# Privacy Preserving OLAP

Rakesh Agrawal
IBM Almaden
ragrawal@us.ibm.com

Ramakrishnan Srikant
IBM Almaden
srikant@us.ibm.com

Dilys Thomas *
Stanford University
dilys@cs.stanford.edu

## ABSTRACT

We present techniques for privacy-preserving computation of multidimensional aggregates on data partitioned across multiple clients. Data from different clients is perturbed (randomized) in order to preserve privacy before it is integrated at the server. We develop formal notions of privacy obtained from data perturbation and show that our perturbation provides guarantees against privacy breaches.We develop and analyze algorithms for reconstructing counts of subcubes over perturbed data. We also evaluate the tradeoff between privacy guarantees and reconstruction accuracy and show the practicality of our approach.

## 1. INTRODUCTION

On-line analytical processing (OLAP) is a key technology employed in business-intelligence systems. The computation of multidimensional aggregates is the essence of on-line analytical processing. We present techniques for computing multidimensional count aggregates in a privacy-preserving way.

We consider a setting in which clients $C_1, C_2, \ldots C_n$ are connected to a server $S$. The server has a table $T(A_1, A_2, \ldots, A_m)$, where each column $A_i$ comes from a numeric domain. Each client $C_i$ contributes a row $r_i(a_{i_1}, a_{i_2}, \ldots, a_{i_m})$ to $T$. The server runs aggregate queries of the form

```
select count(*) from T
where P_j1 and P_j2 ... and P_jk.
```

Here $P_{j_i}$ is a range predicate of the form $a_{l_i} \leq A_{j_i} \leq a_{h_i}$, denoted as $A_{j_i}[a_{l_i}, a_{h_i}]$. We use count($P_{j_1} \wedge P_{j_2} \ldots \wedge P_{j_k}$) to succinctly represent the above aggregate query.

We take the randomization approach to preserving privacy. The basic idea is that every client $C_i$ perturbs its row $r_i$ before sending it to the server $S$. The randomness used in perturbing the values ensures information-theoretic row-level privacy. Figure 1 gives the schematic of our approach. $S$ runs queries on the resultant perturbed table $T'$. The query meant for the original table $T$ is translated into a set of queries on the perturbed table $T'$. The answers to these queries are then reconstructed to obtain the result to the original query with bounded error. We show that our techniques are safe against privacy breaches.

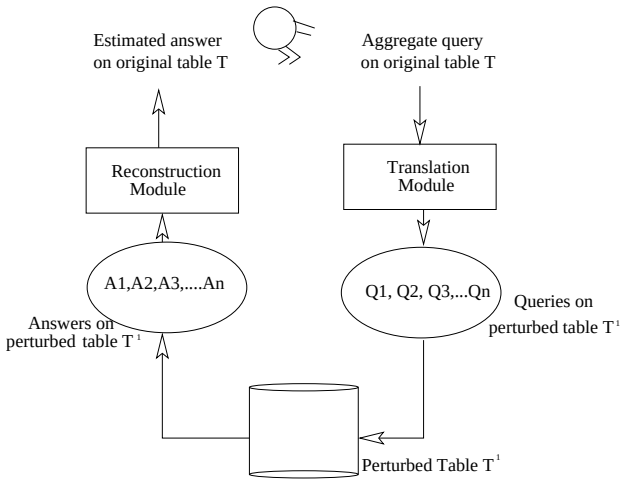


**Figure 1:** *Privacy preserving computation of multidimensional count aggregates.*

The perturbation algorithm is publicly known; the actual random numbers used in the perturbation, however, are hidden. To allow clients to operate independently, we use local perturbations so that the perturbed value of a data element depends only on its initial value and not on those of the other data elements. Different columns of a row are perturbed independently. We use *retention replacement* schemes where an element is decided to be retained with probability $p$ or replaced with an element selected from a probability distribution function (p.d.f.) on the domain of elements.

The proposed techniques can also be used for database tables in which some of the columns are categorical. They are also applicable in the settings in which the database tables are partitioned horizontally or vertically.

The organization of the rest of the paper is as follows. We start off with a discussion of related work in Section 2. Section 3 formally defines the retention replacement perturbation. Section 4 presents the reconstruction algorithms. Section 5 presents the guarantees against privacy breaches offered by our techniques. In Sec-

*Supported in part by NSF Grant ITR-0331640


*SIGMOD 2005* June 14-16, 2005, Baltimore, Maryland, USA.

tion 6, we discuss how our techniques can be extended to categorical data. We also discuss some additional perturbation techniques and describe how our techniques can be used in data mining by showing how to build a decision tree classifier. Section 7 presents an empirical evaluation of our techniques. We conclude with a summary and directions for future work in Section 8. The proofs of our results have been collected in the Appendix.

## 2. RELATED WORK

The techniques for preserving privacy while answering statistical queries developed in the statistical database literature can be classified into *query restriction*, *input perturbation* and *output perturbation* [1]. Both query restriction and output perturbation are applicable when the entire original unperturbed data is available in a single central repository, which is not true in our setting, where clients randomize their data before providing it to the server. Our scenario fits in the framework of input perturbation, where the goal is to create a version of the database that can be publicly released (e.g. census data), yet the individual rows should not be recoverable. Local perturbation for a single column has been studied in [24]. However most previous work (e.g., [20]) assume that during perturbation the entire database is available at a single site, while we require local perturbations at each client.

The use of local perturbation techniques to preserve privacy of individual rows while allowing the computation of data mining models at the aggregate level was proposed in [4]. They used an additive perturbation technique, in which a random perturbation is added to the original value of the row, where the perturbation is picked from another probability distribution function (e.g. Gaussian). They showed that it was possible to build accurate decision tree classification models on the perturbed data.

However, it is difficult to provide guarantees against privacy breaches when using additive perturbation. For instance, if we add a Gaussian random variable with a mean 0 and variance 20 to age, and for a specific row the randomized value happens to be $-60$, one can estimate with high confidence that the original value of age was (say) less than 20. Additive schemes are also restricted to numeric data. Finally, the algorithms in [4] reconstruct each column independently. Since OLAP requires queries over multiple columns, it is essential to be able to reconstruct them together.

The problem of privacy-preserving association-rule mining was studied in [9, 10, 21]. The randomization schemes used in these works are similar to the retention replacement schemes we consider. However these studies are restricted to boolean data.

Formal definitions of privacy breaches were proposed in [9], and an alternate approach to defining privacy guarantees was proposed in [6]. We adapt the definitions from [9] to allow more accurate reconstruction while still providing strong privacy guarantees. As our notion of privacy encompasses multiple correlated columns over vertically partitioned tables, it extends to privacy breaches (called disclosure risk) considering row linkage, studied in statistical disclosure control methods and [11].

There has been recent work [22, 23] to specify authorization and control inferences for OLAP data cubes. However the model assumes that the data resides at a single server, unlike our problem, where private data is integrated from multiple clients.

Another related area is that of *secure multiparty computation* [13, 25], that allows any function, whose inputs are shared between multiple clients to be evaluated, such that nothing other than the result is revealed. Since the general protocols are expensive, efficient protocols have been proposed for specific database and data mining operations, e.g. [3, 8, 12, 16, 19]. However, these protocols are designed for a small number of clients.

## 3. DATA PERTURBATION

A single record of the table is referred to as a *row*, while an attribute is referred to as a *column*. A single column from a single row is the granularity of perturbation and is referred to as a *data element*.

DEFINITION 1. **Perturbation Algorithm:** *A* perturbation algorithm $\alpha$ *is a randomized algorithm that given a table* $T$ *creates a table* $T'$ *having the same number of rows and columns.*

We will denote the unperturbed table as $T$ and the perturbed table as $T'$. The perturbation algorithm is public. However, the actual random numbers used by it are hidden.

Let $t_{ij}$ and $t'_{ij}$ denote the value of the element in the $i^{th}$ row of the $j^{th}$ column in tables $T$ and $T'$ respectively. The perturbation algorithm is said to be *local* if $t'_{ij}$ depends only on $t_{ij}$, while it is said to be *global* if $t'_{ij}$ depends on other elements in the $j^{th}$ column of $T$.

Let $D_j$ denote the domain of elements in the $j^{th}$ column of $T$. $D_j$ is said to be continuous for numeric columns, and discrete for categorical columns. For the class of perturbation algorithms we study, for every column being perturbed, we require the perturbation algorithm to select a fixed probability density function (p.d.f.) on the column's domain. For the $j^{th}$ column we call this p.d.f. the *replacing p.d.f.* on $D_j$. Both $D_j$ as well as the replacing p.d.f. on $D_j$ are public.

DEFINITION 2. **Retention Replacement Perturbation:** Retention replacement *perturbation is a perturbation algorithm, where each element in column* $j$ *is retained with probability* $p_j$*, and with probability* $(1-p_j)$ *replaced with an element selected from the replacing p.d.f. on* $D_j$*. That is,*

$$t'_{ij} = \begin{cases} t_{ij} & \text{with probability } p_j \\ \text{element from replacing p.d.f. on } D_j & \text{with probability } (1\text{-}p_j). \end{cases}$$

If column $j$ of the table can be revealed without perturbation we set $p_j = 1$.

Retention replacement perturbation, where the replacing p.d.f. is the uniform p.d.f. is called *uniform perturbation*. We assume that each column of the table $T'$ has been perturbed independently using uniform perturbation. In Section 6.2, we show that uniform perturbation provides better privacy guarantees for rare events. Other alternatives and comparisons are also given in the same section.

## 4. RECONSTRUCTION

An aggregate function on the original table $T$, must be reconstructed by accessing the perturbed table $T'$. The accuracy of the reconstruction algorithm is formalized below by the notion of approximate probabilistic reconstructability.

DEFINITION 3. **Reconstructible Function:** *Given a* perturbation $\alpha$ *converting table* $T$ *to* $T'$*, a numeric function* $f$ *on* $T$ *is said*

*to be $(n, \epsilon, \delta)$ reconstructible by a function $f'$, if $f'$ can be evaluated on the perturbed table $T'$ so that $|f' - f| < max(\epsilon, \epsilon f)$ with probability greater than $(1-\delta)$ whenever the table $T$ has more than $n$ rows. The probability is over the random choices made by $\alpha$.*

For boolean functions, $(n, \delta)$ reconstructability needs $f$ and $f'$ to agree exactly with probability greater than $(1 - \delta)$.

Referring to Figure 1, to answer the aggregate query $count(P_1 \wedge P_2 \wedge \ldots P_k)$ on $k$ columns of the original table, $T$, a set of $2^k$ queries, $count(P_1 \wedge P_2 \wedge \ldots P_k)$, $count(\neg P_1 \wedge P_2 \wedge \ldots P_k)$, $count(P_1 \wedge \neg P_2 \wedge \ldots P_k)$, $count(\neg P_1 \wedge \neg P_2 \wedge \ldots P_k) \ldots count(\neg P_1 \wedge \neg P_2 \wedge \ldots \neg P_k)$ are generated. These queries are evaluated on the perturbed table $T'$. The answers on $T'$ are reconstructed into estimated answers for the same queries on $T$, which include the answer to the original query.

Without loss of generality, assume that the predicates are only over perturbed columns. We present reconstruction algorithms for numeric columns. These algorithms can be extended to categorical columns too as shown in Section 6.

## 4.1 Reconstructing Single Column Aggregates

Consider the uniform retention replacement perturbation with retention probability $p$ applied on a database with $n$ rows and a single column, $C$, with domain $[min, max]$. Consider the predicate $P = C[low, high]$. Given the perturbed table $T'$, we show how to estimate an answer to the query $count(P)$ on $T$.

Let tables $T$, $T'$ each have $n$ rows. Let $n_r = count(P)$ evaluated on table $T'$, while $n_o = count(P)$ estimated for table $T$. Given $n_r$ we estimate $n_o$ as

$$n_o = \frac{1}{p}(n_r - n(1-p)b) \quad , \quad where \quad b = \frac{high - low}{max - min}.$$

The intuition is that out of the $n$ rows in table $T$, the expected number of rows that get perturbed is $n(1-p)$. For uniform perturbation, a $b$ fraction of these rows, i.e. $n(1-p)b$ rows, will be expected to lie within the $[low, high]$ range. The total number of rows observed in range $[low, high]$ in $T'$, $n_r$, can be seen as the sum of those rows that were decided to be perturbed into $[low, high]$ (from outside, or perturbed and retained within the interval) and those rows that were unperturbed in the original interval. Subtracting the $n(1-p)b$ perturbed rows from $n_r$, we get an estimate for the number of unperturbed rows, with values in $[low, high]$ in $T$. This is scaled up by $1/p$ to get the total number of original rows in $T$ in $[low, high]$, as only a $p$ fraction of rows were retained.

The fraction $f$ of rows originally in $[low, high]$ is therefore estimated as

$$f' = \frac{n_o}{n} = \frac{n_r}{pn} - \frac{(1-p)(high - low)}{p(max - min)}.$$

Not only is the above estimator a Maximum Likelihood Estimator (MLE) as shown in Section 4.2, it reconstructs an approximate answer with high probability.

THEOREM 1. *Let the fraction of rows in $[low, high]$ in the original table $f$ be estimated by $f'$, then $f'$ is a $(n, \epsilon, \delta)$ estimator for $f$ if $n \geq 4\log(\frac{2}{\delta})(p\epsilon)^{-2}$.*

We now formalize the above reconstruction procedure. This formalization provides the basis for the reconstruction of multiple columns in Section 4.2.

Let vector $y = [y_0, y_1] = [count(\neg P), count(P)]$ be the answers on table $T'$, and let vector $x = [x_0, x_1] = [count(\neg P), count(P)]$ denote the estimates for table $T$. Let $b$ be defined as before and $a = 1 - b$. As only table $T'$ is available, $x$ is estimated using the constraint $xA = y$, which gives the estimator $x = yA^{-1}$. Here $A$ is the following transition matrix

$$\begin{bmatrix} (1-p)a + p & (1-p)b \\ (1-p)a & (1-p)b + p \end{bmatrix}.$$

The element in the first row and first column of $A$, $a_{00} = (1-p)a + p$ is the probability that an element originally satisfying $\neg P$ in $T$ after perturbation satisfies $\neg P$ in $T'$. This probability was calculated as the sum of the probabilities of two disjoint events. The first being that the element is retained, which occurs with probability $p$. The second being that the element is perturbed and after perturbation satisfies $\neg P$, which together has probability $(1-p)a$. The element $a_{01}$ is the probability that an element satisfying $\neg P$ in $T$ after perturbation satisfies $P$ in $T'$. The element $a_{10}$ is the probability that an element satisfying $P$ in $T$ after perturbation satisfies $\neg P$ in $T'$. The element $a_{11}$ is the probability that an element satisfying $P$ in $T$ after perturbation satisfies $P$ in $T'$. Their values were similarly derived.

If $y = [n - n_r, n_r]$ and $x = [n - n_o, n_o]$, the solution to the equation below gives the same estimator as derived earlier:

$$\begin{bmatrix} n - n_o & n_o \end{bmatrix} \begin{bmatrix} (1-p)a + p & (1-p)b \\ (1-p)a & (1-p)b + p \end{bmatrix} = \begin{bmatrix} n - n_r & n_r \end{bmatrix}.$$

## 4.2 Reconstructing Multiple Column Aggregates

Assume now that the uniform retention replacement perturbation, with retention probability $p$, has been applied to each of $k$ columns of a table, $T$. Consider the aggregate query $count(P_1 \wedge P_2 \wedge ...P_k)$ on table $T$. In practice $k$ is small.

We create a $k \times 2$ matrix, $R$, with $k$ rows and 2 columns, having 1 row for each query column. $R_{i,1}$ gives the probability that a number randomly selected from the replacing p.d.f. for column $i$ will satisfy predicate $P_i$, while $R_{i,0}$ is the probability of the complementary event, that a number selected from the replacing p.d.f. will satisfy $\neg P_i$.

Take for instance the query, Q=count(age[30-45] $\wedge$ salary[50k-120k] $\wedge$ house-rent[700-1400]) with the domains for age, salary and house-rent being [0-100], [25k-200k], [500-2500]. Then $R$ will be $[[0.85, 0.15], [0.6, 0.4], [0.65, 0.35]]$, since the first column being age[30-45] implies $R_{1,1} = (45 - 30)/(100 - 0) = 0.15$, while $R_{1,0} = 1 - 0.15 = 0.85$, etc.

As stated earlier, to answer the query $count(P_1 \wedge P_2 \ldots P_k)$, we ask $2^k$ aggregate queries on the perturbed table, $T'$. The $2^k$ answers on perturbed table $T'$ are converted into estimated answers to these $2^k$ aggregate queries on the original table T, which includes the estimated answer to the original query.

Let $y$ be a row vector of size $2^k$ that has the answers to the above queries on perturbed table $T'$, and let $x$ be a row vector of size $2^k$ that has the reconstructed estimated answers to the queries on original table $T$. We order the answers to the $2^k$ queries in vectors $x$, $y$ using the bit representation of the vector index as shown in Figure 2. Let $Q(r, 1)$ denote the predicate$(P_r)$ on the $r^{th}$ column of query $Q$, and $Q(r, 0)$ its negation $(\neg P_r)$. Let $bit(i, r)$ denote the $r^{th}$

| Query | Estimated on $T$ | Evaluated on $T'$ |
|---|---|---|
| $count(\neg P_1 \wedge \neg P_2)$ | $x_0$ | $y_0$ |
| $count(\neg P_1 \wedge P_2)$ | $x_1$ | $y_1$ |
| $count(P_1 \wedge \neg P_2)$ | $x_2$ | $y_2$ |
| $count(P_1 \wedge P_2)$ | $x_3$ | $y_3$ |

**Figure 2: Answering query** $count(P_1 \wedge P_2)$

bit from the left in the binary representation of the number $i$ using k bits. Then,
$x_i = count(\wedge_{r=1}^{k} Q(r, bit(i,r)))$ in $T$, for $0 \leq i \leq 2^k - 1$;
$y_i = count(\wedge_{r=1}^{k} Q(r, bit(i,r)))$ in $T'$, for $0 \leq i \leq 2^k - 1$.

For example, for the query $count$(age[30-45] $\wedge$ salary[50k-120k] $\wedge$ house-rent[700-1400]), $y[6_{10}] = y[110_2] = count$(age[30-45] $\wedge$ salary[50k-120k] $\wedge \neg$ house-rent[$700 - 1400$])

By a single scan through the perturbed table $T'$ vector $y$ can be calculated. Vector $x$ is reconstructed from vector $y$ using the matrix inversion technique or the iterative Bayesian technique described below. The data analyst may either be interested only in the component $x_{2^k-1}$, which is the answer to the $count(\wedge_{r=1}^{k} P_r)$ query on $T$, or she may be interested in the entire vector $x$.

### 4.2.1 Matrix Inversion technique

If $p_r$ is the retention probability for the $r^{th}$ column, we calculate vector $x$ from vector $y$ as $x = yA^{-1}$. The transition matrix, $A$, with $2^k$ rows and $2^k$ columns, can be calculated as the tensor product [15] of matrices

$$A = A_1 \otimes A_2 \otimes A_3 .... \otimes A_k$$

where the matrix $A_r$, for $1 \leq r \leq k$ is the transition matrix for column $r$ (see Section 4.1).

$$A_r = \begin{bmatrix} (1-p_r)a_r + p_r & (1-p_r)b_r \\ (1-p_r)a_r & (1-p_r)b_r + p_r \end{bmatrix}$$

where $b_r = R_{r,1}$ and $a_r = R_{r,0} = 1 - R_{r,1}$.

The entries of the tensor product matrix, $A$, can be explicitly calculated to be
$a_{ij} = \prod_{r=1}^{k}((1-p_r) \times R_{r,bit(j,r)} + p_r \times \delta_{(bit(i,r),bit(j,r))})$, $\forall 0 \leq i < 2^k, 0 \leq j < 2^k$
where $\delta_{(c,d)} = 1$ if $c = d$, and 0 if $c \neq d$, for $c, d \in \{0, 1\}$.

We split the space of possible evaluations of a row into $2^k$ states, according to which of the $2^k$ mutually exclusive predicate combinations the row satisfies. We say a row is said to belong to state $i$ if it satisfies the predicate $\wedge_{r=1}^{k} Q_(r, bit(i,r))$. For example, from Figure 2, a row in state 0 satisfies $\neg P_1 \wedge \neg P_2$ while a row in state 1 satisfies $\neg P_1 \wedge P_2$ etc.

The entry $a_{ij}$ of matrix $A$ above represents the probability that a row belonging to state $i$ in $T$, after perturbation belongs to state $j$ in $T'$. As each column was independently perturbed the probability of transition from state $i$ to state $j$ is the product of the probabilities for the transitions on all columns. The contribution from the $r^{th}$ column to the transition probability is the sum of $(1-p_r) \times R_{r,bit(j,r)}$, if the element was decided to be perturbed, and $p_r \times \delta_{(bit(i,r),bit(j,r))}$, if the element was decided to be retained. The term $\delta_{(bit(i,r),bit(j,r))}$ ensures that the retention probability $p_r$ adds up only if the source and destination predicates on the $r^{th}$ column are the same for states $i$ and $j$. Thus the probability of transition from state $i$ to state $j$ on the $r^{th}$ column is $(1-p_r) \times R_{r,bit(j,r)} + p_r \times \delta_{(bit(i,r),bit(j,r))}$. The product of this probability over all columns gives the probability of transition from state $i$ to state $j$, $a_{ij}$.

THEOREM 2. *The vector x calculated as $A^{-1}y$ is the maximum likelihood estimator (MLE) of the relaxed a priori distribution ( $\sum_i x_i = n$ and $0 \leq x_i \leq n$ are the exact constraints, the relaxed constraint only ensures $\sum_i x_i = n$) on the states that generated the perturbed table.*

The multiple column aggregate is $(n, \epsilon, \delta)$ reconstructible, is shown by applying the Chernoff bound, to bound the error in $y$, and then bounding the error added during inversion.

### 4.2.2 Iterative Bayesian technique

Let vectors $x$ and $y$ of size $2^k$ be the a priori distribution on states of the original rows, and posteriori distribution on states of perturbed rows, as introduced above. Let the original states of rows in $T$ selected from the a priori distribution be given by random variables $U_1, U_2, ....U_n$, while the states of the $n$ perturbed rows in $T'$ be given by the random variables $V_1, V_2, ...V_n$. Then for $0 \leq p, q \leq t = (2^k - 1)$ and $1 \leq i \leq n$, we have $Pr(V_i = q) = y_q/n$, and $Pr(U_i = p) = x_p/n$. Also $Pr(V_i = q | U_i = p) = a_{pq}$ is the transition probability from state $p$ to $q$.

From Bayes rule, we get

$$\begin{aligned} Pr(U_i = p | V_i = q) &= \frac{P(V_i = q | U_i = p) P(U_i = p)}{P(V_i = q)} \\ &= \frac{P(V_i = q | U_i = p) P(U_i = p)}{\sum_{r=0}^{t} P(V_i = q | U_i = r) P(U_i = r)} \\ &= \frac{a_{pq} \frac{x_p}{n}}{\sum_{r=0}^{t} a_{rq} \frac{x_r}{n}} \\ &= \frac{a_{pq} x_p}{\sum_{r=0}^{t} a_{rq} x_r}. \end{aligned}$$

We iteratively update $x$ using the equation

$$Pr(U_i = p) = \sum_{q=0}^{t} Pr(V_i = q) Pr(U_i = p | V_i = q).$$

This gives us the update rule,

$$x_p^{T+1} = \sum_{q=0}^{t} y_q \frac{a_{pq} x_p^T}{\sum_{r=0}^{t} a_{rq} x_r^T},$$

where vector $x^T$ denotes the iterate at step $T$, and vector $x^{T+1}$ the iterate at step $T + 1$.

We initialize the vector, $x^0 = y$, and iterate until two consecutive $x$ iterates do not differ much. This fixed point is the estimated a priori distribution. This algorithm is similar to the iterative procedure proposed in [4] for additive perturbation and shown in [2] to be the *Expectation Maximization* (EM) algorithm converging to the *Maximum Likelihood Estimator* (MLE).

### 4.2.3 Error in Reconstruction

We provide here a brief analysis of the error in the reconstruction procedures. A quantitative analysis of the magnitude of error is easy for the inversion method, but such an analysis is much harder for the iterative method. Due to the randomization in the perturbation algorithm there are errors in the transition probabilities in

matrix $A$. This causes $y$, the posteriori distribution after perturbation calculated from $T'$, to have errors. Hence the reconstructed $x$ will have errors.

The error decreases as the number of rows, $n$, increases. Let $a'_{ij}$ denote the actual fraction of original rows of state $i$ that were converted to state $j$. Then as $n$ increases, $a_{ij}$ will be a closer approximation to $a'_{ij}$. The error decreases as $n^{-0.5}$ as indicated by Theorem 1, and verified empirically in Section 7.

The error in reconstruction increases as the number of reconstructed columns, $k$, increases, and the probability of retention, $p$, decreases. The largest and smallest eigenvalues of $A$ can be shown to be 1 and $p^k$ respectively and the condition number of the matrix $A$ grows roughly as $p^{-k}$ (see Section 7). The condition number of a matrix is a good indicator of the error introduced during inversion [14].

# 5. GUARANTEES AGAINST PRIVACY BREACHES

Private data from multiple clients is perturbed before being integrated at the server. In this section, we formalize the privacy obtained by this perturbation.

The notion of a $(\rho_1, \rho_2)$ privacy breach was introduced in [9]. We extend this to introduce a new privacy metric, called the $(s, \rho_1, \rho_2)$ privacy breach. Consider a database of purchases made by individuals. It is quite likely that many people buy bread, but not many buy the same prescription medicine. The new metric is more concerned about whether an adversary can infer from the randomized row which medicine a person bought, and is less concerned about the adversary determining with high probability that the original row had bread, as most individuals buy bread and it does not distinguish the individual from the rest of the crowd.

Assume that the adversary has access to the entire perturbed table $T'$ at the server, and the exact a priori distribution on the unperturbed data (which can be reconstructed [4])[1]. Also assume that any external information is already incorporated into the database.

## 5.1 Review of $(\rho_1, \rho_2)$ Privacy Breach

Consider a data element of domain $V_X$ perturbed by a perturbation algorithm into another domain $V_Y$.

DEFINITION 4. $(\rho_1, \rho_2)$ **Privacy Breach***[9]: Let Y denote the random variable corresponding to the perturbed value and X that corresponding to the original value obtained from the a priori distribution. We say that there is a $(\rho_1, \rho_2)$ privacy breach with respect to $Q \subseteq V_X$ if for some $S \subseteq V_Y$ $P[X \in Q] \leq \rho_1$ and $P[X \in Q | Y \in S] \geq \rho_2$ where $0 < \rho_1 < \rho_2 < 1$ and $P[Y \in S] > 0$.*

Intuitively suppose the probability of an event, (age $\leq 10$) (say), according to the a priori probability is $\leq \rho_1 = 0.1$ (say). After observing the perturbed value, if the posteriori probability of the same event increases to $\geq \rho_2 = 0.95$ (say), then there is a (0.1,0.95) privacy breach with respect to the event (age $\leq 10$).

## 5.2 $(s, \rho_1, \rho_2)$ Privacy Breach

[1]From Section 4.1, the error in the reconstructed a priori distribution for very selective predicates is large. This adds to the privacy of the perturbed rows.

In retention replacement perturbations, which are of interest to us, the column is perturbed back into the same domain, and hence $V_X = V_Y$. Let $S \subseteq V_X$, with $P[X \in S] = p_s$, for $X \in_o V_X$ where $\in_o$ represents selecting an element from $V_X$ according to the a priori distribution on $V_X$. Let $P[Y \in S] = m_s$, for $Y \in_r V_X$, where $\in_r$ represents selecting an element from $V_X$ according to the replacing distribution, which is different from the distribution of the perturbed table. The ratio $p_s/m_s$ is called the *relative a priori probability* of the set $S$.

The relative a priori probability is a dimensionless quantity that represents how frequent a set is according to its a priori probability as compared to the replacing p.d.f. (the uniform p.d.f.). In a database of purchases, medicines will have low relative a priori probability since different people take different medicines, while bread will have high relative a priori probability.

DEFINITION 5. $(s, \rho_1, \rho_2)$ **Privacy Breach:** *Let Y denote the random variable corresponding to the perturbed value and X that corresponding to the original value obtained from the a priori distribution.*

*Let $S \subseteq V_X$, we say that there is a $(s, \rho_1, \rho_2)$ privacy breach with respect to $S$ if the relative a priori probability of $S$, $p_s/m_s < s$, and if $P[X \in S] = p_s \leq \rho_1$ and $P[X \in S | Y \in S] \geq \rho_2$ where $0 < \rho_1 < \rho_2 < 1$ and $P[Y \in S] > 0$.*

The value of $s$ in the privacy breach is addressed by the next result.

THEOREM 3. *The median value of relative a priori probability, over all subsets $S$, $S \subseteq V_X$, is 1.*

We define rare sets as those that have relative a priori probability smaller than 1. We next show that privacy breaches do not happen for rare sets.

## 5.3 Single Column Perturbation

THEOREM 4. *Let $p$ be the probability of retention, then uniform perturbation applied to a single column is secure against a $(s, \rho_1, \rho_2)$ breach, if*

$$s < \frac{(\rho_2 - \rho_1)(1 - p)}{(1 - \rho_2)p}.$$

As a concrete example, for uniform perturbation, with p=0.2, there are no (68, 0.1, 0.95) breaches. This means for any set $S$, if $\rho_2 > 0.95$ with uniform perturbation, $\rho_1$ will be large ($> 0.1$) when $p_s/m_s < 68$. In fact, for a rare set, with $s < 1$, there will be no (0.937, 0.95) privacy breaches in the original $(\rho_1, \rho_2)$ model for this perturbation.

## 5.4 Multiple Independently Perturbed Columns

Let $D_i$ be the domain for column $i$ in a $k$ column table. Then the domain of the table, $D = D_1 \times D_2 \times \ldots D_k$. Each column of the table is perturbed independently by a retention replacement perturbation scheme.

There is an a priori probability distribution of the rows in table $T$. Let $S_i \subseteq D_i$ be a subset of the domain of the $i^{th}$ column for $1 \leq i \leq k$. Let $S = S_1 \times S_2 \times \ldots S_k$, then $S \subseteq D$. Let

$P[S] = p_{S_1 \times S_2 \times \ldots S_k} = p_s$ (say) be the a priori probability of $S$. Let $P[Y_i \in S_i] = m_{S_i}$, for $Y_i \in_{\alpha_i} D_i$, where $\in_{\alpha_i}$ denotes selecting randomly from the replacing p.d.f. on $D_i$, for all $1 \leq i \leq k$. Then $P[Y \in S] = m_{S_1} m_{S_2} .. m_{S_k} = m_s$ (say) for $Y = (Y_1, Y_2, \ldots Y_k) \in_\alpha D$, where $\in_\alpha$ denotes selecting randomly from the replacing p.d.f. for each column independently. $p_s/m_s$, the relative a priori probability, is the ratio of the a priori probability to the replacing probability, of the combination of values for the columns together. Correlated columns with higher a priori probabilities have larger values of $p_s/m_s$.

THEOREM 5. *There will not be a $(\rho_1, \rho_2)$ privacy breach with respect to $(S_1 \times S_2 \times \ldots S_k) = S \subseteq D$, if*

$$\frac{p_s}{m_s} < \frac{\rho_2(1-\rho_1)(1-p)^k}{(1-\rho_2)\prod_{i=1}^{k}((1-p)m_{S_i}+p)}.$$

$S_i$ denotes the subset on column $i$ within which the original value must be identified for the privacy breach. In the case, $S_i$ denotes a single value or a small range within the domain of a continuous column, hence $(1-p)m_{S_i} \ll p$. We approximate $(1-p)m_{S_i} + p$ by $p$ to get

$$\frac{\rho_2(1-\rho_1)(1-p)^k}{(1-\rho_2)\prod_{i=1}^{k}((1-p)m_{S_i}+p)} \geq \frac{\rho_2(1-\rho_1)(1-p)^k}{(1-\rho_2)p^k}(1-\epsilon)$$

for some small constant $\epsilon$. Thus for some small constant $\epsilon$, uniform perturbation applied individually to $k$ columns is secure against $(s, \rho_1, \rho_2)$ breaches for

$$s < \frac{\rho_2(1-\rho_1)(1-p)^k}{(1-\rho_2)p^k}(1-\epsilon).$$

As an example, for uniform perturbation with p=0.2 applied independently to two columns, there are no (273,0.1,0.95) breaches for joint events on the columns (when $m_{S_i}$ are small).

# 6. EXTENSIONS

## 6.1 Categorical Data

Consider a categorical column, $C$, having discrete domain $D$. Let $S \subseteq D$. A predicate $P$, on column $C$, using $S$ is defined as

$$P(x) = \begin{cases} true & \text{if } x \in S \\ false & otherwise. \end{cases}$$

Given the a priori and replacing p.d.f. on $D$, the reconstruction algorithms in Section 4 and the privacy guarantees in Section 5 can be directly applied to the categorical data by computing the probability of the predicate, $P$, being true.

## 6.2 Alternative Retention Replacement Schemes

Our analysis so far considered retention replacement perturbations where the replacing p.d.f is the uniform distribution. We now discuss some other interesting retention replacement schemes:

1 *Identity perturbation:* If the original data element is decided to be perturbed, the data element is replaced by a random element selected uniformly among all data elements [18] (i.e. the replacing p.d.f. is the same as the a priori distribution).

2 *Swapping:* Swapping is closely related to identity perturbation. In swapping with probability $p$ we retain a data element, and with probability $(1-p)$ we decide to replace it. Numbers decided to be replaced are then randomly permuted amongst themselves.

Identity perturbation and swapping are different from uniform perturbation which is a local perturbation. Identity perturbation can be local if there is knowledge of the a priori distribution before perturbation. Swapping is not a local perturbation and requires multiple rows at the client.

### 6.2.1 Reconstructing Aggregates

Identity perturbation and swapping do not affect the answers to single column aggregate queries, i.e. answers to single column aggregate queries on the perturbed table, $T'$, are returned directly as answers to those queries on the original table, $T$.

The difference in multi-column reconstruction for identity perturbation and swapping as compared to uniform perturbation is in the evaluation of vector $R$ in Section 4.2. Recall that $R_{i,1}$ is the probability that an element selected from the replacing p.d.f. on column $i$ satisfies the predicate on the $i^{th}$ column, $P_i$. The replacing p.d.f. (which is the original p.d.f. for identity perturbation and swapping) is required for reconstruction. This requires the server to have the original p.d.f. for each column. This requirement is however obviated by the observation in the previous paragraph, that the fraction of elements satisfying $P_i$ in $T$ is the same as the fraction of elements satisfying $P_i$ in $T'$. Hence $R_{i,1}$ can be calculated from $T'$. $R_{i,0}$ as before is calculated as $1 - R_{i,1}$.

The reconstruction error after identity perturbation and swapping will be smaller than that compared to uniform perturbation for sets, $S$, with small relative a priori probability. This is because in uniform perturbation the noise due to the perturbed data elements that now belong to $S$, but did not before perturbation, exceeds significantly the number of data elements that were in $S$ originally and retained during perturbation.

### 6.2.2 Guarantees against Privacy Breaches

The guarantees for identity perturbation and swapping can be obtained using $m_{S_i} = p_{S_i}$ in Theorems 4 and 5. As an example we restate Theorem 4 for identity perturbation.

LEMMA 1. *For a single column, identity perturbation is secure against $(s, \rho_1, \rho_2)$ privacy breaches for*

$$\rho_1 < \frac{\rho_2 - p}{1-p}.$$

PROOF: For identity perturbation, $m_s = p_s$, hence $p_s/m_s = 1 \quad \forall S$. Repeating the argument in Theorem 4 we get $(\rho_2 - \rho_1)(1-p) > (1-\rho_2)p$, which implies the result. □

The above $(\rho_1, \rho_2)$ guarantee for identity perturbation is independent of the subset $S$. Uniform perturbation gives better $(\rho_1, \rho_2)$ guarantees for a set of rare data elements, i.e. a set with $p_s/m_s < 1$ and worse for sets with $p_s/m_s > 1$. Identity perturbation and swapping have a privacy breach in the presence of external knowledge about rare values (eg. the largest or smallest value). Rare values need to be suppressed (i.e. blanked out) [17] for privacy with these perturbations.

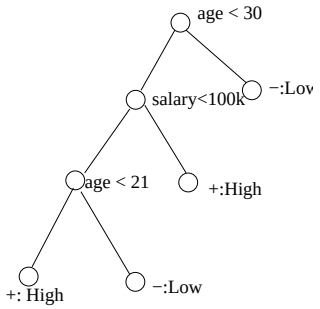


**Figure 3:** *Decision Tree Example*

## 6.3 Application to Classification

We show how aggregate queries on multiple columns can be used for privacy preserving construction of decision trees [4]. Consider the tree in Figure 3 built on randomized table $T'$ with schema (age, salary, house-rent, class-variable) to predict the column class-variable. The column class-variable can take two values: + and − representing high and low credit-risk (say). The private columns among age, salary, house-rent and class-variable, are each independently perturbed by a retention replacement perturbation. Let $Q$ denote the predicate (class-variable = '+') while $\neg Q$ denote the predicate (class-variable='-').

For the first split, say on (age < 30), the gini index is calculated using the estimated answers of the four queries: count(age[0-30] $\wedge$ $\neg$ Q), count($\neg$ age[0,30] $\wedge$ $\neg$ Q), count(age[0-30]$\wedge$ Q ) and count($\neg$ age[0,30]$\wedge$ Q ) on $T$. Now consider the left subtree of elements having (age < 30) using the predicate (salary < 100$k$). We do not partition the randomized rows at any level in the decision tree. Previously with additive perturbation, randomized rows were partitioned, and the columns were reconstructed independently [4]. With multi-column reconstruction the queries count(age[0-30] $\wedge$ salary[25k-100k] $\wedge$ $\neg$ Q), count(age[0,30] $\wedge$ salary[100k-200k] $\wedge$ $\neg$ Q ), count(age[0-30] $\wedge$ salary[25k-100k] $\wedge$ Q ) and count(age[0,30] $\wedge$ salary[100k-200k] $\wedge$ Q ) are reconstructed for $T$, to calculate the gini index or another split criterion at this level.

Now consider the third split, on age once again, but this time (age < 21), is decided after the queries count(age[0-21] $\wedge$ salary[25k-100k] $\wedge$ $\neg$ Q ), count(age[21-30] $\wedge$ salary[25k-100k] $\wedge$ $\neg$ Q ) count(age[0-21] $\wedge$ salary[25k-100k] $\wedge$ Q ) and count(age[21-30] $\wedge$ salary[25k-100k] $\wedge$ Q ) are reconstructed for $T$. The number of columns in the count query did not increase at this split on age, which was already present among the original set of queried columns.

# 7. EXPERIMENTS

We next present an empirical evaluation of our algorithms on real as well as synthetic data. For real data, we used the Adult dataset, from the UCI Machine Learning Repository [5], which has census information. The Adult dataset contains about 32,000 rows with 4 numerical columns. The columns and their ranges are: age[17 - 90], fnlwgt[10000 - 1500000], hrsweek[1 - 100] and edunum[1 - 16].

For synthetic data, we used uncorrelated columns of data having Zipfian distribution with zipf parameter 0.5. We create three such tables with different number of rows. The number of rows is varied in factors of 10 from $10^3$ to $10^5$. The frequencies of occurrences are such that the least frequent element occurs 5 times. This results in the number of distinct values to be approximately one tenth of the number of rows in the table.

## 7.1 Randomization and Reconstruction

In this Section we assume that the vectors, $x$, $y$ described in Section 4.2 have been normalized, i.e. all elements have been divided by $n$, the number of rows, so that the sum of the elements of each vector is 1. These vectors will also be referred to as probability density function (p.d.f.) vectors. $x$ is the reconstructed p.d.f. vector, obtained by the inversion or iterative method in Section 4.2, while $y$ is the p.d.f. vector on the perturbed table before reconstruction. Let the exact original value of the p.d.f. vector calculated directly on the unperturbed table, $T$, be $x'$. The $l_1$ norm of the difference between the estimated ($x$) and actual ($x'$) p.d.f. vectors is used as the metric of error, and is referred to as the *reconstruction error*. The results of the reconstruction algorithm are quite accurate when the reconstruction error is much smaller than 1.

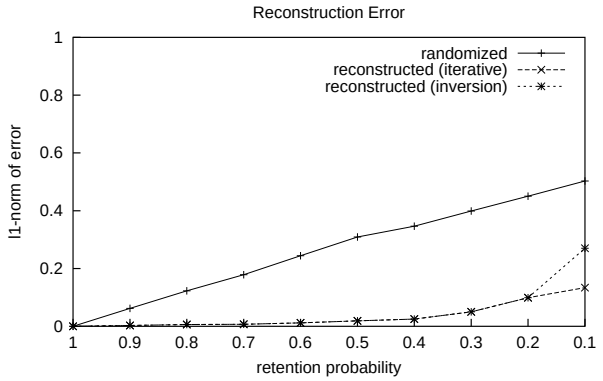


**Figure 4:** *Reconstruction errors for conjunction of 2 predicates for Adult data.*

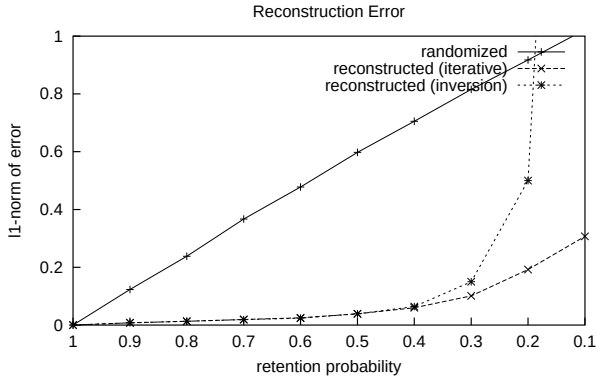


**Figure 5:** *Reconstruction errors for conjunction of 3 predicates for Adult data.*

**Reconstruction algorithms:** We first study the reconstruction error while reconstructing multiple columns of the Adult dataset for varying retention probabilities. The predicates being reconstructed are age[25-45], fnlwgt[100000-1000000] and hrsweek[30-

60]. Figure 4 shows the errors on first two among the above predicates while Figure 5 shows the errors on all three predicates. The retention probability, $p$, plotted on the $x$-axis, is the same for all columns. The reconstruction error is plotted on the $y$ axis. There are three curves in each figure. The curve *randomized*, shows the $l_1$ norm of the difference between the perturbed p.d.f. vector $y$ and the original p.d.f. vector $x'$. It serves as a baseline to study the reduction in error after reconstruction of $y$ to $x$. The other two curves represent the reconstruction errors after the *iterative* and the *inversion* algorithms.

The iterative procedure gives smaller errors than the inversion procedure, especially when a larger number of columns are reconstructed together, and the probability of retention, $p$, is small. This is reconfirmed later by Figures 7 and 8, and similar experiments on synthetic data (which we do not show for the lack of space). This may seem unintuitive as the inversion algorithm was shown to give the MLE estimator for $x$, satisfying $\sum_i x_i = 1$ (after normalization). This can be explained by noting that the iterative algorithm gives the MLE estimator in the constrained space, i.e. for the subspace of $\sum_i x_i = 1$ that satisfies $0 \leq x_i \leq 1 \quad \forall i$. Since the number of rows are always non-negative, this is the subspace that contains the exact original p.d.f. vector $x'$. When the retention probability decreases, and the number of columns to be reconstructed increases, the error during randomization and reconstruction increases, and the inversion algorithm may return a point outside the constrained space. The reconstruction error by the inversion method can grow arbitrarily. However, the iterative algorithm being constrained, will have a reconstruction error of at most two.

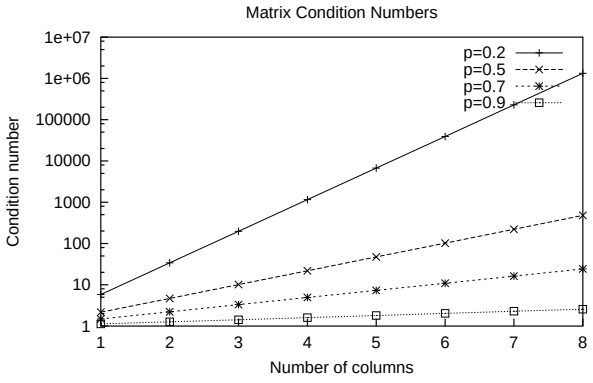


**Figure 6:** *Condition number of the transition matrix*

**Condition number:** Figure 6 shows the condition number [14] of the transition matrix using a logarithmic scale on the $y$ axis, and the number of columns reconstructed on the $x$ axis, for different retention probabilities (p= 0.2, 0.5 etc.). The selectivity of each predicate is set to 0.5. The condition number (which is independent of the dataset) increases as the retention probability decreases and increases exponentially as the number of columns reconstructed increase. The condition number is a good indicator of the reconstruction error by the inversion algorithm [14], and by the iterative Bayesian algorithm at small error values. Unlike the continuous exponential growth in error as the number of reconstructed columns increases for the inversion algorithm, the error flattens out for the iterative algorithm, as it is bounded above by two as discussed earlier.

## 7.2 Scalability

Next we study, how the reconstruction error varies as the number of columns reconstructed, retention probability, number of rows, and selectivity of the predicates vary.

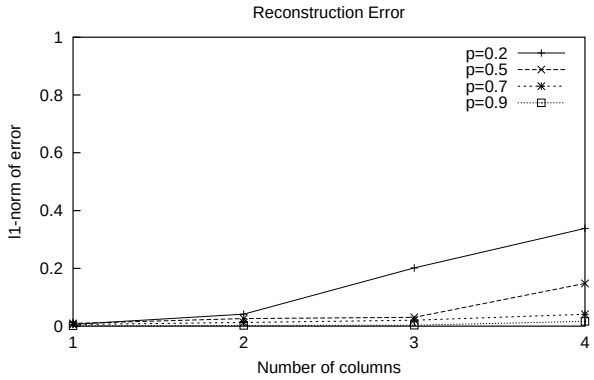


**Figure 7:** *Reconstruction errors for the Adult dataset for varying retention probabilities, p, by the iterative algorithm.*

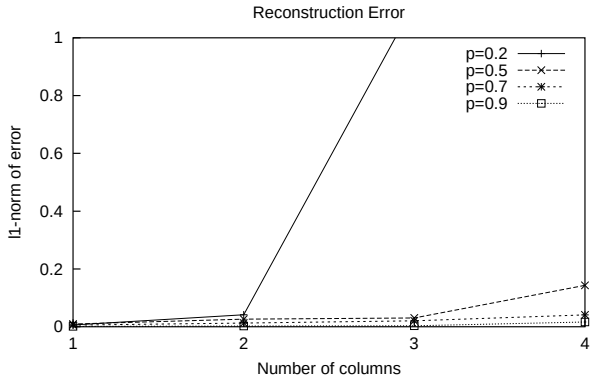


**Figure 8:** *Reconstruction errors for the Adult dataset for varying retention probabilities, p, by the inversion algorithm.*

**Number of columns and retention probability:** We study the reconstruction errors for varying number of columns and retention probabilities on the Adult dataset by the iterative and inversion algorithms. The predicates being reconstructed are age[ 25 - 45], fnlwgt[ 100000 - 1000000], hrsweek[ 30 - 60] and edulevel[ 5 - 10]. For the $i$ ( $1 \leq i \leq 4$ ) column experiment, the first $i$ among the above predicates are selected in the query. Figure 7 shows the reconstruction errors with the iterative algorithm, while Figure 8 shows the reconstruction errors with the inversion algorithm. Both iterative and inversion algorithms show an exponential increase in the error as the number of columns increases and as the probability of retention decreases. For smaller number of columns and higher retention probabilities both algorithms give comparable reconstruction errors. However for larger number of columns and lower retention probabilities the iterative algorithm gives smaller errors than the inversion algorithm. As explained in Section 7.1, unlike the iterative method, the reconstruction error by the inversion method can grow arbitrarily, whereas the error by the iterative method flattens out after an initial exponential increase.

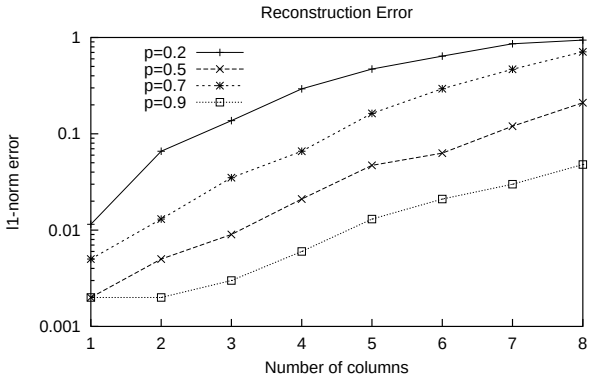


**Figure 9:** *Reconstruction error by iterative method on Zipfian dataset with* $10^5$ *rows varying number of columns*

For all experiments on the Zipfian dataset, the predicate on each column has an independent selectivity of 0.5. Figure 9 shows the reconstruction error after the iterative algorithm is applied to the perturbed Zipfian dataset of size $10^5$. The figure shows the increase in the reconstruction error, plotted on the $y$ axis, for increasing number of columns, plotted on the $x$ axis, for different retention probabilities. After an initial exponential increase, the reconstruction error flattens out.

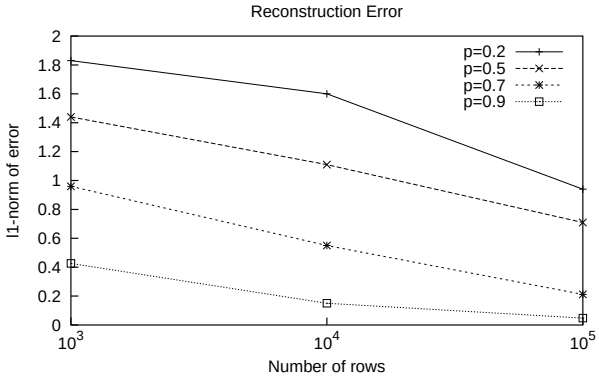


**Figure 10:** *Reconstruction error by iterative method on Zipfian dataset varying number of rows for 8 columns.*

**Number of rows in the table:** Figure 10 shows how the reconstruction error decreases as the number of perturbed rows available for reconstruction increase, for the the iterative reconstruction algorithm. In Figure 10 the retention probabilities are varied while the number of columns remains fixed at 8. For large values of $n$ the reconstruction error decreases as $n^{-0.5}$ as suggested by Theorem 1. This is also ratified by the factor 10 displacement between the reconstruction error lines for $10^3$ and $10^5$ rows in Figures 11 and 12. As the number of rows increases, it is possible to reconstruct more columns together at smaller retention probabilities.

**Selectivity of the predicates:** Recall that $e = x_{2^k-1}$ is estimate for the aggregate query and $a = x'_{2^k-1}$ is the actual answer for this query. $|e-a|$ is the called the absolute error while $|e-a|/a$ is called the relative error. Since we are interested in the variation of the

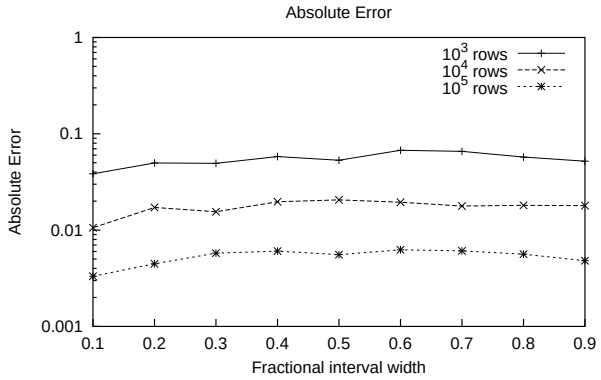


**Figure 11:** *Absolute Error for the Zipfian dataset for p=0.2 for varying interval sizes.*

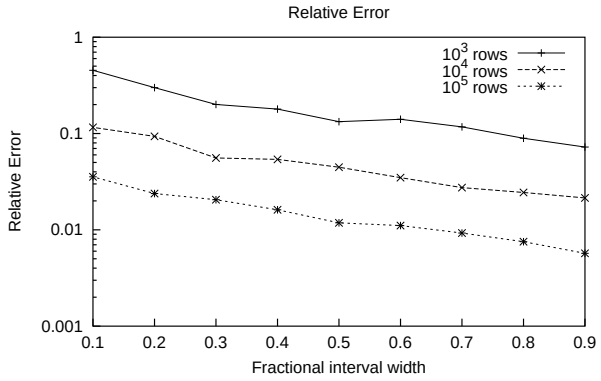


**Figure 12:** *Relative Error for the Zipfian dataset for p=0.2 for varying interval sizes.*

error in the aggregate query with the selectivity of its predicate, for this set of experiments, we use the absolute and relative errors, instead of the $l_1$ norm of the difference of the p.d.f.vectors, as the error metric.

For the experiments a single Zipfian column is used with uniform perturbation with retention probability $p = 0.2$. We vary the selectivity of the predicate of the numeric column by varying the size of the interval in the range predicate. Figure 11 and Figure 12 study the variation in absolute and relative errors respectively, as the size of the interval being queried changes. The fractional interval width, i.e. the ratio of the size of the interval being queried to the entire domain of the column, is plotted on the $x$ axis while the error is plotted on the $y$ axis. The absolute error in Figure 11 does not vary much with the interval width. However the relative error in Figure 12 increases as the interval width decreases. Both the absolute and relative errors decrease as the number of rows available for reconstruction increases.

## 7.3 Privacy Breach Guarantees

We study privacy breaches possible after perturbation on the Adult dataset. Figure 13 and Figure 14 show the maximum retention probability that avoids breaches for varying values of $\rho_1$ for fixed $\rho_2 = 0.95$, according to Theorem 5. To compute the values of $s$ for sample predicates (subsets) of this dataset, we divide each column into 10 equiwidth intervals and consider predicates

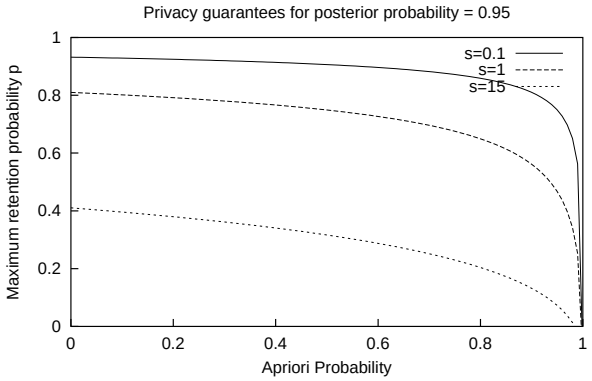


**Figure 13:** *Privacy for two columns for Adult data.*

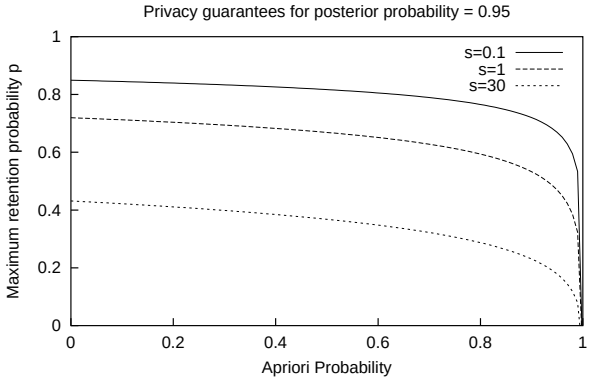


**Figure 14:** *Privacy for three columns for Adult data.*

that are subsets formed by the cross product of the intervals. Thus for two columns we consider $10^2$ subsets and for three columns we consider $10^3$ subsets. The maximum values of $s$ were observed to be 15 and 30 for two and three columns respectively. The median value of $s$ has been shown to be one in Theorem 3. The two figures plot the maximum retention probability, $p$, that would avoid a $(s, \rho_1, \rho_2)$ breach, on the $y$ axis against the a priori probability, $\rho_1$, on the $x$ axis for different values of relative a priori probability, s. The values of $s$ used are the maximum value of $s$, the median value $s = 1$, and $s = 0.1$ for a rare set. Both figures show that if it suffices to just hide rare properties (i.e., with $s \leq 0.1$), then for $\rho_1 > 0.5$, the retention probability $p$ can be as high as 0.8. If we need to hide all the above properties, i.e. even for the largest $s$ (the most common property), then for $\rho_1 > 0.5$ the retention probability can be selected to be as high as $p = 0.3$. For $p = 0.3$ both Figure 4 and Figure 5 show low reconstruction error. Thus reconstructability of 2 and 3 aggregates together, and privacy of data elements, are both achieved by perturbation for the Adult dataset, with $p = 0.3$. Thus our experiments indicate $(s, \rho_1, \rho_2)$-privacy as well as multi-column aggregate reconstructability.

# 8. SUMMARY AND FUTURE WORK

The contributions of the paper are:

- We introduce the problem of privacy preserving OLAP in a distributed environment.
- We introduce the formalism for reconstructible functions on a perturbed table, and develop algorithms to reconstruct multiple columns together. We provide privacy guarantees that take into account correlations between any combination of categorical and numeric columns.
- We provide two reconstruction algorithms to work with retention replacement perturbation: an iterative Bayesian algorithm, and a matrix inversion algorithm that also yields the maximum likelihood estimator. These algorithms can reconstruct count aggregates over subcubes without assuming independence between columns.
- We evaluate proposed reconstruction algorithms both analytically and empirically. We study the privacy guarantees we get for different levels of reconstruction accuracy and show the practicality of our techniques.
- We show the use of our techniques to related applications like classification.

Future work includes extending this work to other aggregates over subcubes.

**Acknowledgements**

We thank Rajeev Motwani and Rajat Raina for discussions on identity perturbation and maximum likelihood estimators. We also thank Alexandre Evfimievski and an anonymous reviewer for insightful comments on the paper.

# 9. REFERENCES

[1] N. R. Adam and J. C. Wortmann. Security control methods for statistical databases: A comparative study. In *ACM Computing Surveys, Vol21, No 4*, Dec. 1989.

[2] D. Agrawal and C. Aggarwal. On the design and quantification of privacy preserving datamining algorithms. In *Proc. of the 2001 ACM Symp. on Principles of Database Systems*.

[3] R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *Proc. of the 2003 ACM SIGMOD Intl. Conf. on Management of Data*.

[4] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the 2000 ACM SIGMOD Intl. Conf. on Management of Data*.

[5] C. Blake and C. Merz. UCI repository of machine learning databases, 1998.

[6] S. Chawla, C. Dwork, F. McSherry, H. Wee, and A. Smith. Towards privacy in public databases. In *Theory of Cryptography Conference*, 2005.

[7] H. Chernoff. Asymptotic efficiency for tests based on the sums of observations. In *Annals of Mathematical Statistics*, 1952.

[8] C. Clifton, M. Kantarcioglu, X. Lin, J. Vaidya, and M. Zhu. Tools for privacy preserving distributed data mining. *SIGKDD Explorations*, 4(2):28–34, Jan. 2003.

[9] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proc. of the 2003 ACM Symp. on Principles of Database Systems*.

[10] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proc. of the 2002 ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*.

[11] J. D. Ferrer and V. Torra. Disclosure risk assesment in statistical microdata protection via advanced record linkage. In *Statistics and Computing*, pages 343–354, 2003.

[12] M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *Proc. Advances in Cryptology – EUROCRYPT 2004*, 2004.

[13] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game – a completeness theorem for protocols with a honest majority. In *Proc. of the 1987 Annual ACM Symp. on Theory of Computing*.

[14] G. Golub and C. V. Loan. *Matrix computations*. John Hopkins Series in the Mathematical Sciences, 1996.

[15] K. Hoffman and R. Kunze. Linear algebra. Prentice-Hall Inc, 1971.

[16] B. A. Huberman, M. Franklin, and T. Hogg. Enhancing privacy and trust in electronic communities. In *Proc. of the 1st ACM Conference on Electronic Commerce*, 1999.

[17] C. A. J. Hurkens and S. R. Tiourine. Model and methods for the microdata protection problem. In *Journal of Official Statistics*, 1998.

[18] C. K. Liew, U. J. Choi, and C. J. Liew. A data distortion by probability distribution. *ACM Transactions on Database Systems*, 10(3), 1985.

[19] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *CRYPTO*, 2000.

[20] R. A. J. Moore. Controlled data-swapping techniques for masking public use microdata sets. In *SRD Report RR 96-04, US Bereau of Census*, 1996.

[21] S. Rizvi and J. R. Haritsa. Maintaining data privacy in association rule mining. In *Proc. of the 2002 Intl. Conf. on Very Large Data Bases*.

[22] L. Wang, S. Jajodia, and D. Wijesekera. Securing OLAP data cubes against privacy breaches. In *In Proc. of the 2004 IEEE Symposium on Security and Privacy*.

[23] L. Wang, D. Wijesekera, and S. Jajodia. Cardinality-based inference control in data cubes. In *Journal of Computer Security*, 2004.

[24] S. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Assoc.*, 60(309).

[25] A. Yao. How to generate and exchange secrets. In *Proc. of the 1986 Annual IEEE Symp. on Foundations of Computer Science*.

## 10. APPENDIX

`Theorem` 1. Let the fraction of rows in $[low, high]$ in the original table, $f$ be estimated by $f'$, then $f'$ is a $(n, \epsilon, \delta)$ estimator for $f$ if $n \geq 4\log(\frac{2}{\delta})(p\epsilon)^{-2}$

PROOF: Let $Y_i$ be the indicator variable for the event that the $i^{th}$ row ($1 \leq i \leq n$) is perturbed and the perturbed value falls within $[low, high]$. $Y_i$ are i.i.d. with $Pr[Y_i = 1] = (1-p)b = q$ (say), $Pr[Y_i = 0] = 1 - q$. Let $X_i$ be the indicator variable for the event that the $i^{th}$ row is not perturbed and it falls within $[low, high]$. Once again $X_i$ are i.i.d. with $Pr[X_i = 1] = pf = r$ (say), $Pr[X_i = 0] = 1 - r$. Let $Z_i$ be the indicator variable for the event that the $i^{th}$ randomized row falls in $[low, high]$. We have $Z_i = X_i + Y_i$, and $Pr[Z_i = 1] = q + r = t$ (say), and $Pr[Z_i = 0] = 1 - t$. Let $Z = \sum_{i=1}^{n} Z_i = n_r$, the number of randomized values in range $[low, high]$. Since $Z_i$'s are independent Bernoulli random variables, $0 \leq t \leq 1$ and $n \geq 4\log(\frac{2}{\delta})(p\epsilon)^{-2} \times t$, applying Chernoff bounds[7] we get

$$Pr[|Z - nt| > nt\theta] < 2e^{\frac{-nt\theta^2}{4}} \leq \delta \quad for \;\; \theta = \frac{p\epsilon}{t}$$

Thus with probability $> 1-\delta$, we have $-np\epsilon < Z - nt = n_r - n(pf + (1-p)b) < np\epsilon$, which implies

$$f - \epsilon < \frac{n_r}{pn} - \frac{(1-p)(high - low)}{p(max - min)} < f + \epsilon$$

Hence $|f - f'| < \epsilon$ with probability $> 1 - \delta$.

□

`Theorem` 2. The vector $x$ calculated as $A^{-1}y$ is the maximum likelihood estimator (MLE) of the relaxed a priori distribution [2] on the states that generated the perturbed table.

PROOF: Let $V = (v^1, v^2, v^3, ....v^n)$ be the observed state values for the $n$ perturbed rows in $T'$. Let $t = 2^k - 1$. Note that $v^i \in \{0, 1, 2, ...., t\}$ for all $i \in [1..n]$. If $L(x)$ is the likelihood of the observations, $V$, given a probability distribution on the states, $x$, in the original data, then $L(x) = Pr(V|x) = \prod_{i=1}^{n} Pr(v^i|x) = \prod_{i=1}^{n}(\frac{1}{n}\sum_{j=0}^{t} a_{jv^i}x_j) = \prod_{i=0}^{t}(\frac{1}{n}\sum_{j=0}^{t} a_{ji}x_j)^{y_i}$ (reordering according to the values of $v^i$.) Maximizing $L(x)$ is equivalent to maximizing $\log(L(x))$.

$$\max \log(L(x)) = \sum_{i=0}^{t}(y_i \times \log(\frac{1}{n}\sum_{j=0}^{t} x_j a_{ji}))$$

subject to the constraint $\sum_{j=0}^{t} x_j = n$.
This is equivalent to

$$\max_x \min_\lambda l(x, \lambda) = \sum_{i=0}^{t}(y_i \log(\frac{1}{n}\sum_{j=0}^{t} x_j a_{ji})) - \lambda(\sum_{j=0}^{t} x_j - n)$$

where $\lambda$ is the Lagrangian multiplier.

If $\sum_{j=0}^{t} x_j - n > 0$ then setting $\lambda$ to arbitrarily large positive value, you can minimize the term $-\lambda(\sum_{j=0}^{t} x_j - n)$ to an arbitrarily small negative number, similarly when $\sum_{j=0}^{t} x_j - n < 0$, as $\lambda$ tends to $-\infty$ the term becomes arbitrarily small. So the optimum ensures that the constraint $\sum_{j=0}^{t} x_j = n$ is satisfied.

To maximize the expression, setting the partial derivatives to be zero we get,

$$\frac{\partial l}{\partial x_s} = \sum_{i=0}^{t} y_i \frac{a_{si}}{\sum_{j=0}^{t} x_j a_{ji}} - \lambda = 0 \;\forall 0 \leq s \leq t$$

and $\frac{\partial l}{\partial \lambda} = \sum_{j=0}^{t} x_j - n = 0$.

Matrix $A$ is stochastic, i.e. $\sum_{i=0}^{t} a_{si} = 1 \;\forall 0 \leq s \leq t$, as they are probabilities of transition out of a state. Consider the row vector, $x = yA^{-1}$ calculated by the algorithm. For this vector $x$, $\sum_{j=0}^{t} x_j a_{ji} = y_i$. Hence substituting above, $\frac{\partial l}{\partial x_s} = \sum_{i=0}^{t} y_i \frac{a_{si}}{y_i} - \lambda = \sum_{i=0}^{t} a_{si} - \lambda = 1 - \lambda$

Also $\sum_{j=0}^{t} y_j = \sum_{j=0}^{t}\sum_{i=0}^{t} x_i a_{ij} = \sum_{i=0}^{t}\sum_{j=0}^{t} x_i a_{ij}$
$= \sum_{i=0}^{t} x_i \sum_{j=0}^{t} a_{ij} = \sum_{i=0}^{t} x_i$. As $\sum_{j=0}^{t} y_j = n$ we have $\sum_{i=0}^{t} x_i = n$, satisfying $\frac{\partial l}{\partial \lambda} = 0$.

Thus at $x$, given by $x = yA^{-1}$, and $\lambda = 1$ we get a local maximum of $l(x, \lambda)$. We show that the local maximum is the global maximum,

[2] i.e. $\sum_i x_i = n$ and $0 \leq x_i \leq n$ are the exact constraints, the relaxed constraint only ensures $\sum_i x_i = n$

by analyzing the Hessian matrix, $H$, of $l(x,\lambda)$ and showing $x^T Hx \leq 0$, for all $x \in R^t$. Elements of H are given by,

$$h_{si} = \frac{\partial l}{\partial x_s \partial x_i} = -\sum_{h=1}^{m} y_h \frac{a_{ih}a_{sh}}{(\sum_{j=0}^{t} x_j a_{jh})^2} \forall 0 \leq s,i \leq m$$

.

$$\begin{aligned} x^T Hx &= \sum_{i=0}^{t}\sum_{s=0}^{t} h_{si}x_s x_i \\ &= \sum_{i=0}^{t}\sum_{s=0}^{t} -\sum_{h=0}^{t} y_h \frac{a_{ih}a_{sh}}{(\sum_{j=0}^{t} x_j a_{jh})^2} x_s x_i \\ &= -\sum_{i=0}^{t}\sum_{s=0}^{t}\sum_{h=0}^{t} \phi_h a_{ih}a_{sh}x_i x_s \end{aligned}$$

where

$$\phi_h = \frac{y_h}{(\sum_{j=0}^{t} x_j a_{jh})^2} \geq 0$$

Thus

$$\begin{aligned} x^T Hx &= -\sum_{h=0}^{t}\phi_h \sum_{i=0}^{t}\sum_{s=0}^{t} a_{ih}a_{sh}x_i x_s \\ &= -\sum_{h=0}^{t}\phi_h \sum_{i=0}^{t} a_{ih}x_i (\sum_{s=0}^{t} a_{sh}x_s) \\ &= -\sum_{h=0}^{t}\phi_h (\sum_{s=0}^{t} a_{sh}x_s)\sum_{i=0}^{t} a_{ih}x_i \\ &= -\sum_{h=0}^{t}\phi_h (\sum_{s=0}^{t} a_{sh}x_s)(\sum_{i=0}^{t} a_{ih}x_i) \\ &= -\sum_{h=0}^{t}\phi_h (\sum_{i=0}^{t} a_{ih}x_i)^2 \leq 0 \end{aligned}$$

□

Theorem 3. The median value of relative a priori probability, over all subsets $S$, $S \subseteq V_X$, is 1.

PROOF: Consider, any subset $S \subseteq V_X$, and $\overline{S} = V_X - S$. Using notation as in Definition 5 we have $p_s + p_{\overline{s}} = 1$ and $m_s + m_{\overline{s}} = 1$. Hence if $p_s/m_s \geq 1$ , we have $p_{\overline{s}}/m_{\overline{s}} \leq 1$ and if $p_s/m_s < 1$ we have $p_{\overline{s}}/m_{\overline{s}} > 1$ Since this is true for any pair of complementary subsets, among all subsets of $V_X$, half the subsets have relative a priori probability $\geq 1$ and half $\leq 1$. Hence the median value of $s$ over all subsets of $V_X$ will be 1, if the median is not constrained to be one of the values attained.

□

Theorem 4. Let $p$ be the probability of retention, then uniform perturbation applied to a single column is secure against a $(s,\rho_1,\rho_2)$ breach, if

$$s < \frac{(\rho_2 - \rho_1)(1-p)}{(1-\rho_2)p}$$

PROOF: Let $S \subseteq V_X$ with $P[S] = p_s$ according to the a priori distribution and $P[S] = m_s$ according to the replacing p.d.f. Let $X$ and $Y$ denote the random variables for the original and perturbed value respectively. Let $R$ denote the event that $X$ was replaced and $R^c$ it being retained. For a $(\rho_1,\rho_2)$ privacy breach with respect to $S$ we need $P[X \in S] \leq \rho_1$. Also $P[(X \in S)|(Y \in S)]$

$$= \frac{P[(X \in S) \cap (Y \in S) \cap R] + P[(X \in S) \cap (Y \in S) \cap R^c]}{P[Y \in S]}$$

$$\leq \frac{\rho_1(1-p)m_s + pp_s}{(1-p)m_s + pp_s}$$

This is because $P[(X \in S) \cap (Y \in S) \cap R] = P[X \in S]P[Y \in S|R]P[R] \leq \rho_1(1-p)m_s$ and $P[(X \in S) \cap (Y \in S) \cap R^c] = P[(X \in S) \cap R^c] = pp_s$. Thus if $P[X \in S|Y \in S] \geq \rho_2$,

$$\frac{\rho_1(1-p)m_s + pp_s}{(1-p)m_s + pp_s} \geq \rho_2$$

Hence for a $(\rho_1,\rho_2)$ privacy breach with respect to $S$, we need

$$\frac{p_s}{m_s} \geq \frac{(\rho_2 - \rho_1)(1-p)}{(1-\rho_2)p}$$

□

Theorem 5. There will not be a $(\rho_1,\rho_2)$ privacy breach for $(S_1 \times S_2 \times \ldots S_k) = S \subseteq D$, if

$$\frac{p_s}{m_s} < \frac{\rho_2(1-\rho_1)(1-p)^k}{(1-\rho_2)\prod_{i=1}^{k}((1-p)m_{S_i} + p)}$$

PROOF: Let $X = (X_1, X_2, ..., X_k)$ be the random variable corresponding to the original value of the $k$ column row from the a priori distribution on table $T$, and $Y = (Y_1, Y_2, ....Y_k)$ that corresponding to the perturbed row, where each column is perturbed independently by a retention replacement perturbation. For $A_i, B_i \subseteq D_i$ we have $P[Y_i \in B_i | X_i \in A_i] = (1-p)m_{B_i} + p\frac{p_{A_i \cap B_i}}{p_{A_i}}$, for $1 \leq i \leq k$. Thus $(1-p)m_{B_i} \leq P[Y_i \in B_i | X_i \in A_i] \leq (1-p)m_{B_i} + p$. Thus for $A, B \subseteq D$ we have $L_B$ (say) $= \prod_{i=1}^{k}(1-p)m_{B_i} \leq P[Y \in B | X \in A] \leq \prod_{i=1}^{k}((1-p)m_{B_i} + p) = U_B$ (say). $P[(X \in S)|(Y \in S)] =$

$$\frac{P[(Y \in S)|(X \in S)]P[X \in S]}{P[(Y \in S)|(X \in S)]P[X \in S] + P[(Y \in S)|\neg(X \in S)]P[\neg(X \in S)]}$$

$$\leq \frac{U_S p_s}{U_S p_s + L_S(1-p_s)}$$

Suppose there is a $(\rho_1,\rho_2)$ privacy breach with respect to $S$, we need $P[X \in S] \leq \rho_1$, and $P[(X \in S)|(Y \in S)] \geq \rho_2$ Thus

$$\frac{U_S p_s}{U_S p_s + L_S(1-p_s)} \geq \rho_2$$

This implies

$$\frac{p_s}{L_S(1-p_s)} \geq \frac{\rho_2}{U_S(1-\rho_2)}$$

Substituting values of $U_S$, $L_S$ and noting that $p_s \leq \rho_1$ hence $1-p_s \geq 1-\rho_1$, we get

$$\frac{p_s}{\prod_{i=1}^{k} m_{S_i}} \geq \frac{\rho_2(1-\rho_1)(1-p)^k}{(1-\rho_2)\prod_{i=1}^{k}((1-p)m_{S_i} + p)}$$

□