# Optimal Random Perturbation at Multiple Privacy Levels

Xiaokui Xiao, Yufei Tao, Minghua Chen

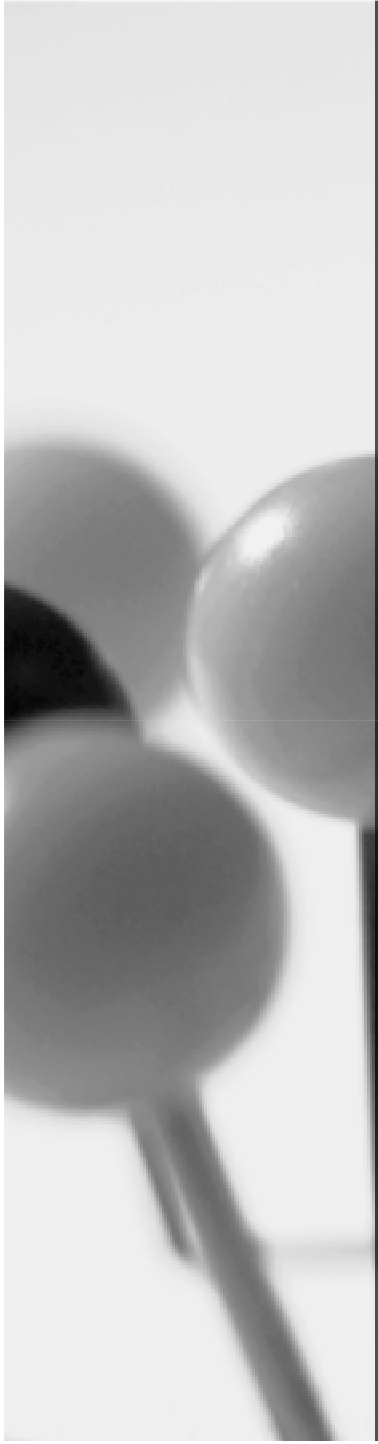
In Proc. International Conferece on Very Large Data Bases 2009 (VLDB'09)

Presented by

Amiya Kumar Maji

# Motivation

- Existing randomization schemes perturb data at one privacy level
- Need to have multiple privacy levels
  - Govt. organization may require data with high usability and low privacy
  - Private organizations may have more perturbed data
  - May define a cost model based on perturbation level
- Naïve Solution
  - Perturb each version of data independently
  - Problem of collusion



# Uniform Perturbation

- Original dataset D, perturbed data D*
- D* retains all non-sensitive values in D
- For every sensitive value x in D perturb as

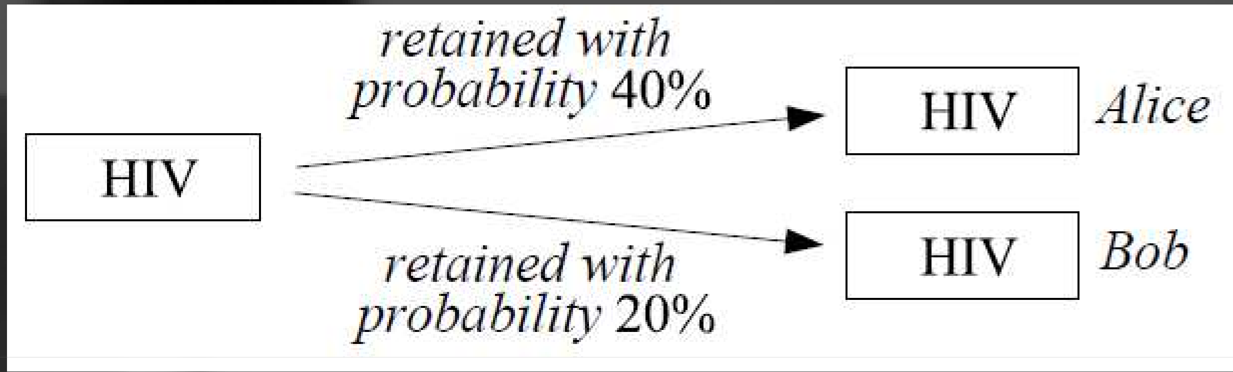
**Algorithm** ***uni-pert*** $(x, p)$

/* $x$ is the value being perturbed, and $p$ the retention probability */

1. toss a coin with head probability $p$
2. if the coin heads then return $x$
3. else return a random value in the domain of $x$

- p = retention probability
- If p = 1, then D = D*
- If p = 0, then all sensitive values are randomized in D*

# Problem with Independent Perturbation




- Each value perturbed independently
- Chances of both independently perturbed values to be HIV is small
- Original value is HIV with high confidence
- Pr[Both Alice and Bob gets HIV | original disease not HIV] is less than 1%

# Contributions

- Present a multi-level uniform perturbation with two properties
  - The confidence about original value is no more than the most trusted recipient (valid for any number of colluding parties)
  - Each recipient’s data can be considered as an application of uni-pert with its retention probability
- Consumes O(n+m) expected space
- Produces a perturbed version in O(n+log m) time
- n = no. of tuples in D, m = no. of versions

# Preliminaries

- X: a random variable denoting original value
- Y: a random variable denoting perturbed value
- X, Y distribute in a domain DOM
- |DOM| = s
- p = retention probability
- For x, y in DOM

$$Pr[Y = y | X = x] = \begin{cases} p + (1-p)/s & \text{if } x = y \\ (1-p)/s & \text{if } x \neq y \end{cases}$$

# Privacy Guarantees

- Uniform perturbation guarantees
  - $\rho_1$-$\rho_2$ privacy
- Let, Q(X) be a predicate on X
- Pr[Q(X)] = adversary's (prior) belief in Q(X)
- Pr[Q(X) | Y] = adversary's belief in Q(X) after observing Y
- $\rho_1$-$\rho_2$ privacy requires

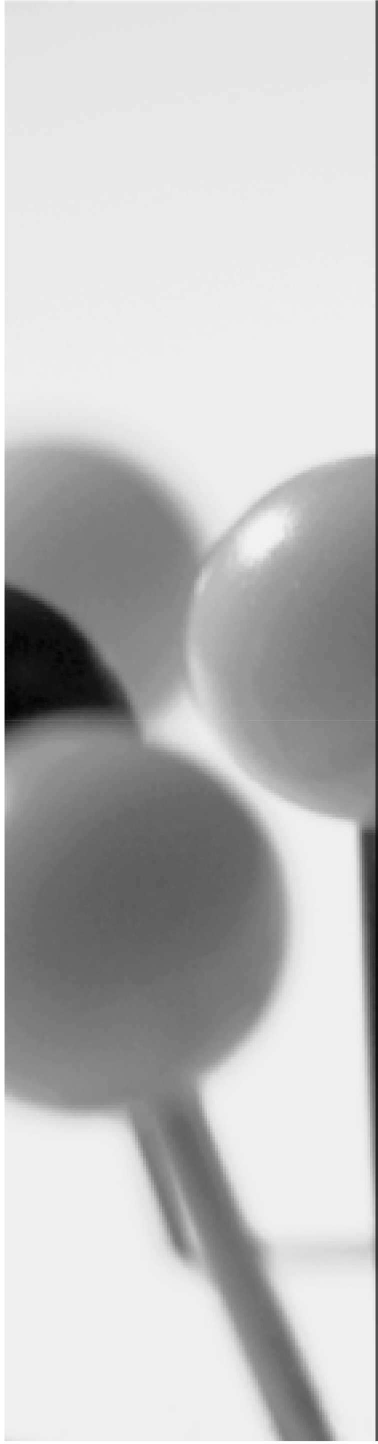
$$
\begin{aligned}
& Pr[Q(X)] < \rho_1 \Longrightarrow Pr[Q(X) \mid Y] < \rho_2, \\
\text{and} \quad & Pr[Q(X)] > \rho_2 \Longrightarrow Pr[Q(X) \mid Y] > \rho_1.
\end{aligned}
$$

# Problem Definition

| Symbol | Description |
|---|---|
| $D$ | The original dataset |
| $A$ | The sensitive attribute of $D$ |
| $B$ | The set of non-sensitive attributes of $D$ |
| $DOM$ | The domain of $A$ |
| $s$ | The size of $DOM$ |
| $n$ | The cardinality of $D$ |
| $H$ | The set of recipients we responded to before |
| $m$ | The size of $H$ |
| $p_i$ $(1 \leq i \leq m)$ | The $i$-th highest retention probability of the recipients in $H$ |
| $D_i^*$ $(1 \leq i \leq m)$ | The perturbed version of $D$ returned to the recipient with retention probability $p_i$ |
| $p$ | The retention probability of the incoming request |

# Contd.

- Let, t: an arbitrary tuple in D
- X: r.v. denoting the sensitive value in t
- $S_{share}$: Set of colluding recipients
- L: Set of perturbed values of X
- best(L): value in L that is most authentic
- H: set of all recipients that we have responded to
- $|H| \geq 1$



# Problem Definition

- Given a new request with retention probalility p, return a perturbed dataset D* of D where every tuple t* corresponds to a tuple t in D such that

1. t* keeps all the non-sensitive values in t
2. If Y is the r.v. denoting the perturbed version of X, then distribution of Y is given by

$$Pr[Y = y|X = x] = \begin{cases} p + (1-p)/s & \text{if } x = y \\ (1-p)/s & \text{if } x \neq y \end{cases}$$

# Contd.

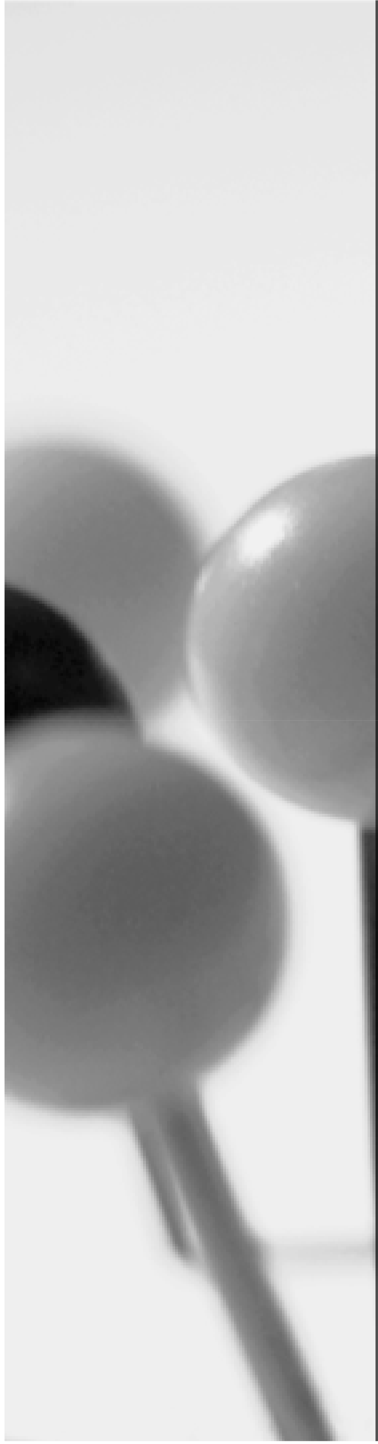
3. If L is a non-empty subset of all perturbed values of t we returned (including the current recipient) then we can guarantee

$$Pr[Q(X)|L] = Pr[Q(X)|best(L)]$$

# Multi-level Uniform Perturbation

- Let, m be the size of H
- $p_1, p_2, .., p_m$ are retention probabilities of recipients in H in non-ascending order
- $D_i$* is the anonymized version of D with retention probability $p_i$
- Need to compute D* with p
- p is different from $p_1, p_2, .., p_m$
- D* must be derived from $D_1$*, $D_2$*, .., $D_m$*

- Let $p_l$ is the smallest probability in $\{p_1, p_2, .., p_m\}$ larger than p
- $p_r$ is the largest probability in $\{p_1, p_2, .., p_m\}$ smaller than p
- If $p_l$ does not exist, set $p_l=1$
- If $p_r$ does not exist, $p_r$ is undefined
- $D_l^*$, $D_r^*$ are the data sets corresponding to $p_l$, and $p_r$
- $D^*$ can be computed from $D_l^*$, $D_r^*$



# Algorithm

**Algorithm** ***multi-pert*** $(p)$

/* $p$ is the retention probability of a new request */

1. let $p_1, p_2, ..., p_m$ be the retention probabilities of the previous requests in non-ascending order
2. if $p$ equals $p_i$ for any $i \in [1, m]$, then return $D_i^*$
3. $l$ = the largest subscript $i \in [1, m]$ such that $p_i > p$
   /* $p_l$ = the lowest of $p_1, p_2, ..., p_m$ greater than $p$ */
4. if $p_l$ does not exist then $p_l = 1$ and $D_l^* = D$
5. $r$ = the smallest subscript $i \in [1, m]$ such that $p_i < p$
   /* $p_r$ = the greatest of $p_1, p_2, ..., p_m$ lower than $p$ */
6. if $p_r$ does not exist
7.     for each tuple $t_l \in D_l^*$
8.         create a tuple $t^*$ in $D^*$ with $t^*[B] = t_l[B]$
           /* $B$ is the set of non-sensitive attributes */
9.         set $t^*[A]$ to $t_l[A]$ with probability $p/p_l$, or to a random value in $DOM$ with probability $1 - p/p_l$
           /* $A$ is the sensitive attribute */

# Contd.

10. else
11. for each tuple $t_l \in D_l^*$
12. identify its matching tuple $t_r \in D_r^*$
13. create a tuple $t^*$ in $D^*$ with $t^*[B] = t_l[B]$
14. set $t^*[A]$ to $t_l[A]$ with probability $u$, to $t_r[A]$ with probability $v$, or to a random value in $DOM$ with probability $1 - u - v$, where $u, v$ are given in Equations 3 and 4, respectively
15. return $D^*$

$$u = \begin{cases} p/p_l & \text{if } y_l = y_r \\ (p - p_r)/(p_l - p_r) & \text{if } y_l \neq y_r \end{cases}$$

$$v = \begin{cases} (1 - \frac{p}{p_l})(1 - \frac{1-p_r/p}{(s-1)p_r/p_l+1}) & \text{if } y_l = y_r \\ \frac{p_r(p_l-p)}{p(p_l-p_r)} & \text{if } y_l \neq y_r \end{cases}$$

# Example

- Assume D has a single sensitive attribute x=HIV
- DOM is domain of diseases with |DOM|=10
- Alice request perturbed data with probability $p_1$=40%
- Assume HIV is retained in Alice's data set
- H contain Alice and value of $p_1$
- Bob requests data with p=20%
- $P_r$ = undefined, $p_l$ = 40%
- $p/p_l$ = 50%
- Retain Alice's value with 50% probability

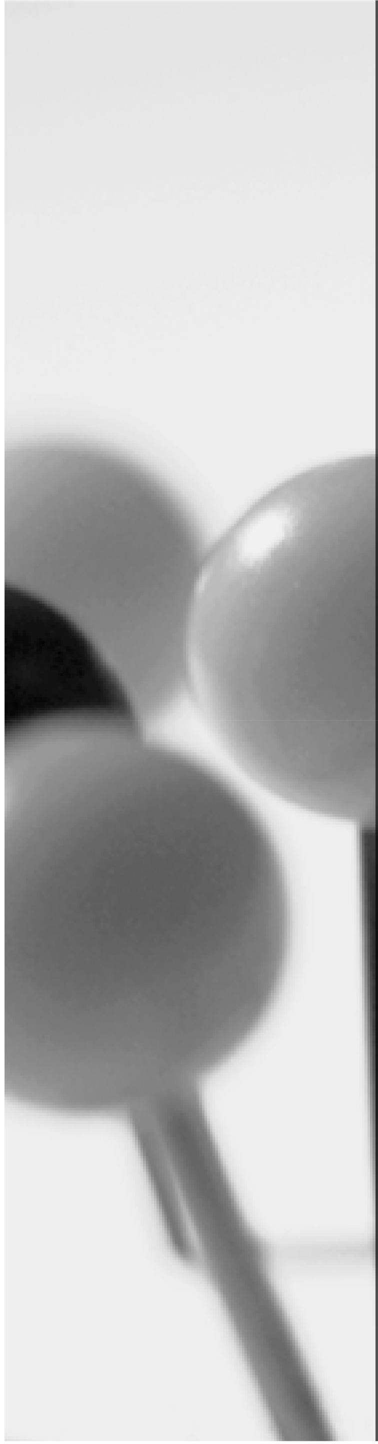
# Contd.

- Verify requirements 2, and 3 in problem definition
- y for Bob is solely computed from Alice's value, hence 3 is satisfied
- Compute Pr[Y = HIV | X = HIV] for Bob
- 3 cases

  I. Alice receives HIV and the coin we toss for Bob heads

$$[0.4 + (1 - 0.4)/10] * 0.5 = 0.23$$

- 0.4: Alice's coin heads
- (1 - 0.4)/10: Alice's coin tails, random disease selected is HIV



# Contd.

II. Alice receives HIV, coin for Bob tails, and the random value drawn from DOM is HIV

$0.46 * 0.5 * 0.1 = 0.023$

III. Alice doesn't receive HIV, coin for Bob tails, and the random value selected is HIV

$(1 - 0.46) * 0.5 * 0.1 = 0.027$

- Pr[Y=HIV | X=HIV] = 0.23 + 0.023 + 0.027 = 0.28
- Consider uni-pert with X = x = HIV
- For Bob, p = 20%
- Using uni-pert

Pr[Y=HIV | X=HIV] = 0.2 + (1 – 0.2) * 0.1 = 0.28

# Derivation of u, v

- Recall $p_l$, $p_r$ are probabilities s.t. $p_l > p_{new} > p_r$
- Let $y_l$, $y_r$ are the perturbed values for $p_l$, $p_r$
- When $y_l = y_r$
  - Pr[head] = $u_1$, Pr[tail] = $v_1$
- When $y_l$ != $y_r$
  - Pr[head] = $u_2$, Pr[tail] = $v_2$
- Let $Y_a$, $Y_b$ be the r.v. corresponding to the perturbed values for Alice and Bob respectively
- $p_a = 40\%$, $p_b = 80\%$

- The algorithm requires

$$Pr[Q(X)|Y_a = y_a, Y_b = y_b] = Pr[Q(x)|Y_b = y_b]$$

$$Pr[Y_b = y_b|X = x] = \begin{cases} p_b + (1 - p_b)/s & \text{if } x = y_b \\ (1 - p_b)/s & \text{if } x \neq y_b \end{cases}$$

- Both are satisfied when

$$Pr[Y_b = y_b|Y_a = y_a, X = x]$$
$$= \begin{cases} \frac{(p_b+\frac{1}{s}(1-p_b))(p_a/p_b+\frac{1}{s}(1-p_a/p_b))}{p_a+\frac{1}{s}(1-p_a)} & \text{if } y_a = y_b = x \\ \frac{(1-p_b)(1-p_a/p_b)}{s^2(p_a+\frac{1}{s}(1-p_a))} & \text{if } y_a = x \neq y_b \\ \frac{1-p_b}{1-p_a}(p_a/p_b + \frac{1}{s}(1 - p_a)) & \text{if } y_a = y_b \neq x \\ \frac{1-p_a/p_b}{1-p_a}(p_b + \frac{1}{s}(1 - p_b)) & \text{if } x = y_b \neq y_a \\ \frac{(1-p_b)(1-p_a/p_b)}{s(1-p_a)} & \text{otherwise} \end{cases}$$

- Constitute equations for $u_1$, $v_1$, $u_2$, $v_2$ from these cases
- Solve for $u_1$, $v_1$, $u_2$, $v_2$

# Theoretical Analysis

- Lemma 1:

  For any i in {1, .., m} we have

$$Pr[Y_i = y_i | Y_0 = y_0, Y_1 = y_1, ..., Y_{i-1} = y_{i-1}] \\ = Pr[Y_i = y_i | Y_{i-1} = y_{i-1}],$$

  and

$$Pr[Y_i = y_i | Y_{i-1} = y_{i-1}] \\ = \begin{cases} \frac{p_i}{p_{i-1}} + \left(1 - \frac{p_i}{p_{i-1}}\right)/s & \textit{if } y_i = y_{i-1} \\ \left(1 - \frac{p_i}{p_{i-1}}\right)/s & \textit{if } y_i \neq y_{i-1} \end{cases}$$

# Contd.

- Theorem 1:

  Collusion is useless. For any subset L of {$Y_1$=$y_1$, $Y_2$=$y_2$, .., $Y_m$=$y_m$} we have

$$Pr[Q(X)|L] = Pr[Q(X)|best(L)]$$

- Theorem 2:

  For all recipient i in 1 ≤ i ≤ n, $Y_i$ is statistically same as the output of uni-pert, i.e.,

$$Pr[Y_i = y_i | X = x] = \begin{cases} p_i + (1-p_i)/s & \text{if } x = y_i \\ (1-p_i)/s & \text{if } x \neq y_i \end{cases}$$

# Minimizing Space and Time

- Naïve approach
- Let |H| = m
- For each sensitive value x store all the m released values
- Computation cost:
  - O(log m) to find l, r
  - O(n) to perturb
- Space overhead:
  - O(n*m)

# Efficient Implementation

- Notice that many consecutive values in $y_1, y_2, .., y_m$ are same
- We only need to save when y values change
- $Y_1, Y_2, .., Y_m$ make m-1 consecutive pairs $(Y_1, Y_2), (Y_2, Y_3), .., (Y_{m-1}, Y_m)$
- A pair is disparate if $(Y_{i-1}, Y_i)$ are different
- Let disp(t) = no. of disparate pairs in history
- Lemma 2:

  $E[disp(t)] < \ln(1/c)$,

  c is a constant such that $1 \geq p_1 \geq p_2 \geq .. \geq p_m \geq c$

# Contd.

- Save the list of probabilities $p_1, p_2, .., p_m$
- Build a list history(t) where each element has form

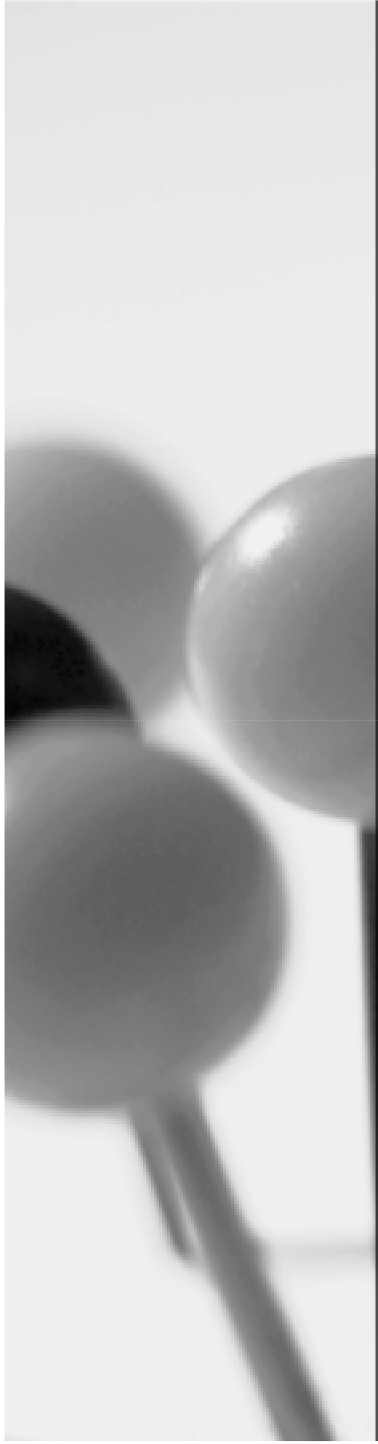
  <p, Y>
- Space complexity: O(n + m)
- To compute new perturbed version find $p_l$, $p_r$ in O(log m) time
- To retrieve $y_i$ for $p_i$
  - Find the smallest probability $p_j \geq p_i$
  - Return $y_j$
- Time complexity: O(n + log m)

# Experiments

- Verify the following experimentally
  - Ineffectiveness of collusion
  - Equivalence to uniform perturbation
  - Failure of independent perturbation
  - Space and computation cost

# Parameters

- Let X denote the original sensitive value
- $Y_a$, $Y_b$, $Y_c$ are three perturbed versions
- $p_a$=30%, $p_b$=10%, $p_c$=50%
- Set X as uniform dist, gaussian dist, salary dist, or occupation dist
- Compute $y_a$, $y_b$, $y_c$ for each X=x
- Prepare a 4D array F[X, $Y_a$, $Y_b$, $Y_c$] with all cells initially 0
- Run simulation $10^{10}$ times



- Collusion is ineffective
  - We must show

$$Pr[X = x | Y_a = y_a, Y_b = y_b, Y_c = y_c] = Pr[X = x | Y_c = y_c]$$

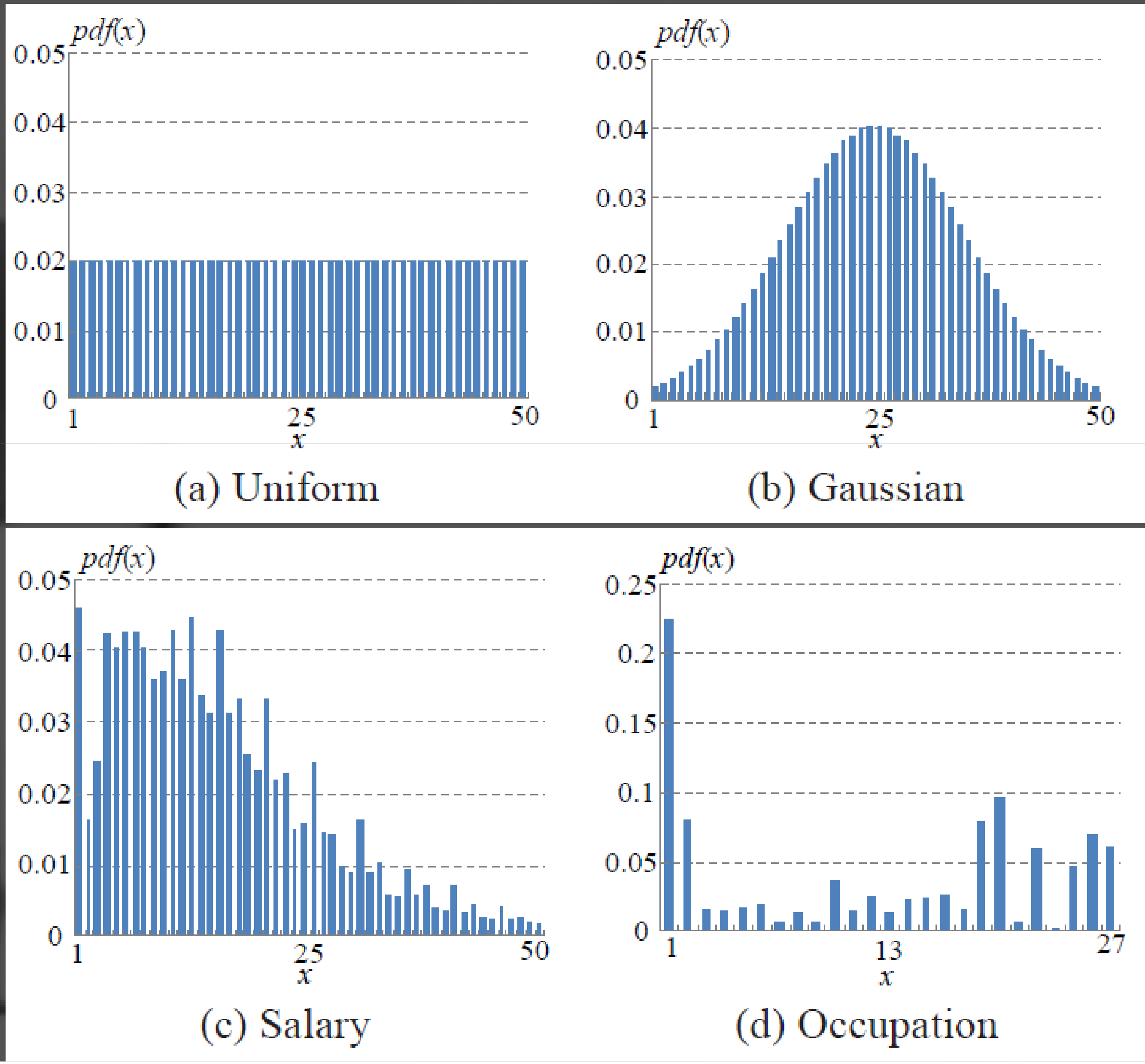
- Compute $Pr[X=x \mid Y_a=y_a, Y_b=y_b, Y_c=y_c]$ as

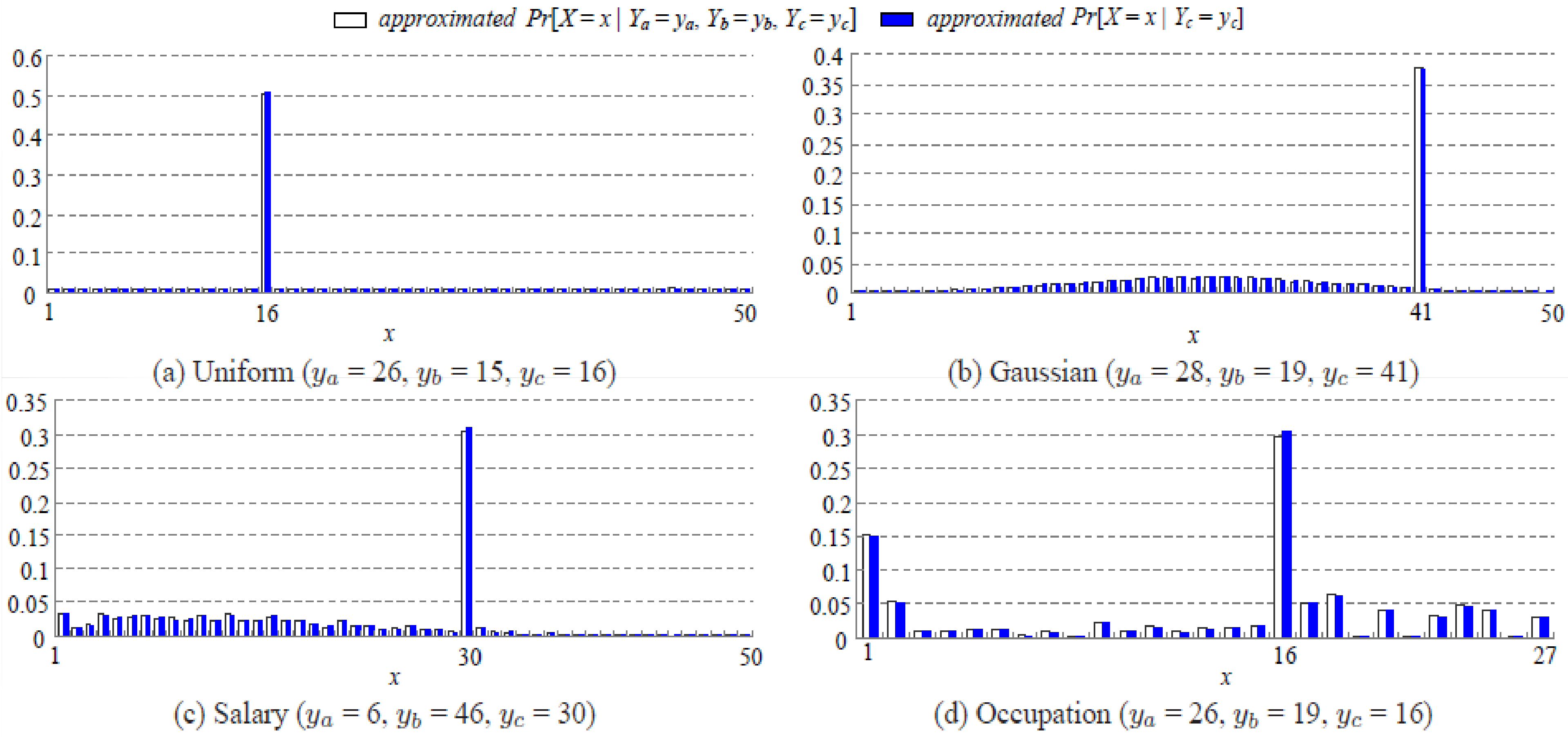
$$\frac{F[x, y_a, y_b, y_c]}{\sum_{\forall x'} F[x', y_a, y_b, y_c]}$$

- Compute $Pr[X=x \mid Y_c=y_c]$ as

$$\frac{\sum_{\forall y'_a, y'_b} F[x, y'_a, y'_b, y_c]}{\sum_{\forall x', y'_a, y'_b} F[x', y'_a, y'_b, y_c]}$$

# Distribution of Sensitive Values






(a) Uniform ($y_a = 26$, $y_b = 15$, $y_c = 16$)

(b) Gaussian ($y_a = 28$, $y_b = 19$, $y_c = 41$)

(c) Salary ($y_a = 6$, $y_b = 46$, $y_c = 30$)

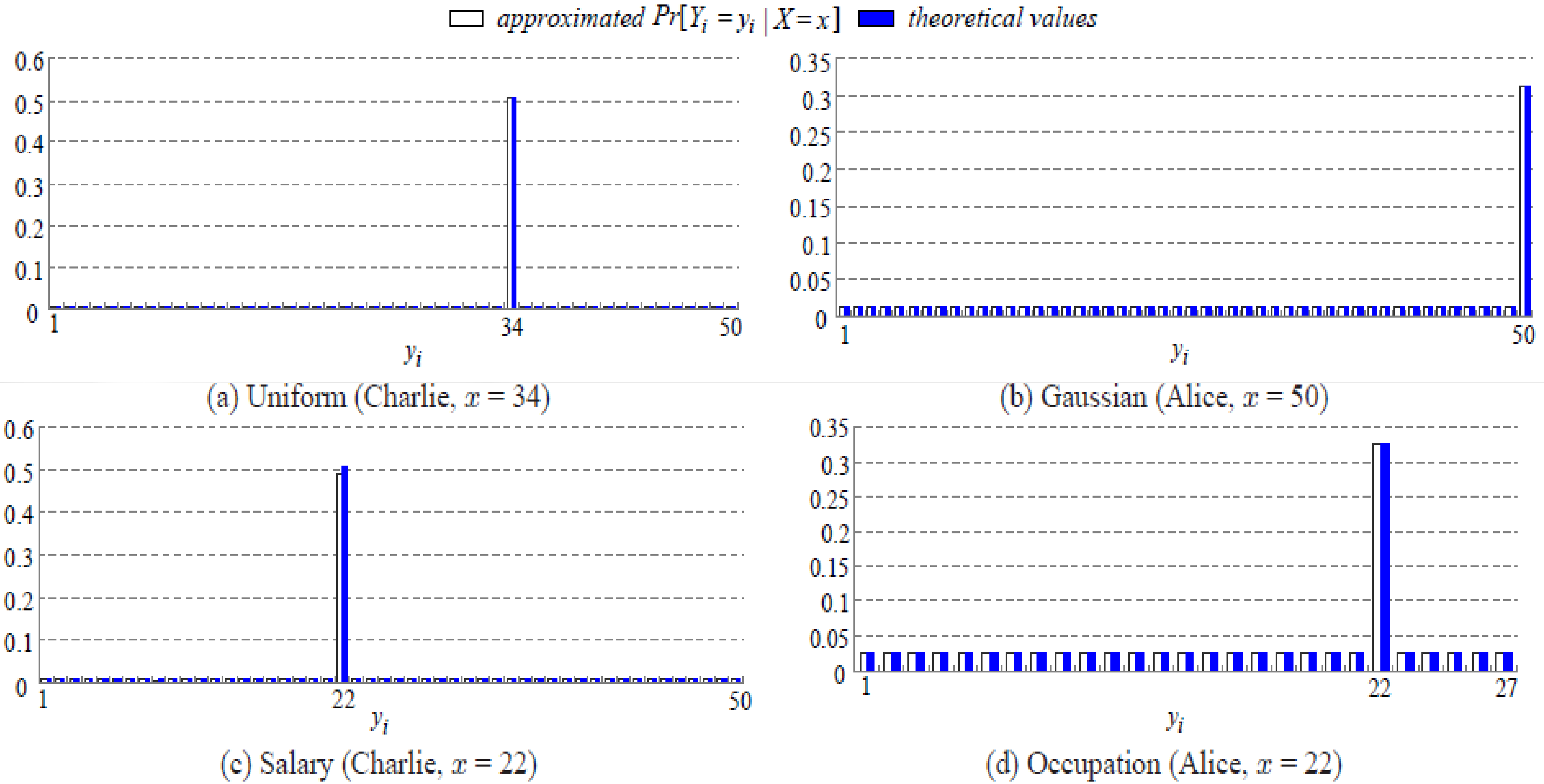
(d) Occupation ($y_a = 26$, $y_b = 19$, $y_c = 16$)

- Equivalence to uni-pert
  - Need to show

$$Pr[Y_i = y_i | X = x] = \begin{cases} p_i + (1 - p_i)/s & \text{if } x = y_i \\ (1 - p_i)/s & \text{if } x \neq y_i \end{cases}$$

- Compute Pr[$Y_a$=$y_a$ | X=x] as

$$\frac{\sum_{\forall y'_b, y'_c} F[x, y_a, y'_b, y'_c]}{\sum_{\forall y'_a, y'_b, y'_c} F[x, y'_a, y'_b, y'_c]}$$


approximated $Pr[Y_i = y_i \mid X = x]$
theoretical values
0.6
0.5
0.4
0.3
0.2
0.1
0
1
34
50
$y_i$
(a) Uniform (Charlie, $x = 34$)
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
1
50
$y_i$
(b) Gaussian (Alice, $x = 50$)
1
22
50
$y_i$
(c) Salary (Charlie, $x = 22$)
1
22
27
$y_i$
(d) Occupation (Alice, $x = 22$)

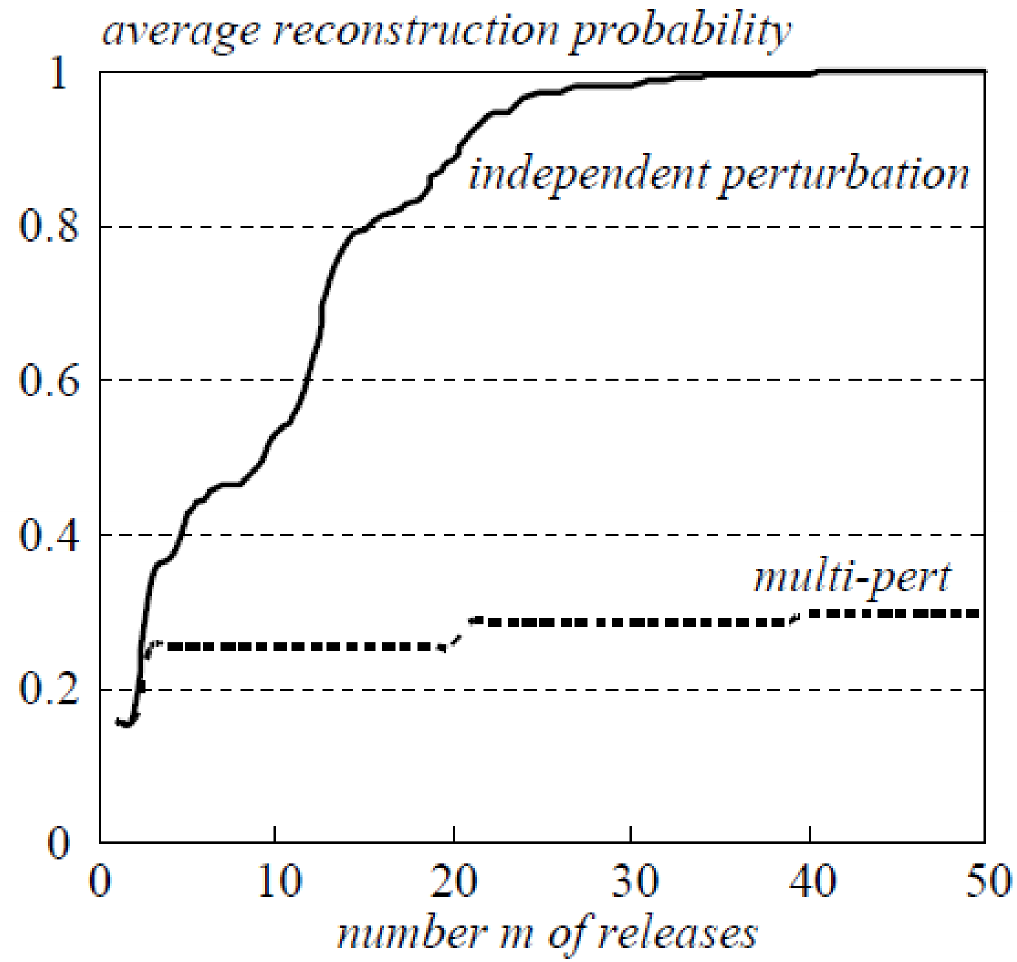
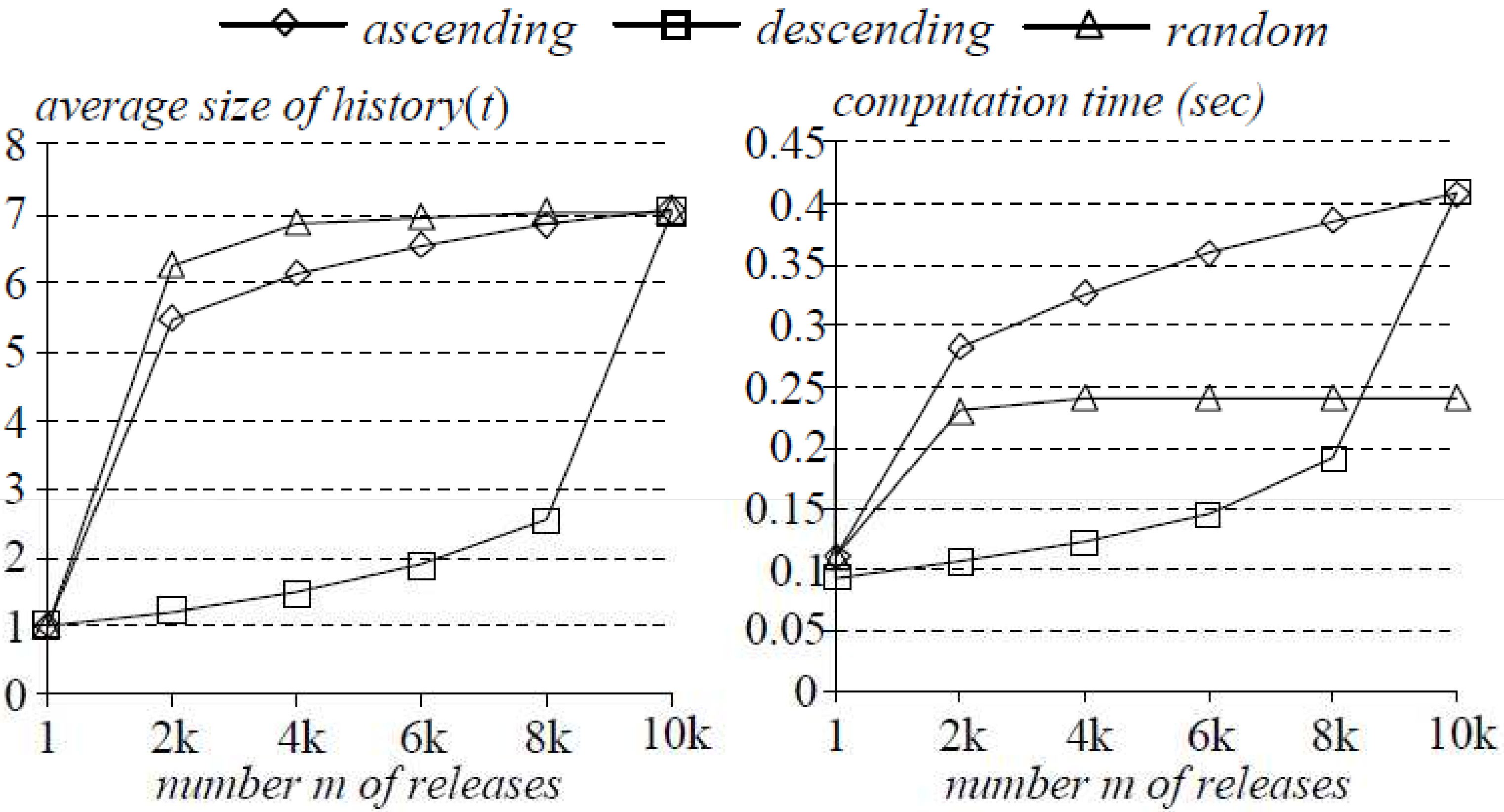



**Figure 7: Vulnerability of independent perturbation**




(a) Space vs. $m$ (b) Time vs. $m$

**Figure 8: Overhead of *multi-pert***

# Conclusion

- Allows us to compute multiple perturbed versions of data
- Protects against collusion
- Privacy (retention probabilities) of sensitive values may be specified in arbitrary order
- Expected space and time complexity are asymptotically optimal