *Misclassification Error Percentage* is used to compute the distortion produced by data obfuscation. The metric was proposed in [26] to evaluate the number of data points that have moved from one cluster to another. The average number of clusters that have moved from their original clusters is computed using Equation 9, where n is th e total number of records in the data set, $X : X \in D_{k,n}$ represents a data item with n fields, K is the number of clusters into which the data are grouped, and $Cluster_i(X)$ is the original cluster and $Cluster_i(X')$ is the new cluster formed from the obfuscated data.

$$MCE = \frac{1}{N} * \sum_{i=1}^{K} (|Cluster_i(X)| - |Cluster_i(X')|) \tag{9}$$

This section also evaluates the effect of the clustering results when different neighborhood sizes are used for NeNDS and GT-NeNDS. The tests are carried out for minimum neighborhood sizes NH_{size} varying from 1% of the database to 20% of the database for real and synthetic data. The different neighborhood sizes for each database are listed in Table 1. The maximum number of neighborhoods into which the data sets are partitioned are $NH = \lfloor \frac{N}{NH_{size}} \rfloor$, where n is the number of items in the data set. The maximum number of neighborhoods for each of the

databases specified in Table 1 are $[100, 50, 20, 10, 5]$ corresponding to the different neighborhood sizes $[1\%, 2\%, 5\%, 10\%, 20\%]$ of n respectively. Each data set of the database is divided into a maximum of NH neighborhoods. NeNDS is applied to each neighborhood of each data set to produce NH obfuscated neighborhoods for every data set.

Database	1 %	2 %	5 %	10 %	20 %
Synthetic (3000)	30	60	150	300	600
Synthetic (5000)	50	100	250	500	1000
Thyroid (7200)	72	144	360	720	1440

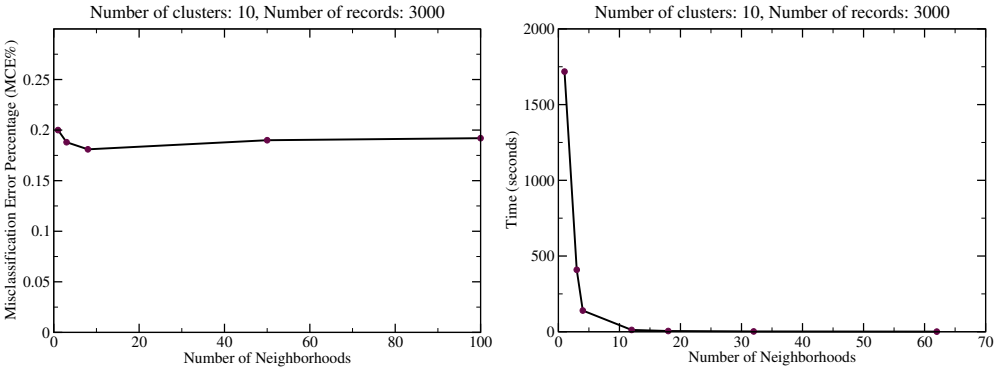
Table 1 Neighborhood sizes.

7.1 Neighborhood Size and Time Complexity

The distortions produced when clustering algorithms are applied to obfuscated data have been evaluated for two types of clustering techniques- K-means and hierarchical clustering. The results of the experiments show that NeNDS and GT-NeNDS produce very small distortions to the inherent clustering of the databases. Both types of clustering technique produced similar results for all the experiments. The effect of varying the neighborhood size for NeNDS was analyzed for two types of clustering algorithms and different neighborhood sizes. Clustering experiments were carried out for different neighborhood sizes to evaluate the effect of different neighborhood sizes on the creation of clusters. The experiments showed that the distortions introduced by changing the neighborhood sizes are very small.

Figure 3(a) shows a graph of the $MCE\%$ versus the number of neighborhoods, where the number of neighborhoods is increased from 1 to 100 for the synthetic database generated with an inherent clustering $C_{in} = 10$ and $3,000$ records. The number of neighborhoods NH is expressed as $\lfloor \frac{N}{NH_{size}} \rfloor$, where NH_{size} is the neighborhood size and N is the size of the data set. The graph shows the $MCE\%$ varies only slightly when the number of neighborhoods is increased from 1 to 100. The difference between the extreme values of $MCE\%$ is 0.02% indicating that by changing the neighborhood size from 1 to 100 alters less than 1 record in the database. Plotting the $MCE\%$ versus number of neighborhoods for different databases produced similar results. This shows that the choice of neighborhood size NH_{size} (or the number of neighborhoods NH) has little or no effect on the misclassification error.

Although the number of neighborhoods does not affect the $MCE\%$ of NeNDS significantly, the permutation of a large neighborhood is likely to take longer time to compute than for a smaller neighborhood. Figure 3(b) shows a graph of the computation time (t) versus the number of neighborhoods for a synthetic database containing $3,000$ records and an inherent clustering degree $C_{in} = 10$. The graphs show that the computation time decreases exponentially when the number of neighborhoods is increased. When the number of neighborhoods $NH = 1$, each data set (field) in the database is treated as a single neighborhood with $3,000$ data items. Increasing the number of neighborhoods decreases the size of each neighborhood. This results in reducing the size of the tree for computing the optimum permutation candidate. Although the graph is exponential in nature, the computation time



(a) MCE % vs. neighborhood size (b) Computation time vs. neighborhood size

Figure 3 Effect of neighborhood size on MCE % and computation time.

for the worst case ($NH = 1$, $NH_{size} = 3,000$) was completed in $1,700$ seconds. This manageable computation time is enabled by the heuristic branch and bound tree traversal algorithm for NeNDS. It is seen that by increasing the number of neighborhoods by a small number, a significant reduction in computation time is obtained, while the $MCE\%$ is hardly impacted.

7.2 Cluster Analysis

A quantitative evaluation of the distortions in clustering due to NeNDS-based data obfuscation is performed in this section using hierarchical and K-means clustering algorithms. The tabulated results represent the average distortion produced by the obfuscated data for the different neighborhood sizes specified in Table 1.

Tables 2 and 3 show the misclassification error resulting from performing clustering on the geometrically transformed database, NeNDS-obfuscated database, as well as GT-NeNDS-obfuscated database. For NeNDS and GT-NeNDS, the average MCE % indicated in the tables is the average MCE % obtained when NeNDS based obfuscation is performed with the different neighborhood sizes mentioned in Table 1.

Table 2 shows the effect that the obfuscation process has on the MCE % when k-means clustering is performed on the obfuscated data for $K = [10, 20, 30]$. The three columns in the table indicate the clustering errors produced due to a geometric transformation (in this case, rotation), NeNDS, and GT-NeNDS respectively. The results show that as the value of K increases, there is an increase in the MCE %. However, the absolute value of the MCE % is less than one percent, which is a small percentage. The MCE % resulting from NeNDS- and GTNeNDS- transformed data are only slightly higher values of MCE % than geometric transformations. This table shows that NeNDS-transformed data are still highly usable for K-means clustering.

Table 3 shows the effect that the obfuscation process has on the MCE % when

No.	Rot.	NeNDS	GT-NeNDS
K-means $K = 10$			
3000	0.04	0.04	0.04
5000	0.11	0.11	0.12
7200	0.24	0.25	0.27
K-means $K = 20$			
3000	0.13	0.13	0.14
5000	0.25	0.23	0.27
7200	0.45	0.49	0.49
K-means $K = 30$			
3000	0.27	0.30	0.30
5000	0.25	0.27	0.28
7200	0.82	0.83	0.90

Table 2 MCE% for GT, NeNDS and GT-NeNDS data using the K-means clustering algorithm.

No.	Rot.	NeNDS	GT-NeNDS
Hierarchical Clustering $K = 10$			
3000	0.14	0.13	0.14
5000	0.11	0.12	0.13
7200	0.28	0.30	0.32
Hierarchical Clustering $K = 20$			
3000	0.28	0.27	0.29
5000	0.11	0.13	0.14
7200	0.31	0.33	0.34
Hierarchical Clustering $K = 30$			
3000	0.28	0.30	0.31
5000	0.36	0.37	0.39
7200	0.81	0.85	0.88

Table 3 MCE% for GT, NeNDS, GT-NeNDS transformed data using Hierarchical clustering.

hierarchical clustering is performed on the obfuscated data. In this case, the hierarchical tree is cut at levels corresponding to $K = [10, 20, 30]$. The MCE % results for hierarchical clustering are similar to the MCE % obtained for K-means. This table shows that NeNDS-transformed data are still highly usable for hierarchical clustering as well.

The distortion of clusters produced by clustering data obfuscated using random data perturbation (RDP) is evaluated to assess the usability of randomized data. The RDP algorithm takes as input the mean and standard deviation $[\mu, \sigma]$ and generates a Gaussian distribution, which is then added to the original data set. Following this, a second Gaussian distribution with the same parameters is generated and subtracted from the noise added data set. The final database consists of the datasets to which a noise distribution is added and a second distribution with the same parameters is subtracted. Four randomized databases are generated, each with a different set of parameters for the noise distribution. The parameters chosen are $[0, 1], [0, 10], [0, 20], [1, 5]$. The first parameter produces a noise distribution with 0 mean and a standard deviation of 1. The resulting data sets are distorted by a very small value. The second and third parameter sets generate noise distributions that have 0 mean but larger deviations. The last parameter generates a skewed distribution because of the non-zero mean. The experiments are carried out for the two synthetic databases, $n = [3000, 5000]$ and the real database with $7,200$ records of Thyroid data.

Tables 4 and 5 show the $MCE\%$ results when the randomized databases are clustered using the K-means clustering algorithm and hierarchical clustering algorithm respectively for $K = [10, 20, 30]$. The misclassification error is very small 0.01% for randomized data with a small noise distribution [0.1]. For all other cases, the $MCE\%$ is much larger, ranging from 13.5% to 45.6%. The misclassification error is worse when the number of clusters is increased. The large percentage of data items that are displaced from their original clusters makes randomized DO unsuitable for clustering-based data mining applications. Randomization techniques provide good clustering for small values of σ. However, the privacy risk resulting

No.	[0, 1]	[0, 10]	[0, 20]	[1, 5]
K-means $K = 10$				
3000	0.02	13.5	11.4	15.5
5000	0.05	12.3	14.4	12.37
7200	0.05	18.1	12.32	11.2
K-means $K = 20$				
3000	0.04	0.18	22.3	31.7
5000	0.07	0.22	28.2	33.9
7200	0.08	25.3	32.6	35.4
K-means $K = 30$				
3000	0.1	33.2	0.50	44.5
5000	0.08	39.6	0.44	45.6
7200	0.07	30.3	43.7	42.1

Table 4 MCE % for randomized data using K-means clustering.

No.	[0, 1]	[0, 5]	[0, 20]	[1, 5]
Hierarchical Clustering $K = 10$				
3000	0.0	13.1	14.2	13.2
5000	0.08	12.1	13.1	13.8
7200	0.07	12.0	13.3	14.1
Hierarchical Clustering $K = 20$				
3000	0.02	19.1	18.4	20.4
5000	0.07	23.8	22.4	21.5
7200	0.06	21.9	24.8	25.4
Hierarchical Clustering $K = 30$				
3000	0.03	32.3	45.0	41.8
5000	0.05	36.4	41.7	45.5
7200	0.05	33.6	45.2	42.7

Table 5 MCE % for randomized data using Hierarchical clustering.

from small offsets, as discussed in Section 4.1 makes it unfeasible to use such small offsets for data randomization.

Table 6 shows a summary of the misclassification error for the different data obfuscation techniques. Random Data Perturbation (RDP) is performed by adding a noise vector of mean $\mu = 0$ and variance $\sigma^2 = 100$. The angle of rotation for rotation-based geometric transformation is 89.4 degrees. The value of k for *NeNDS* as well as *GT-NeNDS* is computed by finding the average performance for $NH =$ [50, 100, 150, 300, 1000]. The size of the database used for comparison is $n = 5,000$, and the inherent clustering factor $C_{in} = 10$. The error percentages resulting from k-means and hierarchical clustering are comparable, and an average of the two results is used in the table. The table provides a comparison of the misclassification error as a percentage. It is observed that *RDP* performs poorly for all cluster sizes, whereas the other obfuscation techniques are comparable. Although *rotation* provides the smallest error percentage, its vulnerability to reverse engineering makes it unusable for the data obfuscation of sensitive data. The performance of the hybrid data obfuscation approach is observed to be almost as good as geometric transformations. The robust privacy-preservation capability of GT-NeNDS makes it a more suitable candidate for data protection. The performance of the obfuscation techniques degrade if the number of clusters required is chosen as a number much larger than the inherent clustering of the data, as can be noted in the case where the number of clusters is 20. This is twice the value of C. The loss of information in this case is a necessary condition for privacy preservation to prevent individual records from being exposed. The results of the preliminary analysis indicate that *NeNDS* and $GT - NeNDS$ provide cluster preserving obfuscated data that is difficult to reverse-engineer.

The experimental evaluation provided here shows that the cluster-preservation capability of *NeNDS* is comparable to the inherent cluster-preserving geometrical transformations. The usability of a data obfuscation technique is defined in terms of its preservation of statistical distribution characteristics as well as its cluster-preservation capability. An ideal obfuscation technique would be one that preserves multi-variate distribution characteristics, but such a technique would be vulnerable

Obfuscation - Clusters	RDP [0,10]	RDP [0,1]	Rotation Random	NeNDS Average	GT-NeNDS Average
2	3.1	0.0	0.0	0.0	0.0
3	8.3	0.02	0.02	0.01	0.02
5	13.5	0.03	0.04	0.04	0.04
10	18.1	0.05	0.13	0.14	0.14
20	22.3	0.18	0.45	0.51	0.59
40	42.1	0.25	0.87	0.92	0.95
60	47.3	0.21	1.12	1.64	1.71

Table 6 Comparison of misclassification error %.

to privacy breaches. The next important statistical characteristics to be preserved are marginal distributions. *NeNDS* preserves marginal distributions of variables because the data is not modified. The cluster-preservation capability of *NeNDS* is evaluated experimentally in this section and found to be as good as geometrical transformations. The robustness of the privacy preservation of *NeNDS* is studied in Section 6.2. Although *NeNDS* falls under the non-linear bijective transformation, the large fraction of *minimum information* required for complete as well as partial reversal strengthens the privacy-preservation capability of this technique, making it very difficult to reverse engineer.

8 Conclusion

The main contributions of this paper are: (1) the proposal of a robust DO technique for clustered data, (2) the definition of a new measure of privacy preservation for DO techniques, and (3) the demonstration of the weak privacy provided by existing obfuscation techniques such as linear transformations and data swapping. Table 7 provides a comparison of *NeNDS* and *GT-NeNDS* with existing DO techniques with respect to four parameters: displacement, reversibility, preservation of statistics, and cluster preservation. The *Displacement* metric indicates the similarity between the absolute values of the original and obfuscated data. A low displacement implies high vulnerability to approximate privacy breaches. The *Reversibility* metric evaluates the amount of information required for retrieving the original data from the obfuscated data. The *Stat* metric evaluates the extent to which the statistical distributions of the original data are maintained. The cluster preservation property of the DO techniques is measured by the *Cluster* metric. A robust DO technique is one with *High* displacement, that is *Difficult* to reverse engineer, and that has *Good* cluster preservation.

Data randomization with small offsets (Random-Low) and high offsets (Random-High) is robust to absolute reversibility. The small offset of Random-Low makes it vulnerable to approximate privacy invasion and unsuitable for applications where approximate information is considered a breach. The large offset of Random-High makes it unsuitable for data mining applications because of the distortion of the original clusters. Data anonymization for small values of k (k-Anon-Low) and large values of k (k-Anon-High) perform similar to data randomization and are unsuit-

Obfuscation	Displacement	Reversibility	Stat	Cluster
Random-Low	Very Low	Very Difficult	Good	Fair
Random-High	High	Very Difficult	Good	Poor
k-Anon-Low	Very Low	Easy	Good	Fair
k-Anon-High	High	Difficult	Fair	Poor
Data Swapping	Low	Moderate $\lfloor \frac{N}{2} \rfloor$	Moments	Good
NeNDS	Low	Difficult $\lfloor \frac{cN}{c+1} \rfloor$	Moments	Very Good
Geo-Trans	High	Easy $d+1$	Poor	Very Good
GT-NeNDS	High	Difficult $> \lfloor \frac{cN}{c+1} \rfloor$	Poor	Very Good

Table 7 Comparison of DO techniques.

able for data that are used for data mining applications. The DO techniques data swapping, NeNDS, geometric transformations, and GT-NeNDS can be used for obfuscation in data mining applications. Data swapping and *NeNDS* are vulnerable to approximate reversal. Data swapping is vulnerable to absolute reversibility only if $\lfloor \frac{N}{2} \rfloor$ of the data elements in a database of size n are known *a priori*. The amount of *a priori* information that leads to complete reversal of a neighborhood of *NeNDS* is $\lfloor \frac{cN}{c+1} \rfloor$, where n is the number of records in the database, and $c+1$ is the size of each neighborhood that is permuted. Reversal of an entire dataset requires the knowledge of the permutation pattern of all the neighborhoods into which the data is distributed. Geometric transformations offer very little resistance to privacy and are unsuitable for use by themselves. GT-NeNDS, which combines *NeNDS* and geometric transformations, provides robust protection against approximate privacy invasion as well as absolute reversibility. GT-NeNDS preserves the original clusters and also preserves moments over individual datasets.

References and Notes

1 C. Aggarwal and P. Yu. A Condensation Approach to Privacy Preserving Data Mining. In *Advances in Database Technology - EDBT 2004*, pages 183–199, 2004.

2 R. Agrawal and S. Ramakrishnan. "Privacy-Preserving Data Mining". In *ACM Special Interest Group on Management of Data*, pages 439–450, 2000.

3 D. Bakken, R. Parameswaran, and D. Blough. "Data Obfuscation: Anonymity and Desensitization of Usable Data Sets". *IEEE Security and Privacy*, 2(6):34–41, Nov-Dec 2004.

4 J. Burridge. "Information Preserving Statistical Obfuscation". In *Statistics and Computing*, pages 321–327, 2003.

5 T. Dalenius and S. P. Reiss. "Data-swapping A Technique for Disclosure Control". In *American Statistical Association Proceedings of the Section on Survey Research Methods*, pages 191–194, 1978.

6 T. Dalenius and S. P. Reiss. "Data-swapping A Technique for Disclosure Control". In *Journal of Statistical Planning and Inference*, pages 73–85, 1982.

7 D. Denning and M. Schwartz. "The Tracker A Threat to Statistical Database Security". In *ACM Transactions on Database Systems*, volume 4, pages 76–96, 1979.

8 J. Domingo-Ferrer, V. Torra, J. Mateo-Sanz, and F. Seb'e. Emperical disclosure risk assessment of the ipso synthetic generators. In *Eurostat Work Session on Statistical Data Confidentiality*, pages 227–238, 2005.

9 G. T. Duncan and R. W. Pearson. "Data-swapping Variation on a Theme by Dalenius and Reissenhancing access to microdata while protecting confidentiality. In *Prospects for Future Statistical Science*, volume 6, pages 219–232, 1991.

10 V. Estivill-Castro and L. Brankovic. "Data Swapping: Balancing Privacy Against Mining of Association Rules". In *Proc. of Knowledge Discovery and Data Warehousing*, pages 389–398, Florence, Italy, Aug 1999.

11 A. Evfimievski, J. Gehrke, and R. Srikant. "Limiting Privacy Breaches in Privacy Preserving Data Mining". In *Principles of Database Systems*, San Diego, CA, June 2003.

12 S. Gomatam and A. Karr. "Distortion Measures for Categorical Data Swapping". Technical Report 131, US National Institute for Statistical Sciences, Jan 2003.

13 R. Gonzalez and R. Woods. *"Digital Image Processing"*. Addison-Wesley Publishing Company, 1992.

14 L. Ishitani, V. Almeida, and W. Meiru. "Masks: Bringing Anonimity and Personalization Together". In *Ninth INFORMS Conference on Information Systems and Technology*, Oct 2004.

15 W. Klosgen. "Anonimization Techniquesfor Knowledge Discovery in Databases". In *Proc. of the First International Conference on Knowledge and Discovery in Data Mining*, pages 186–191, Montreal, Canada, Aug 1995.

16 A. Mayerson and R. Williams. "On the Complexity of Optimal k-Anonimity". In *Proc. of the 23rd ACM-SIGMOD-SIGACT-SIGART Symposium on the Principles of Database Systems*, pages 223–238, 2004.

17 R. Moore. "Controlled Data-swapping Techniques for Masking Public Use Microdata Sets". In *SRD Report RR 96-04, U.S. Bureau of the Census*, 1996.

18 D. Moursmund. "Chebyshev Solution of n+1 Linear Equations in n Unknowns". *Journal of the ACM*, 12:383 – 387, July 1965.

19 K. Muralidhar and R. Sarathy. "Disclosure Risk and Data Utility Characteristics of Data Swapping: A Preliminary Investigation". In *IEEE Security and Privacy*, volume 1, pages 18–23, May 2003.

20 S. Oliveira and O. Zaane. "Privacy Preserving Clustering by Data Transformation". In *Proc. of the 18th Brazilian Symposium on Databases*, pages 304–318, Manaus, Brazil, Oct 2003.

21 S. Oliveira and O. Zaane. "Achieving Privacy Preservation When Sharing Data for Clustering". In *Workshop on Secure Data Management in conjunction with VLDB2004*, Toronto, Canada, Aug 2004. Springer Verlag LNCS 3178.

22 R. Parameswaran and D. Blough. "a robust data obfuscation approach for privacy preserving data mining of clustered data". Technical report, Georgia Institute of Technology, 2005.

23 S. P. Reiss. "Practical Data-swapping The First Steps". In *ACM Transactions on Database Systems*, volume 9, pages 20–37, Mar 1984.

24 R. Rivest, L. Adleman, and M. Dertouzas. "On Data Banks and Privacy Homomorphisms". In R. A. D. et al, editor, *Foundations of Secure Computations*, pages 169–179. Academic Press, 1978.

25 L. Sweeney. "k-Anonymity: A Model for Protecting Privacy". *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.

26 G. Toussaint. "Bibliography on Estimation of Misclassification". *IEEE Transactions on Information Theory*, 20(4):472–479, July 1974.

27 W. Winkler. "Using Simulated Annealing for k-Anonymity". In *Research Report Series, U.S. Census Bureau*, 2002.