

Syracuse University

**SURFACE**

---

Electrical Engineering and Computer Science

College of Engineering and Computer Science

---

2008

## OptRR: Optimizing Randomized Response Schemes for Privacy-Preserving Data Mining

Zhengli Huang  
*Syracuse University*

Wenliang Du  
*Syracuse University, [wedu@ecs.syr.edu](mailto:wedu@ecs.syr.edu)*

Follow this and additional works at: <https://surface.syr.edu/eecs>



Part of the [Computer Sciences Commons](#)

---

### Recommended Citation

Huang, Zhengli and Du, Wenliang, "OptRR: Optimizing Randomized Response Schemes for Privacy-Preserving Data Mining" (2008). *Electrical Engineering and Computer Science*. 128.  
<https://surface.syr.edu/eecs/128>

This Article is brought to you for free and open access by the College of Engineering and Computer Science at SURFACE. It has been accepted for inclusion in Electrical Engineering and Computer Science by an authorized administrator of SURFACE. For more information, please contact [surface@syr.edu](mailto:surface@syr.edu).

# OptRR: Optimizing Randomized Response Schemes for Privacy-Preserving Data Mining

Zhengli Huang and Wenliang Du

Department of Electrical Engineering and Computer Science

Syracuse University, Syracuse, NY 13244-1240, USA

Email: zhuang@ecs.syr.edu, wedu@ecs.syr.edu

**Abstract**—The randomized response (RR) technique is a promising technique to disguise private categorical data in Privacy-Preserving Data Mining (PPDM). Although a number of RR-based methods have been proposed for various data mining computations, no study has systematically compared them to find optimal RR schemes. The difficulty of comparison lies in the fact that to compare two PPDM schemes, one needs to consider two conflicting metrics: privacy and utility. An optimal scheme based on one metric is usually the worst based on the other metric.

In this paper, we first describe a method to quantify privacy and utility. We formulate the quantification as estimate problems, and use estimate theories to derive quantification. We then use an evolutionary multi-objective optimization method to find optimal disguise matrices for the randomized response technique. The experimental results have shown that our scheme has a much better performance than the existing RR schemes.

## I. INTRODUCTION

The increasing concerns over privacy have introduced many challenges in data mining. In most scenarios, data are first collected from individuals or different data owners, and data mining technologies are then applied to these data to extract useful knowledge. However, due to privacy concerns, more and more data owners become unwilling to share their data. This new challenge motivates a field called Privacy-Preserving Data Mining (PPDM).

Randomization is one of the promising approaches in privacy-preserving data mining. With this approach, original data are first disguised before being released to data collectors. Randomization-based disguise protects the private information of individuals, while still allowing the aggregated information (e.g., data distribution and patterns) to be preserved with reasonable accuracy. We use *privacy* to refer to the degree of preservation on individual information; the higher the degree, the worse the privacy. We use *utility* to refer to the degree of preservation on the aggregated information; the higher the degree, the better the utility. Both privacy and utility are critically important for PPDM.

Randomized response (RR) is one of the randomization techniques that have been extensively studied in the PPDM community. The RR technique for binary data was first introduced by Warner [1] in 1965; it can be extended to general categorical data [2]. In the RR technique, each sensitive data item is probabilistically replaced by another value. The probability of replacing a value  $c_i$  with another value  $c_j$  is captured by a transformation matrix  $M$ ;  $M$  is called an *RR matrix* in this paper.

Privacy and utility of the randomized response technique depend on RR matrices. However, although a number of RR-based schemes have been proposed for various data mining computations [3]–[6], not much effort has been put on the selection of  $M$ . It is not even clear how to compare different RR matrices, much less finding an optimal  $M$  for the randomized response technique. The difficulty is caused by the conflicting nature of privacy and utility; it is not fair to compare different RR matrices only based on privacy or utility. Both aspects need to be considered.

We formulate the above problem as a multi-objective optimization problem. Unlike a single-objective optimization problem, we have two objectives: privacy and utility. A solution is optimal if it is not *dominated* by any other solutions; namely no other solutions are better than an optimal solution. Such optimal solutions are not unique; we can have a set of optimal solutions, each corresponding to a different RR matrix. We call these matrices optimal RR matrices. *The objective of this paper is to find an optimal set of RR matrices.* Moreover, to satisfy diversified requirements (application dependent) on privacy (or utility), we would like the optimal set to *cover a broad range of privacy (or utility).*

To find optimal solutions, we need to first quantify privacy and utility. We consider the quantification as estimation problems. For privacy, we quantify how accurately adversaries can estimate the original data (individual data items) from the disguised data. We prove that the best adversaries can do is to use Bayes estimate. We then use the Bayes estimate to quantify privacy. For utility, we quantify how accurately we can estimate the distribution (aggregate information) of the original data, also from the disguised data. There are a number of ways to measure such an estimate. We choose to use the Mean Square Error (MSE), which is the same as the variance of the estimate in randomized response.

Based on our metrics, we can search for optimal RR matrices. The brute-force method is infeasible due to the large search space even for a categorical attribute with as few as 10 categories. We use an improved Evolutionary Multi-Objective Optimization (EMOO) method to find optimal RR matrices. Our solutions have shown a significant improvement over the RR matrices used by existing work.

The rest of this paper is organized as follows. In Section II, we review the related work. We then formulate our problem in Section III. After describing how to quantify privacy and

utility in Section IV, we describe our solution in Section V and our evaluation results in Section VI. Finally, we draw the conclusions and outline the future work in Section VII.

## II. RELATED WORK

Randomization is one of the approaches in achieving privacy-preserving data mining. Two classes of randomization methods have been extensively studied in the literature: Random Perturbation and Randomized Response.

Random perturbation is primarily used for quantitative data, first explored by Agrawal and Srikant [7]. Agrawal and Aggarwal later improve the distribution reconstruction method using Expectation Maximization (EM) algorithm [8]. Kargupta et al. [9] and Huang et al. [10] point out that arbitrary randomization can reveal significant amount of information under certain conditions, especially when the attributes of data are highly correlated.

Randomized Response (RR) is primarily used for categorical data. Its basic idea was proposed by Warner [1]. RR disguises data by probabilistically changing the value of a sensitive attribute to another value. Similar to random perturbation, the distribution of the original data can also be reconstructed using the disguised data. Based on the RR technique, Rizvi and Haritsa propose a scheme to mine association rules from disguised data [3]; Evfimievski et al. propose a different approach to conduct privacy-preserving association rule mining [4]; Du and Zhan propose a scheme to conduct privacy-preserving decision tree building [5]. Agrawal et al. study how to optimize the efficiency of the distribution reconstruction process [6].

There is a lack of effort in finding the optimal randomized response matrices. Agrawal and Haritsa took an initial step towards this direction [11]. They discuss how to find optimal RR matrices. However, the proposed scheme only tries to find optimal RR matrices among symmetric matrices. Moreover, when comparing different RR matrices, it only chooses accuracy as the comparison criterion. Our work is different in two aspects: first, we search among both symmetric matrices and asymmetric matrices; second, we use both accuracy and privacy as our criteria when comparing two different RR matrices. We formulate the problem as a multi-objective optimization problem. More importantly, our solution achieves better results than that in [11].

## III. RANDOMIZED RESPONSE (RR)

The Randomized Response (RR) technique is mainly used for randomizing categorical data. In this paper, we use  $C$  to represent the domain of the data attribute that needs to be disguised. This attribute domain contains  $n$  different values (or categories), denoted by  $\{c_1, c_2, \dots, c_n\}$ . Note that the randomized response technique can be extended to multi-dimensional, i.e., the techniques are applied to several dimensions altogether. However, in this paper, for the sake of simplicity, we focus on one-dimensional RR; namely, we apply RR for each dimension independently.

### A. The General RR Technique

The basic idea of the randomized response technique is to replace each  $c_i$  in the original data set with another value  $c_j$  with a certain probability. We use  $\theta_{j,i}$  to denote the probability that a category  $c_i$  is randomized to  $c_j$ , where  $i, j = 1, \dots, n$ . We use  $P^*(c_i)$  to represent the probability of  $c_i$  in the disguised data, and  $P(c_i)$  to represent the probability of  $c_i$  in the original data.

Let  $\vec{P}^* = (P^*(c_1), P^*(c_2), \dots, P^*(c_n))^T$ , and  $\vec{P} = (P(c_1), P(c_2), \dots, P(c_n))^T$ . Let  $M$  be the following matrix:

$$M = \begin{pmatrix} \theta_{1,1} & \dots & \theta_{1,n} \\ \vdots & \ddots & \vdots \\ \theta_{n,1} & \dots & \theta_{n,n} \end{pmatrix}.$$

We can easily derive the following equation:

$$\vec{P}^* = M\vec{P}. \quad (1)$$

In the above equation,  $\vec{P}$ , the distribution in the original data set, is what we are trying to find out;  $\vec{P}^*$ , the distribution of the disguised data, can be estimated using the frequency of each category in the disguised data. Let  $N_i$  ( $i = 1, \dots, n$ ) be the number of  $c_i$ 's in the disguised data and  $N$  be the total number of records in the data set. We have the MLE (Maximum Likelihood Estimate) of  $P^*$ , which is  $\widehat{\vec{P}}^* = (\frac{N_1}{N}, \dots, \frac{N_n}{N})^T$ . If  $M$  is invertible, we substitute  $\widehat{\vec{P}}^*$  for  $\vec{P}^*$  in Equation (1), solve the equation and get the estimate of  $\vec{P}$ . The estimate is an unbiased MLE. These are ensured by the following theorem<sup>1</sup>.

*Theorem 1:* The MLE (Maximum Likelihood Estimate) of  $\vec{P}^*$  is  $\widehat{\vec{P}}^* = (\frac{N_1}{N}, \dots, \frac{N_n}{N})^T$ . If  $M$  is invertible, the unbiased MLE of  $\vec{P}$  is

$$\widehat{\vec{P}} = M^{-1}\widehat{\vec{P}}^*. \quad (2)$$

Intuitively speaking, the above theorem indicates that when the number of records in the disguised data is large enough, the estimated distribution derived from Equation (2) is very close to the original distribution.

The above estimation approach uses matrix inversion and is thus called the *inversion approach*. Agrawal et al. proposed an *iterative approach* to estimate the original distribution [2]: let the initial distribution  $P(X = c_i)^0$  for  $i = 1, \dots, n$  be some non-negative values but  $\sum_{i=1}^n P(X = c_i)^0 = 1$  (at step 0) and run the following iterative process (at step  $k$ ):

$$P(X = c_i)^{k+1} = \sum_{i=1}^n P^*(X = c_i) \frac{\theta_{i,j} P(X = c_j)^k}{\sum_{j=1}^n \theta_{i,j} P(X = c_j)^k}. \quad (3)$$

The above iteration stops until two consecutive  $P(X)$  are close enough. If it stops after step  $s$ , the  $P(X = c_i)^{s+1}$  for  $i = 1, \dots, n$  is the estimated distribution.

We decide to use the inversion approach in our optimization algorithm for the following reason: the inversion approach give us a close-form formula to quantify the estimation error, but for the iterative approach, getting the errors takes a much

<sup>1</sup>The proofs of all theorems are omitted due to the page limitation.

longer time, because we have to finish the iterative process before we can calculate the errors. Our optimization method is based on genetic algorithms, and optimal solutions surface after many generations, so being able to compute error fast at each generation is essential.

### B. Different RR Matrices

Several specific RR matrices have been proposed in the literature. The first one is the Warner scheme [1], which was originally developed for binary data. It can be easily extended for categorical data. It assigns  $p$  to all the diagonal elements of the matrix and  $\frac{1-p}{n-1}$  to all the non-diagonal elements. Another type of RR matrix is proposed by Agrawal et al. [2]. It is called Uniform Perturbation (UP), in which each category value is retained with a probability  $q$  and replaced, with a probability  $1 - q$ , by a value selected from a uniform distribution on the domain of category values. The third scheme is called FRAPP, proposed by Agrawal and Haritsa [11]. Its diagonal elements are chosen according to the privacy requirement. The matrices of all these three schemes are described in the following:

$$M \text{ in the Warner scheme} = \begin{pmatrix} p & \frac{1-p}{n-1} & \cdots \\ \frac{1-p}{n-1} & p & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix},$$

$$M \text{ in UP} = \begin{pmatrix} q + \frac{1-q}{n} & \frac{1-q}{n} & \cdots \\ \frac{1-q}{n} & q + \frac{1-q}{n} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix},$$

$$M \text{ in FRAPP} = \frac{1}{\lambda + n - 1} * \begin{pmatrix} \lambda & 1 & \cdots \\ 1 & \lambda & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}.$$

With all these different matrices, it is important to know which one is better. More importantly, we would like to know whether any of these matrices is optimal, and if not, how we can find optimal RR matrices. The main objective of this paper is to answer these questions.

### C. Finding Optimal RR Matrices

The performances of privacy-preserving data mining schemes are usually measured using two metrics: *privacy* and *utility*. Privacy measures how much of *individual* information is protected, while utility measures how much of *aggregated* information (e.g. data distribution) is preserved. Achieving high privacy and utility seems to be two conflicting goals: if one wants to increase utility, privacy usually gets worse, and vice versa. Therefore, it is unreasonable just to use one criterion to compare different privacy-preserving data mining schemes.

The following example demonstrates the conflicting nature of privacy and utility. Let  $M_1$  and  $M_2$  be two different RR matrices for a data attribute with three categories:

$$M_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad M_2 = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{pmatrix}.$$

$M_1$  is the Identity matrix; it does not disguise data at all, i.e., neither individual information nor aggregate information is lost. Therefore the privacy is the worst, but the utility is the best. On the other hand,  $M_2$  changes any original value into one of the three possible values with  $1/3$  probability each. This basically turns any information into uniformly distributed random data, regardless of what individual and aggregation information is. Therefore, both individual information and aggregate information of the original data are lost, which means that privacy is perfectly preserved, but the data become totally worthless. This example indicates that if we simply use one criterion (either privacy or utility) to compare randomized response schemes,  $M_1$  and  $M_2$  are optimal for their respective criteria. They are obviously not what we are looking for.

Let  $P(M)$  represent the privacy of an RR matrix  $M$  and  $U(M)$  represent the utility of  $M$ . We define optimality as the following:

*Definition 3.1: (Optimal RR Matrix)* An RR matrix  $M$  is optimal if for any matrix  $M'$ , one of the following conditions holds:

- 1)  $P(M) = P(M')$  and  $U(M) = U(M')$ ;
- 2) at least one of  $P(M)$  and  $U(M)$  is better than that of  $M'$ .

Optimal RR matrices are not unique, and actually, there should be many optimal RR matrices. For example, both  $M_1$  and  $M_2$  in the above example are optimal. For these two optimal solutions, it is obvious that they are useless for privacy-preserving data mining. However, for other RR matrices, whether they are useful or not is less obvious. In fact, the usefulness of an optimal solution depends on users' requirements on privacy and utility. It is more desirable to provide a range of optimal matrices for users to choose, rather than providing a single optimal matrix.

Even if the users need a single optimal RR matrix for certain privacy or utility, they still need to find a set of optimal RR matrices. The reason is that it is difficult to decide all those RR matrices with fixed privacy (or utility). Thus, we can not search the set of RR matrices with fixed privacy (or utility) for an optimal RR matrix with the best utility (or privacy). Instead, we have to search the set of RR matrices covering the whole utility (privacy) range for a set of optimal RR matrices and then pick the optimal RR matrix with certain privacy (utility).

Therefore, we formulate the following problem: *Given a data set  $D$ , find a set of optimal RR matrices that provide a diversified coverage of privacy (or utility)*. We present our solution to this problem in the rest of this paper.

It should be noted that the existing schemes described in Section III-B do provide a diversified coverage (although not necessarily optimal). For example, for the Warner scheme, when  $p = 1$ , we get the Identity matrix (the same as  $M_1$ ); when  $p = \frac{1}{3}$ , we get  $M_2$ . By changing the variables ( $p$ ,  $q$ , and  $\lambda$ ), each of the three schemes produces a set of solutions with different privacy and utility. A closer look at these solution sets reveals the following theorem:

*Theorem 2:* The solution sets for the Warner, UP, and FRAPP schemes are identical.

#### IV. QUANTIFYING PRIVACY AND UTILITY

For each attribute of a data set, we use  $X$  to represent the attribute variable of the original data; and we use  $Y$  to represent the attribute variable of the disguised data. We use lower case  $x$  and  $y$  to represent an instance of these variables. Since we only consider one-dimensional RR schemes (i.e., we only consider one attribute at a time), we can consider that our entire data set consists of one single attribute. The values of this data set are instances of the attribute variables. We use  $X_s = \{x_1, \dots, x_N\}$  and  $Y_s = \{y_1, \dots, y_N\}$  to represent the original and disguised data sets, respectively, where  $N$  is the number of records in each data set. We use  $C$  to represent the domains of  $X$  and  $Y$ , where  $C$  consists of  $n$  categorical values, represented by  $c_1, \dots, c_n$ .

As we have mentioned before, privacy measures how much of individual information can be *estimated* from the disguised data, and utility measures how much of the original data distribution (aggregate information) can be *estimated* from the disguised data. We base our privacy and utility definition on estimation theories.

##### A. Privacy

Privacy quantification can be considered as the following estimation problem:

*Problem 1:* Given a disguised data set  $Y_s = \{y_1, \dots, y_N\}$  and certain prior knowledge of the original data  $X$ , adversaries would like to estimate the original value  $x_i$  from  $y_i$ , for  $i = 1, \dots, N$ . How accurate can their estimates be?

The less accurate their estimates are, the higher privacy is. Therefore, to quantify privacy, we just need to quantify the accuracy of the estimates. We define an accuracy function  $G$  to represent the accuracy score of an estimate (denoted by  $\hat{x}_i$ ) against the actual value  $x_i$ .  $G$  can be defined based on the nature of the data, as well as on the privacy requirements of applications. With this function, the average accuracy score of all the estimates can be computed in the following:

$$A = \frac{1}{N} \sum_{i=1}^N G(\hat{x}_i, x_i). \quad (4)$$

We will derive the optimal value for  $A$ , which represents the best estimates that can be achieved by adversaries. We will then use this optimal result to quantify privacy.

For the simplicity of explanation, we first consider consistent estimates defined below. Later we extend to inconsistent estimates.

*Definition 4.1:* (Consistent Estimation) We call an estimation is *consistent* if for any two observed data  $Y = y_i$  and  $Y = y_j$ ,  $y_i = y_j$ , their respective estimates  $\hat{x}_i$  and  $\hat{x}_j$  are also the same.

When estimates are consistent, for all the records that share the same disguised value  $Y$ , the estimates are always the same. We use  $\widehat{X}_Y$  to represent this estimate. However, according to the randomized response technique, the same value  $Y$  might be the disguised results of different values from  $C = \{c_1, \dots, c_n\}$ . To maximize accuracy, given a specific

$Y$ , adversaries would like to find an (consistent) estimate  $\widehat{X}_Y$ , such that the expected accuracy score is maximized. The expected accuracy score can be computed using the following formula:

$$E_X[G(\widehat{X}_Y, X) | Y] = \sum_{X \in C} G(\widehat{X}_Y, X) \cdot P(X | Y). \quad (5)$$

This is actually the *Bayes Estimate*, the theory of which not only provides optimal estimates for a variety of accuracy functions  $G$ , but also provides a methodology to derive optimal estimates for an arbitrary  $G$ . In this paper, we study one specific accuracy function:

$$G(\widehat{X}, X) = \begin{cases} 1, & \text{when } \widehat{X} = X; \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

The intuitive meaning of the above function says that when an estimate is correct, the score is 1; otherwise, the score is 0.

*Theorem 3:* For the accuracy function  $G$  defined in Equation (6), the optimal estimate  $\widehat{X}_Y$  for a given  $Y$  is the MAP (maximum a posteriori) estimate, i.e.,

$$\widehat{X}_Y = \operatorname{argmax}_{X \in C} P(X | Y) \quad (7)$$

Theorem 3 indicates that the MAP estimate is the best that adversaries can achieve when their estimation is consistent. However, adversaries do not need to follow the consistency constraints. For two data records  $i$  and  $j$  with the same disguised value  $y$ , the adversaries can use  $x_i$  to estimate the original value for record  $i$ , and then use a different value  $x_j$  to estimate that for record  $j$ . We call such an estimation *inconsistent estimation*. The question is whether the adversaries can achieve a better expected accuracy value with inconsistent estimation. The following theorem rules out this possibility:

*Theorem 4:* Given the same prior information about  $X$ , the inconsistent estimation is no better than the MAP estimate.

From Theorem 3 and Theorem 4, we know that the MAP estimate is optimal. In other words, MAP estimate gives an upper bound on what adversaries can achieve. Therefore, MAP estimate can be used to quantify the privacy.

In Equation (5), the expected accuracy value is computed for a particular  $Y$  over all  $X \in C$ . To consider all  $Y \in C$  from the disguised data set, we compute another expected value, but this time, over  $Y$ . This is the expected accuracy value for the entire disguised data. Namely, the value  $A$  defined in Equation (4) can be computed using the following formula, rather than using the sample means:

$$\begin{aligned} A &= E_Y\{E_X[G(\widehat{X}_Y, X) | Y]\} \\ &= \sum_{Y \in C} P(\widehat{X}_Y | Y) \cdot P(Y), \end{aligned}$$

where  $\widehat{X}_Y$  is the MAP estimate for an given  $Y$ . Using Bayes rules, we can further simplify the above formula to the following:

$$A = \sum_{Y \in C} P(Y | \widehat{X}_Y) \cdot P(\widehat{X}_Y).$$

The value of  $A$  represents the average estimation accuracy from adversary's perspective; the larger  $A$  is, the worse for privacy. Therefore, we define the privacy as  $1 - A$ , i.e.,

$$\text{Privacy} = 1 - \sum_{Y \in \mathcal{C}} P(Y | \widehat{X}_Y) \cdot P(\widehat{X}_Y). \quad (8)$$

It should be noted that the above equation is an average metric; in addition to the average privacy, data owners are often concerned about worst-case scenarios. To deal with this concern, we set an upper bound for the estimation accuracy of each data value as the following:

$$P(\widehat{X}_Y | Y) \leq \delta \text{ for } 0 < \delta < 1. \quad (9)$$

The smaller the upper bound is, the harder it takes the adversaries to correctly estimate the original data. However, this upper bound cannot be set to an arbitrary small value. For example, if a value  $c_q$  appears in the original data set with the probability of 99%, no matter how the data set is randomized, the adversaries can always guess the original data values to be  $c_q$ ; therefore, the probability of being correct is 99%. This means that the highest estimation accuracy by the adversaries cannot be less than 99%. Actually, the upper bound  $\delta$  ( $\max_Y P(\widehat{X}_Y | Y)$ ) is always larger than or equal to the prior distribution of  $X$ :

*Theorem 5:* If  $\widehat{X}_Y$  is MAP estimate,  $\max_Y P(\widehat{X}_Y | Y) \geq \max_X \{P(X)\}$ .

### B. Utility

In privacy-preserving data mining, data utility quantifies how much of data distribution can be reconstructed. We consider this quantification also as an estimation problem. However, unlike the privacy quantification, we do not try to estimate individual values of the original data; instead, we estimate aggregate information (i.e., distribution) of the original data. The estimation method used by the randomized response technique is already given by Equation (2).

Let  $P(X)$  represent the distribution of the original data  $X$ , and let  $\widehat{P}(X)$  represent the estimated distribution of  $X$  based on the disguised data. From Theorem 1, we know that the estimation method based on Equation (2) is an unbiased MLE estimate. Namely, the mean of the estimate  $\widehat{P}(X)$  is the same as the original distribution  $P(X)$ . However, each specific estimate  $\widehat{P}(X)$  deviates from  $P(X)$  to a certain degree. The closer  $\widehat{P}(X)$  is to  $P(X)$ , the better the utility. Therefore, we can use the *Mean Squared Errors (MSE)* between  $\widehat{P}(X)$  and  $P(X)$  to quantify utility. MSE can be computed in the following:

$$\text{MSE} = E(\widehat{P}(X) - P(X))^2.$$

Because the estimate is unbiased, the mean of  $\widehat{P}(X)$  is the same as the original  $P(X)$ . Replacing  $P(X)$  of the above equation with the mean of  $\widehat{P}(X)$ , we basically get the variance of  $\widehat{P}(X)$ . Therefore, by applying the variance operation on the both sides of Equation (2), we can get the following theorem:

*Theorem 6:* The Mean Squared Error (MSE) between  $P(\widehat{X} = c_k)$  and  $P(X = c_k)$  can be computed by the following:

$$\begin{aligned} \text{MSE}(X = c_k) &= E(P(\widehat{X} = c_k) - P(X = c_k))^2 \\ &= \sum_{i=1}^n \beta_{k,i}^2 \text{Var}\left(\frac{N_i}{N}\right) + \sum_{i=1, j=1, i \neq j}^n 2 \cdot \beta_{k,i} \beta_{k,j} \text{Cov}\left(\frac{N_i}{N}, \frac{N_j}{N}\right), \end{aligned}$$

where  $\beta_{g,h}$  is the  $g$ -th row and  $h$ -th column element of the  $M^{-1}$ , and

$$\text{Var}\left(\frac{N_i}{N}\right) = \frac{1}{N} \cdot P(Y = c_i)(1 - P(Y = c_i)),$$

$$\text{Cov}\left(\frac{N_i}{N}, \frac{N_j}{N}\right) = -\frac{1}{N} \cdot P(Y = c_i) \cdot P(Y = c_j).$$

We measure the utility using the average MSE for all  $c_1, \dots, c_n$ , i.e.,

$$\text{Utility} = \frac{1}{n} \sum_{k=1}^n E(P(\widehat{X} = c_k) - P(X = c_k))^2. \quad (10)$$

It should be noted that since utility is defined based on mean squared errors, the larger its value, the worse the utility.

## V. EVOLUTIONARY MULTI-OBJECTIVE OPTIMIZATION

Equipped with the privacy and utility quantification, we search for the optimal RR matrices for a given data set. The most straightforward search method is brute-force; namely we exhaustively compare all possible RR matrices. Let us see how expensive it is: assume that a single-attribute data set has  $n$  category values, so there are  $n^2$  elements in an RR matrix. These elements have to satisfy two constraints: (1) each element is between 0 and 1, and (2) the sum of the elements in each column is 1. If the range is discretized to  $0, 1/d, 2/d, \dots, 1$  for a positive integer  $d$ , the number of combinations of element values in an RR matrix is given in the following fact:

*Fact 1:* If each element of an RR matrix can be any of values in  $\{0, 1/d, 2/d, \dots, 1\}$ , the total number of combinations for the RR matrix is  $\binom{d+n-1}{d}^n$ .

The number of combinations can be very large even for an attribute with a small number of categories. For example, if  $n = 10$  and  $d = 100$ , the number of combinations can be  $1.98 \times 10^{126}$ , which is infeasible to search. To reduce the search space, we resort to genetic algorithms.

Genetic algorithms define a scalar fitness function for each solution; the ones with the best fitness value are optimal. This approach is quite natural for a single-objective optimization problem, but has difficulty dealing with multiple objectives, especially when these objectives are conflicting, such as utility and privacy. A simple solution to extend a genetic algorithm to a multiple-objective problem is to combine all the objectives into a single fitness function. However, there are a number of problems with this approach: first, it is difficult to find a function to combine these different objectives that reflect the nature of the problem; second, it has been shown that the

approach cannot generate proper members of the optimal set if the optimal front is concave [12].

Evolutionary Multi-Objective Optimization (EMOO) algorithms have been developed to deal with multiple objectives. A comparison study has shown that SPEA2 outperforms other recent approaches in many well-known problems [13]. Therefore, we choose to use SPEA2. We customized it for our search problem; moreover, we slightly modified the algorithm to improve the performance for our specific problem.

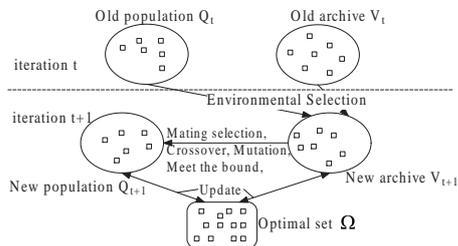


Fig. 1. Our SPEA2-based algorithm

### A. Outline of Our SPEA2-based Algorithm

Similar to the evolution in the nature, SPEA2 iteratively searches the optimal RR matrices using two sets of RR matrices: *archive* and *population*. *Archive* is the initial set for a generation (generated from the previous generation); the RR matrices in the archive set will evolve during the current generation via crossover and mutation. The evolution results are put in a set called *population*. The archive set and the new population set will be used to generate a new archive for the next generation.

To keep computation manageable, the sizes of population and archive are fixed, so computation cannot grow out of control. As a consequence, some good RR matrices will be thrown away during the evolution when the population and archive sets become too “crowded”. To deal with this problem, we introduce a new set called *optimal set*, which stores those good RR matrices that are thrown away from each iteration. This set does not participate in the costly evolution process; instead, it is only used at the end of an iteration, in which, the archive set (or the population set) and optimal set are used to update each other. This last step is quite efficient, compared to the evolution process, so the size of the optimal set can be much larger than the sizes of population and archive. As a result, many good RR matrices can still be kept. The process of our SPEA2-based algorithm is illustrated in Figure 1.

We use  $Q_t$  and  $V_t$  to represent the population and the archive of the  $t$ -th iteration or generation, respectively. We also use  $\Omega$  to represent the optimal set. The initial population  $Q_0$  consists of a collection of randomly-generated RR matrices. The initial archive  $A_0$  and optimal set  $\Omega$  are set to be empty. The iteration step is described in the following:

#### Algorithm Optimization for RR Matrices

**Input:**  $N_Q$ : population size,  
 $N_V$ : archive size,

$L$ : maximum number of iterations,

$N_\Omega$ : size of the optimal set.

**Output:** the optimal set  $\Omega$

**Initialization:** Generate an initial population  $Q_0$ ,  
and set  $V_0 = \Omega = \emptyset$ .

**Repeat:**

1. **Fitness assignment:** Calculate fitness values of RR matrices in  $Q_t$  and  $V_t$
2. **Environmental selection:** Select RR matrices in  $Q_t$  and  $V_t$  and copy them to  $V_{t+1}$ .
3. **Mating selection:** Select promising RR matrices from  $V_{t+1}$  for later evolution.
4. **Crossover and mutation:** Apply crossover and mutation operators to those promising RR matrices and store the resulting RR matrices in  $Q_{t+1}$ .
5. **Meeting the bound:** Make each RR matrix in the resulting population  $Q_{t+1}$  meet the bound  $\max\{P(X|Y)\} \leq \delta$ .
6. **Updating the three sets:** Update the resulting population  $Q_{t+1}$ , archive  $V_{t+1}$  and the optimal set  $\Omega$ . Increment the generation counter ( $t = t + 1$ ).
7. **Termination:** If the stopping criterion is satisfied, stop and output the optimal set  $\Omega$ ; otherwise, go to the next iteration.

### B. Fitness Assignment

The quality of an RR matrix in a set can be quantified using a fitness value, which is based on the *dominance* relationship defined below:

*Definition 5.1:* (Dominance) An RR matrix  $M_i$  dominates another  $M_j$  if none of the properties (privacy and utility) of  $M_i$  is worse than the corresponding properties of  $M_j$ ; moreover, at least one property of  $M_i$  is better than the corresponding property of  $M_j$ . We use  $M_i \succ M_j$  to denote that  $M_i$  dominates  $M_j$ .

When an RR matrix is dominated by another RR matrix, the former has a worse fitness value (higher quality) than the latter. Accordingly, if an RR matrix is dominated by more RR matrices, it also has a worse fitness value. When two RR matrices are dominated by the same number of RR matrices, according to SPEA2, the RR matrices dominating the two RR matrices are further considered to discriminate the qualities of the two. Among the two RR matrices, if one is dominated by some RR matrices, which dominate more RR matrices, and the other is dominated by some RR matrices, which dominate fewer RR matrices, the former has a worse fitness value than the latter.

Based on dominance relationship, SPEA2 defines a *raw fitness* for each RR matrix in the archive and population sets. The calculation of raw fitness consists of two steps: first, each  $M_i$  in the archive  $V_t$  and the population  $Q_t$  is assigned a strength value  $S(M_i)$ , representing the number of RR matrices dominated by  $M_i$ , i.e.,

$$S(M_i) = |\{M_j | M_j \in (Q_t + V_t) \text{ and } M_i \succ M_j\}|,$$

where  $|\cdot|$  is the cardinality of a set,  $+$  means multiset union. Second, based on  $S$  values, the raw fitness  $F'(i)$  of an  $M_i$  is

the sum of the strength of the RR matrices that dominate  $M_i$ :

$$F'(M_i) = \sum_{M_j \in Q_t + V_t, M_j \succ M_i} S(M_j).$$

When two RR matrices have the same raw fitness values, another important factor, *diversity*, is used to further discriminate them. The use of diversity is based on the following observation: users' requirements on privacy (or utility) are often diversified, so we would like the set of solutions to cover a wide range of privacy (or utility). In other words, in the search algorithm, we do not like to always pick solutions from a dense area. SPEA2 measures the density of each solution. The density for an RR matrix is a (decreasing) function of the distance to the  $k$ -th nearest data point, denoted as  $\sigma_i^k$  ( $k$  is usually set to 1 in practice):

$$d(M_i) = \frac{1}{\sigma_i^k + 2},$$

where 2 is added to ensure that the denominator value is greater than zero and that  $d(M_i) < 1$  (this ensures that density only makes a difference when two RR matrices have the same raw fitness value, because the difference of two different fitness values is always greater than or equal to 1).

Figure 2 illustrates the use of density using an example. In the figure, three RR matrices  $M_3$ ,  $M_4$  and  $M_5$  have the same raw fitness 0 (they are not dominated by any solution). Because  $M_3$  has a shorter distance to its nearest neighbor than  $M_4$  and  $M_5$ , it has a higher density value; therefore,  $M_3$  has a worse fitness value than  $M_4$  and  $M_5$ .

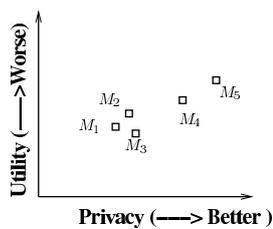


Fig. 2. Compare the quality of  $M_3$ ,  $M_4$ ,  $M_5$

Based on dominance and diversity, the fitness  $F(M_i)$  of an RR matrix  $M_i$  is defined as the sum of density  $d(M_i)$  and raw fitness  $F'(M_i)$ :

$$F(M_i) = F'(M_i) + d(M_i). \quad (11)$$

From the above computations, the lower fitness an RR matrix has, the higher quality it has. The running time for assigning fitness to all RR matrices is dominated by the density calculation  $O((N_Q + N_V)^2 \log(N_Q + N_V))$ .

### C. Environmental Selection

After each RR matrix is assigned a fitness value, high-quality RR matrices are selected from the old population and archive sets, and they are stored in a new archive of the next generation. This process is called *environmental selection*. The process consists of two steps: first, all non-dominated RR

matrices (the fitness values of which are less than 1) are copied from the archive and population sets to the new archive  $V_{t+1}$ :

$$V_{t+1} = \{M_i | M_i \in (Q_t + V_t) \text{ and } F(M_i) < 1\} \quad (12)$$

Second, if  $|V_{t+1}|$  (the size of the new archive) is equal to  $N_V$  (the limit on archive size), the environmental selection step is finished; otherwise, there can be two cases:

- 1) If  $|V_{t+1}| < N_V$  (i.e., we do not have enough non-dominated RR matrices),  $N_V - |V_{t+1}|$  dominated RR matrices with the best fitness values are copied from the archive and population sets to the new archive.
- 2) If  $|V_{t+1}| > N_V$  (i.e., we have too many non-dominated RR matrices), some non-dominated RR matrices will be removed iteratively from the new archive  $V_{t+1}$  until  $|V_{t+1}| = N_V$ . At each iteration, to maintain diversity, an RR matrix is removed if, among all the RR matrices in  $V_{t+1}$ , it has the shortest distance to its nearest neighbor. If several RR matrices have the same shortest distances, the distances to their second nearest neighbor are considered, and so on.

The worst run-time complexity of the environmental selection is dominated by the removal process, which is  $O((N_Q + N_V)^3)$ .

### D. Mating Selection

After the new archive is constructed, RR matrices are selected from the archive to fill a pool called *mating pool*. At each selection step, two RR matrices are randomly picked from the new archive, the one with the better (lower) fitness value wins and is copied to the mating pool. This procedure, called *binary tournament selection*, is repeated until the mating pool is full. If the size of mating pool is  $N_M$ , the run-time complexity of mating selection is  $O(N_M)$ . All the matrices in the mating pool are called *parents*.

### E. Crossover

To mimic the mating in evolution, a crossover operator is applied to the mating pool. Before we do crossover, we randomly pair the RR matrices in the mating pool; each pair is then used as parents to produce two new RR matrices (called *children*) using crossover, i.e., the parents exchange some columns of their matrices. Since the sum of elements in each column is 1, not every position can be chosen as the crossover position. Instead, we randomly pick a line delimiting two neighboring columns. all elements on the right side of the line are swapped (between the two parent RR matrices). The two resulting RR matrices become the children. For example, in Figure 3, after the crossover, the two parents  $M_1$  and  $M_2$  produce two children  $M_1'$  and  $M_2'$ . The run-time complexity is  $O(n^2)$ , where  $n$  is the number of columns (or rows) in an RR matrix.

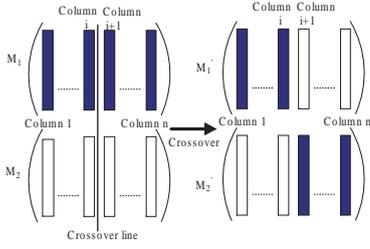


Fig. 3. Crossover operation on 2 parent RR matrices

### F. Mutation

After the crossover, we apply the mutation operator to modify RR matrices. Mutation changes small parts of an RR matrix according to a given mutation rate. In general, mutation only changes an arbitrary bit in a genetic sequence (RR matrix) with a small probability; however, to ensure that a mutated matrix is still an RR matrix, our mutation operator changes a column, rather than a bit. We first randomly select a column, and then add (or subtract) a small random positive value ( $< 1$ ) to an arbitrary element (denoted by  $\theta_{i,j}$ ) in the selected column. Because all elements in an RR matrix should be non-negative, the added value should be less than or equal to  $1 - \theta_{i,j}$  while the subtracted value should be less than or equal to  $\theta_{i,j}$ . To satisfy the RR matrix constraint (i.e., the sum of each column in any RR matrix is 1), certain values should be subtracted from or added to other elements of the same column.

Because we consider that a characteristic of a good RR matrix is represented by the correlations among the elements of a column in the RR matrix, we do not want the mutation to destroy the characteristic. To maintain it, we change all other elements in the column in the following way: if the first changed element is obtained by addition, the subtracted values from all other elements are proportional to the corresponding element values. If it is obtained by subtraction, the added values to all other elements are proportional to 1 minus the corresponding element values. The run-time complexity of the mutation is  $O(n)$ .

### G. Meeting the Privacy Bound

After the mutation step, we want to make sure that the largest posterior probability  $P(X|Y)$  is less than the upper bound  $\delta$ . What we do is similar to that in the mutation. We decrease the elements which make the  $P(X|Y)$  too large in a column and increase the other elements in the same column. The increased value is proportional to the maximal value added to the element to achieve the bound  $\delta$ . For example, if the current element  $\theta_{i,j}$  can be changed to  $\theta'_{i,j}$  to make  $P(X = c_j|Y = c_i) = \delta$ , the added value to  $\theta_{i,j}$  is proportional to the difference  $\theta'_{i,j} - \theta_{i,j}$ . The updating cost for one column is  $O(n)$ .

### H. Updating the Three Sets

Because the run-time complexity of the environmental selection is cubic of the size of the population and the archive, to prevent computation costs from getting too high, the sizes

of the population and archive sets have to be limited. Consequently, when the population and archive become “crowded”, many good RR matrices will be discarded during the evolution. To deal with the problem, we use  $\Omega$  to store those good RR matrices that are discarded in each iteration.  $\Omega$  does not participate in the costly environmental selection process; it is involved only in the updating process depicted below.

In  $\Omega$ , those good RR matrices are indexed by their privacy values. For example, if the size of  $\Omega$  is 1000, the good RR matrix with privacy value 0.1523 can be stored in the 152th element. According to the indexing, after the mutation operation, each RR matrix in  $Q_{t+1}$  and  $V_{t+1}$  is compared with its corresponding RR matrix in  $\Omega$ . The RR matrices with better utility values in one set are used to update the corresponding RR matrices in the other set. The updating takes  $O(N_Q + N_V)$  time.

Because the run-time complexity does not depend on the size of  $\Omega$ , we can set  $\Omega$  as large as the memory allows, such that as many good RR matrices as possible can be stored. After updating  $\Omega$ ,  $Q_{t+1}$  or  $V_{t+1}$ , we can always use the good RR matrices to evolve (selecting, mating, crossover and mutation). When  $\Omega$  is output as the set of optimal RR matrices, the users can get more detailed spectrum of the optimal RR matrices for making choices.

Using  $\Omega$  to store a wider spectrum of the optimal RR matrices can only be employed for problems with a few objectives, because the number of optimal RR matrices will increase exponentially with the increment of the number of objectives, i.e., the size of the needed memory to store the optimal matrices will increase exponentially.

### I. Termination

There are several ways to decide the termination criterion. One way is to set a limit on the number of generations that RR matrices can evolve. Another way is to specify a threshold for the number of consecutive generations that do not modify  $\Omega$  (i.e., the evolution makes no progress during these generations). When the criterion is met, the optimal set  $\Omega$  is output as the set of optimal RR matrices.

Our scheme is based on SPEA2 but differs from it in three aspects. First, our scheme adopts the optimal set  $\Omega$  which can keep much more optimal RR matrices than SPEA2 without increasing the computation complexity. Second, our crossover method is specific for RR matrices because of the constraints of our problem. Third, we use a novel mutation approach by which the correlations among most elements could be maintained when an RR matrix is mutated.

## VI. EXPERIMENTS

### A. Using Pareto Fronts

For multi-objective optimization, the set of optimal solutions is called the *Pareto optimal set* [14]. The Pareto optimal set is often plotted in the objective space (e.g. in our case, the space consists of utility and privacy), and the plot is called the *Pareto front*. By plotting the Pareto fronts for different schemes on the same figure, we can compare which schemes are better.

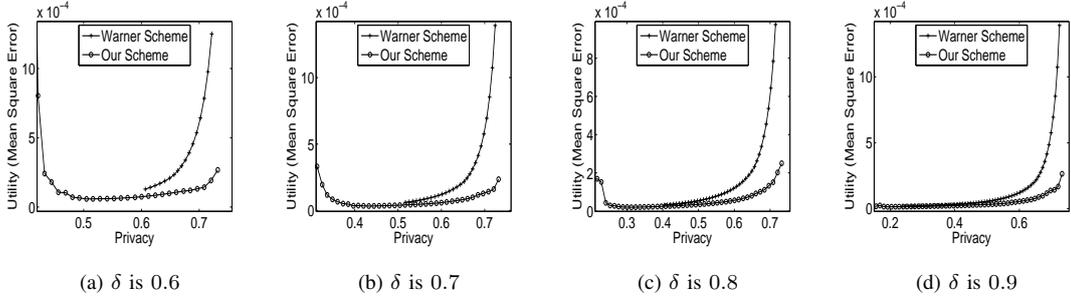


Fig. 4. For normal distribution with different  $\delta$

In our optimization problem, each RR-matrix-generation scheme decides a Pareto front in a two-dimensional space (we use the privacy objective as the x-axis and the utility objective as the y-axis). We can draw a Pareto front for each of the Warner, FRAPP, UP, and our schemes. Difference of these schemes can be illustrated by their Pareto fronts. For example, if a Pareto front of scheme  $A$  is consistently below that of scheme  $B$  within a privacy range, we can tell that  $A$  is better than  $B$  in that range, because to achieve the same degree of privacy within the range,  $A$  always has a smaller mean square error, the metrics used for measuring utility.

### B. Methodology

We would like to compare the Warner [1], FRAPP [11], UP [2] schemes with ours using Pareto fronts. However, Theorem 2 indicates that the Pareto fronts for the Warner, UP, and FRAPP schemes are the same. Therefore, it is sufficient to just compare our scheme with the Warner scheme.

For the Warner scheme, we let the diagonal element values (denoted by  $p$ ) change from 0 to 1 with a small step length of 0.001; accordingly we set the non-diagonal element values (based on the Warner scheme, these values are  $(1-p)/(n-1)$ , where  $n$  is the number of categories). This way, we can generate 1001 different RR matrices. For a given dataset, we can calculate the utility and privacy values for each of these RR matrices. After removing those non-optimal solutions, we can plot a Pareto front for the Warner scheme. For our scheme, we conduct the SPEA2-based algorithm for the same dataset to obtain an optimal set of RR matrices. We let the iteration stops after it loops 20,000 times. We also plot the Pareto front for our optimal set.

We have conducted our experiments for two types of data: synthetic and real data. The synthetic data sets are generated from various distributions. The real data set is the *Adult* data set from the UCI Machine Learning Repository [15].

### C. Experiments On Synthetic Dataset

To study how different distributions affect the performance of the Warner scheme and our scheme, we randomly generate a single-dimensional data set of 10,000 records. There are 10 different category values in these records; the probability of the categories in the data set follows a specific distribution.

We use DELL precision 340 (2.65Ghz CPU and 512M RAM) to conduct our experiments. It takes about 12 minutes to finish each experiment.

1) *For Different Privacy Bound  $\delta$* : We first show the experimental results for data sets from normal distribution using different privacy bounds  $\delta$  in Figure 4. From the figure, with the increment of  $\delta$ , the Pareto fronts of our results cover a larger range of the privacy than the Warner scheme. Namely, when the worst-case bound on privacy is set to  $\delta = 0.6, 0.7, 0.8, 0.9$ , Warner scheme cannot find an RR matrix with privacy less than 0.6, 0.5, 0.4, 0.22, respectively, while our scheme can find an RR matrix with privacy close to 0.4, 0.3, 0.22, 0.17. Moreover, our scheme achieves a lower MSE (better utility) than the Warner scheme when the RR matrices are compared in the same privacy range. Therefore, no matter what  $\delta$  we choose, our scheme always achieves better performance than the Warner scheme. Although we only show the results for normal distribution, the trends with respect to different  $\delta$  are similar for other distributions.

2) *For Different Distribution*: To study how different data distributions affect our results, we set  $\delta = 0.75$  and conduct experiments using the gamma and uniform distributions. We do not show the results with other  $\delta$  values because the trends are similar to those for normal distribution in Figure 4.

**(a) Gamma Distribution** For the gamma distribution, we have conducted experiments for different  $\alpha$  and  $\beta$  values ( $\alpha$  and  $\beta$  are parameters of the gamma distribution). We only depict the Pareto fronts using  $\alpha = 1.0$  and  $\beta = 2.0$  in Figure 5(a); other  $\alpha$  and  $\beta$  values have similar results. The figure shows that, for data based on gamma distribution, our scheme outperforms the Warner scheme in the entire privacy range. In particular, our scheme has about two times larger privacy range than the Warner scheme and has a much better performance than it when the privacy value is larger than 0.62.

**(c) Uniform Distribution** For the uniform distribution, the Pareto fronts of the Warner scheme and ours are plotted in Figure 5(b). Similar to other distributions, our scheme can find much better RR matrices than the Warner scheme. An exception is that our scheme covers the same privacy range as the Warner scheme.

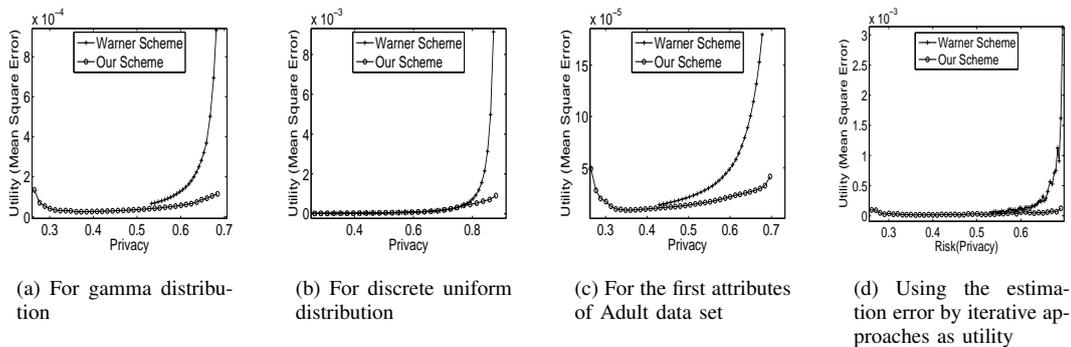


Fig. 5. For synthetic data and real data, and for the iterative approach.

#### D. Experiments On Real Data

For real data, we use the Adult data from the UCI Machine Learning Repository [15]. The Adult data set has 14 attributes, including both categorical and continuous attributes. We discretize those continuous attributes in order to apply the randomized response technique. Because of the page limitation, we only show the results for the first attributes in Figure 5(c); the results for the other attributes have shown a similar trend. The results clearly show that our scheme consistently outperforms the Warner scheme for all the attributes in the Adult data set. With the same DELL computer, it takes about 10 minutes to finish the experiment.

#### E. Use the Iterative Approach

In this experiment, we want to see whether the optimal set found in our scheme outperforms the Warner matrices if the iterative approach [2] is used to estimate the original data distribution. We use the same data set from gamma distribution of  $\alpha = 1.0$  and  $\beta = 2.0$  and the same optimal set of RR matrices as those in Figure 5(a). Instead of using the utility defined in Equation (10), we conduct the iterative approach to estimate the original distribution, and then calculate the mean square error (MSE) between the estimation and the original distribution. The results are plotted in Figure 5(d). The figure shows that our scheme still outperforms Warner scheme; namely our scheme has a wider privacy range and a much lower MSE (or higher accuracy) than the Warner scheme. Using other data sets, we get the similar results.

### VII. CONCLUSION

In this paper, based on estimate theories, we provide a method to quantify privacy and utility for the randomized response technique. We then apply an evolutionary multi-objective optimization technique to search for optimal RR matrices. We have conducted an extensive evaluation on our method. The evaluation shows that our scheme achieves much better performance than the existing RR schemes. Although we cannot claim that the RR matrices obtained from our scheme are optimal in the entire search space, our results do indicate that our scheme approaches the optimal RR matrices closer than any other existing RR schemes. In our future

work, we plan to extend our approach to the multi-dimensional randomized response technique.

### REFERENCES

- [1] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *The American Statistical Association*, vol. 60, no. 309, pp. 63–69, March 1965.
- [2] R. Agrawal, R. Srikant, and D. Thomas, "Privacy preserving olap," in *Proceedings of the ACM SIGMOD Conference (SIGMOD 2005)*, Baltimore, Maryland, USA, June 14-16 2005.
- [3] S. Rizvi and J. R. Haritsa, "Maintaining data privacy in association rule mining," in *Proceedings of the 28th VLDB Conference*, Hong Kong, China, 2002.
- [4] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, "Privacy preserving mining of association rules," in *Proceedings of 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada, July 2002.
- [5] W. Du and Z. Zhan, "Using randomized response techniques for privacy-preserving data mining," in *Proceedings of The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, August 24-27 2003, pp. 505–510.
- [6] V. K. S. Agrawal and J. R. Haritsa, "On addressing efficiency concerns in privacy-preserving mining," in *Proceedings of the 9th International Conference on Database Systems for Advanced Applications (DASFAA-2004)*, Jeju Island, Korea, March 2004.
- [7] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the 2000 ACM SIGMOD on Management of Data*, Dallas, TX USA, May 15 - 18 2000, pp. 439–450.
- [8] D. Agrawal and C. Aggarwal, "On the design and quantification of privacy preserving data mining algorithms," in *Proceedings of the 20th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Santa Barbara, California, USA, May 21-23 2001.
- [9] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the privacy preserving properties of random data perturbation techniques," in *the IEEE International Conference on Data Mining*, Melbourne, Florida, November 19 - 22, 2003 2003.
- [10] Z. Huang, W. Du, and B. Chen, "Deriving private information from randomized data," in *Proceedings of the ACM SIGMOD Conference (SIGMOD 2005)*, Baltimore, Maryland, USA, June 14-16 2005.
- [11] S. Agrawal and J. Haritsa, "A framework for high-accuracy privacy-preserving mining," in *In Proceedings of 21st IEEE Intl. Conf. on Data Engineering (ICDE)*, Tokyo, Japan, April 2005.
- [12] I. Das and J. Dennis, "A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems," *Structural Optimization*, vol. 14, no. 1, pp. 63–69, March 1997.
- [13] E. Zitzler, M. Laumanns, and L. Thiele, "Pea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization," in *Proceedings of Congress on Evolutionary Computation*, K. Giannakoglou, D. Tsahalis, J. Periaux, K. Papailiou, and T. Fogarty, Eds., Washington DC, USA, 2002, pp. 19–26.
- [14] V. Pareto, *Cours D'Economie Politique*. Lausanne: F. Rouge, 1896.
- [15] UCI: <http://www.ics.uci.edu/~mllearn/MLRepository.html>.