Instant Anonymization

MEHMET ERCAN NERGIZ, Zirve University
ACAR TAMERSOY and YUCEL SAYGIN, Sabanci University

Anonymization-based privacy protection ensures that data cannot be traced back to individuals. Researchers working in this area have proposed a wide variety of anonymization algorithms, many of which require a considerable number of database accesses. This is a problem of efficiency, especially when the released data is subject to visualization or when the algorithm needs to be run many times to get an acceptable ratio of privacy/utility. In this paper, we present two instant anonymization algorithms for the privacy metrics k-anonymity and ℓ-diversity. Proposed algorithms minimize the number of data accesses by utilizing the summary structure already maintained by the database management system for query selectivity. Experiments on real data sets show that in most cases our algorithm produces an optimal anonymization, and it requires a single scan of data as opposed to hundreds of scans required by the state-of-the-art algorithms.

Categories and Subject Descriptors: H.2.8 [**Database Applications**]: Statistical Databases; K.4.1 [**Public Policy Issues**]: Privacy

General Terms: Algorithms, Security, Legal Aspects

Additional Key Words and Phrases: *k*-anonymity, *ell*-diversity, privacy, algorithms

ACM Reference Format:
Nergiz, M. E., Tamersoy, A., and Saygin, Y. 2011. Instant anonymization. ACM Trans. Datab. Syst. 36, 1, Article 2 (March 2011), 33 pages.
DOI = 10.1145/1929934.1929936 http://doi.acm.org/10.1145/1929934.1929936

1. INTRODUCTION

With the advance of technology, data collection and storage costs plummeted, which resulted in pervasive data collection efforts with the hope of turning this data into profit. If the data collector has the capacity to perform data analysis, then this could be done internally. However, in some cases, data needs to be outsourced for analysis or it may need to be published for research purposes like health related data in medical research. In order to preserve the privacy of individuals, data needs to be properly anonymized before publishing, which cannot be achieved by just removing personal identifiers. In fact, Samarati [2001] and Sweeney [2002] show that using publicly available sources of information such as age, gender, and zip code, data records can be reidentified accurately even if there is no direct personally identifying information in the dataset.

k-Anonymity was proposed as a standard for privacy protection, which requires that an individual should be indistinguishable from at least $k-1$ others in the anonymized dataset [Samarati 2001; Ciriani et al. 2007]. Two individuals are said to be indistinguishable if their records agree on the set of quasi-identifier attributes, which are not

This work is funded by the Scientific and Technical Research Council of Turkey (TUBITAK) National Young Researchers Career Development Program under grant number 106E116.
Author's address: M. E. Nergiz; email: mehmet.nergiz@zirve.edu.tr.

DOI 10.1145/1929934.1929936 http://doi.acm.org/10.1145/1929934.1929936

unique identifiers by themselves but may identify an individual when used in combination (e.g., age, address, nation, ...). Researchers working in this area have further proposed a wide variety of privacy metrics such as ℓ-diversity [Machanavajjhala et al. 2006] which overcomes the privacy leaks in k-anonymized tables due to lack of diversity in sensitive attribute (e.g., salary, GPA, ...) values. However, currently proposed techniques to achieve the desired privacy standard still require a considerable number of database accesses. This creates a problem especially for large datasets, and especially when response time is important. Indeed, to balance privacy vs. utility, the data releaser might need to run the anonymization algorithm many times with different parameters, might even need to visualize the outputs before deciding to release the best anonymization addressing the expectations. This arouses the need for efficient anonymization algorithms with acceptable utility guarantees.

In this paper, we present instant algorithms for k-anonymity and ℓ-diversity that require few data scans. We show that such an anonymization could be achieved by using a summary structure describing the data statistically. There are two ways to obtain such a summary structure. First, one can construct the summary structure by preprocessing the dataset. Preprocessing is done only once but should still be relatively efficient. As a representative for such summary structures, we use histograms that can be constructed with a single scan of data. Second, one can obtain the summary structure from the underlying database management system (DBMS). There exist summary structures that are maintained by DBMS mainly for query selectivity and are freely available for use to other applications as well. We use histograms and bayesian networks (BNs) as a case study to demonstrate the effectiveness of the proposed methods on real datasets. To the best of our knowledge this is the first work which utilizes statistical information for efficiently anonymizing large data sets.

The method we propose for instant anonymization has two phases: First, by using the summary structure, we build a set of candidate generalization mappings that have a high probability of satisfying the privacy constraints. Our methodology of calculating such a probability is the first statistical analysis of k-anonymity and ℓ-diversity given a generalization mapping. In this first phase, we work only on the summary structure which in most cases fits into memory, and we do not require any data accesses. Second, we apply the generalization mappings in the candidate set to the dataset until we find a mapping that satisfies the anonymity constraints. The performance of our technique depends on the candidate set, thus depends on the accuracy of our statistical analysis. Experimental results show that our algorithms greatly reduce the number of database accesses while producing an optimal or close to optimal anonymization, and in most cases they require a single scan of data as opposed to hundreds of scans required by the state-of-the-art algorithms.

The article is organized as follows: In Section 2, we give background and notations used in the article, followed by related work. In Sections 4 and 5, we show how to calculate the probability of achieving k-anonymity and ℓ-diversity given a mapping and a summary structure. In Section 6, we provide several heuristics to speed up the calculations. In Section 3, we present the instant anonymization algorithms for k-anonymity and ℓ-diversity. In Section 7, we report experimental results regarding the performance of our approach. In Section 8, we conclude with a discussion of future work in this area.

2. BACKGROUND AND RELATED WORK

2.1. Table Generalizations and Privacy Metrics

Given a dataset (table) T, $T[c][r]$ refers to the value of column c, row r of T. $T[c]$ refers to the projection of column c on T and $T[.][r]$ refers to selection of row r on T. We write $|t \in T|$ for the cardinality of tuple $t \in T$ (the number of times t occurs in T).

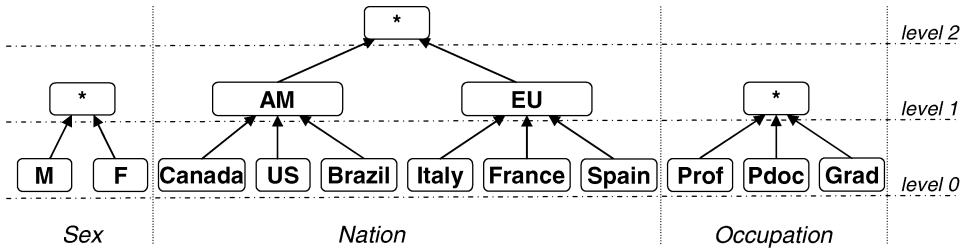


Fig. 1. DGH structures.

Although there are many ways to generalize a given value, in this article, we stick to generalizations according to domain generalization hierarchies (DGH) given in Figure 1.

Definition 2.1 (*i-Gen Function*). For two data values v^* and v from some attribute A, we write $v^* = \Delta_i(v)$ if and only if v^* is the ith parent of v in the DGH for A. Similarly for tuples t, t^*, $t^* = \Delta_{i_1 \cdots i_n}(t)$ iff $t^*[c] = \Delta_{i_c} t[c]$ for all columns c. Function Δ without a subscript returns all possible generalizations of a value v. We also abuse notation and write $\Delta^{-1}(v^*)$ to indicate the children of v^* at the leaf nodes.

For example, given the DGH structures in Figure 1. Δ_1(USA)=AM, Δ_2(Canada)=*, $\Delta_{0,1}$(<M,USA>)=<M,AM>, Δ(USA)={USA,AM,*}, Δ^{-1}(AM) ={USA, Canada, Brazil}

Definition 2.2 (μ-*Generalization*). A generalization mapping μ is any surjective function that maps tuples from domain D to a generalized domain D^* such that for $t \in D$ and $t^* \in D^*$; we have $\mu(t) = t^*$ (we also use notation $\Delta_\mu(t) = \mu(t)$ for consistency) only if $t^* \in \Delta(t)$. We define $\Delta_\mu^{-1}(t^*) = \{t_i \in D \mid \Delta_\mu(t_i) = t^*\}$. We say a table T^* is a μ-generalization of a table T with respect to a set of attributes QI and write $\Delta_\mu(T) = T^*$, if and only if records in T^* can be ordered in such a way that $\Delta_\mu(T[QI][r]) = T^*[QI][r]$ for every row r.

In Table I, T_1^*, T_2^* are two generalizations of T with mappings μ_1 and μ_2 respectively; E.g, $\Delta_{\mu_1}(T) = T_1^*$. Δ_{μ_1}(<F,US,Prof>) = <*,AM,Prof>; <F,US,Prof> $\in \Delta_{\mu_1}^{-1}$(<*,AM,Prof>).

Definition 2.3 (*Single Dimensional Generalization*). We say a mapping μ is $[i_1, \ldots, i_n]$ single dimensional iff given $\mu(t) = t^*$, we have $t^* = \Delta_{i_1 \cdots i_n}(t)$. We define in this case the level of μ as $i_1 + \cdots + i_n$.

Each attribute in the output domain of a single dimensional mapping contains values from the same level of the corresponding DGH structure. In Table I, T_2^* is a [0,1,1] generalization of T. (E.g., μ^{-1}(<M,AM,*>) = {<M,x, y> | $x \in \{US, Canada, Brazil\}$, $y \in \{Prof, Grad, Pdoc\}$}.) T_1^* is not single dimensional. (E.g., values * and AM both appear in T_1^*.) Single dimensional mappings are easily represented with a list of numbers.

Given two single dimensional mappings $\mu^1 = [i_1^1, \ldots, i_n^1]$ and $\mu^2 = [i_1^2, \ldots, i_n^2]$, we say μ^1 is a higher mapping than μ^2 and write $\mu^1 \subset \mu^2$ iff $\mu^1 \neq \mu^2$ and $i_j^1 \geq i_j^2$ for all $j \in [1 - n]$.

To explain concepts, in this article, we stick to single-dimensional generalizations. But we also briefly cover multidimensional generalizations that do not have the restriction that all values in generalizations should belong to the same generalized domain:

Definition 2.4 (*Multidimensional Generalization*). We say a mapping μ is multidimensional iff the following condition is satisfied. Whenever we have $\mu(t) = t^*$, we also have $\mu(t_i) = t^*$ for every $t_i \in \Delta^{-1}(t^*)$.

Table I. Private Table T, 2-Anonymous Generalization T_1^*, and 2-Diverse Generalization T_2^*

Name	Sex	Nation	Occ.	Sal.
q1	M	US	Grad	L
q2	M	Spain	Grad	L
q3	F	Italy	Pdoc	H
q4	F	Brazil	Pdoc	L
q5	M	Canada	Prof	H
q6	F	US	Prof	H
q7	F	France	Prof	L
q8	M	Italy	Prof	H

T

Name	Sex	Nation	Occ.	Sal.
q1	M	*	Grad	L
q2	M	*	Grad	L
q3	F	*	Pdoc	H
q4	F	*	Pdoc	L
q5	*	AM	Prof	H
q6	*	AM	Prof	H
q7	*	EU	Prof	L
q8	*	EU	Prof	H

T_1^*

Name	Sex	Nation	Occ.	Sal.
q1	M	AM	*	L
q5	M	AM	*	H
q2	M	EU	*	L
q8	M	EU	*	H
q3	F	EU	*	H
q7	F	EU	*	L
q4	F	AM	*	L
q6	F	AM	*	H

T_2^*

Every single dimensional mapping is also multidimensional. In Table I, both T_1^* and T_2^* are multidimensional generalizations of T.

While multidimensional generalizations are more flexible than single dimensional generalizations, single dimensional algorithms still offer three main advantages:

—Single dimensional algorithms produce *homogeneous* outputs meaning each value is generalized the same way throughout the anonymization. This makes it easy to modify applications designed for classical databases to work with higher level anonymizations. (E.g., in T_2^* every `US` becomes `AM`. T_2^* can be utilized by any regular database application without any modification costs. In T_1^* however some `US` values generalize to `AM` while others to $*$. Fuzziness introduced by T_1^* needs to be handled by specialized applications.)

—The mappings used in single dimensional algorithms translate well into the current legal standards that draw boundaries in terms of generalized domains. (For instance, for deidentification, the United States Healthcare Information Portability and Accountability Act (HIPAA) [HIPAA 2001] suggests the removal of any information regarding dates more specific than the year.)

We review the literature on anonymization algorithms in Section 2.2.

The main disadvantage of single dimensional algorithms is that they produce less utilized outputs compared to multidimensional algorithms. As we shall see in Section 7, tolerating some suppression helps negate the effects of inflexibility and can substantially increase utility in anonymizations. Since single dimensional algorithms are more sensitive to outliers [Nergiz and Clifton 2007], they tend to benefit more from suppression tolerance. Thus, the disadvantage of single dimensional algorithms can be reduced to some extent via suppression.

Next we briefly revisit some anonymity metrics.

While publishing person specific sensitive data, simply removing uniquely identifying information (SSN, name) from data is not sufficient to prevent identification because partially identifying information, quasi-identifiers (QI), such as age, sex, nation, occupation,... can still be mapped to individuals (and possibly their sensitive

information such as salary) by using external knowledge. [Samarati 2001; Sweeney 2002]. (Even though T of Table I does not contain information about names, releasing T is not safe when external information about QI attributes is present. If an adversary knows some person Bob is a male professor from US; she can map Bob to tuple q1 thus to salary High.) The goal of k-anonymity privacy protection is to limit the linking of a record from a set of released records to a specific individual even when adversaries can link individuals to QI:

Definition 2.5 (*k-Anonymity [Samarati 2001; Ciriani et al. 2007]*). A table T^* is k-anonymous with respect to a set of quasi-identifier attributes QI if each tuple in $T^*[QI]$ appears at least k times.

T_1^*, T_2^* are 2-anonymous generalizations of T. Note that given T_1^*, the same adversary can at best link Bob to tuples q1 and q2.

Definition 2.6 (*Equality group*). The equality group of tuple t in dataset T^* is the set of all tuples in T^* with identical quasi-identifiers to t.

In dataset T_1^*, the equality group for tuple $q1$ is $\{q1, q2\}$. We use colors to indicate equality groups in T_1^* and T_2^*.

While k-anonymity limits identification of tuples, it fails to enforce constraints on the sensitive attributes in a given equality group. Thus, sensitive information disclosure is still possible in a k-anonymization. (E.g., in T_1^*, both tuples of equality group {q1,q2} have the same sensitive value.) This problem has been addressed [Machanavajjhala et al. 2006; Li and Li 2007; Øhrn and Ohno-Machado 1999; Wong et al. 2006] by enforcing diversity on sensitive attributes within a given equivalence class. In this article, we will be covering the naive version of ℓ-diversity [Xiao and Tao 2006a][1]:

Definition 2.7 (ℓ-*Diversity*). Let r_i be the frequency of the most frequent sensitive attribute in an equality group G_i. An anonymization T^* is ℓ-diverse if for all equality groups $G_i \in T^*$, we have $\frac{r_i}{|G_i|} \leq \frac{1}{\ell}$.

In Table I, T_2^* is a 2-diverse generalization of T meaning the probability of a given individual having any salary is no more than .5. T_1^* violates ℓ-diversity for all $\ell > 1$.

In the following sections, we use the following property of k-anonymity and ℓ-diversity proved respectively by LeFevre et al. [2005] and Machanavajjhala et al. [2006]:

Definition 2.8 (*Anti-monotonicity*). Given $\mu^1 \subset \mu^2$ and a dataset T, if $\Delta_{\mu^1}(T)$ is not k-anonymous (or ℓ-diverse), neither is $\Delta_{\mu^2}(T)$.

In Table I, if T_2^* is not k-anonymous (or ℓ-diverse), neither is T.

There may be more than one k-anonymization (or ℓ-diverse anonymization) of a given dataset, and the one with the most information content is desirable. Previous literature has presented many metrics to measure the utility of a given anonymization [Iyengar 2002; Nergiz and Clifton 2007; Kifer and Gehrke 2006; Domingo-Ferrer and Torra 2005; Bayardo and Agrawal 2005]. We revisit Loss Metric, defined in Iyengar [2002] and previously used in [Domingo-Ferrer and Torra 2005; Nergiz et al. 2007; Nergiz et al. 2009c; Nergiz and Clifton 2007; Gionis et al. 2008]. Given a is the number of attributes:

$$LM(T^*) = \frac{1}{|T^*| \cdot a} \sum_{i,j} \frac{|\Delta^{-1}(T^*[i][j])| - 1}{|\Delta^{-1}(*)| - 1}$$

[1]The original definition given in [Machanavajjhala et al. 2006] protects against adversaries with additional background that we do not consider in this paper.

LM metric can be defined on individual data cells. It penalizes the value of each data cell in the anonymized dataset depending on how general it is (how many leaves are below it on the DGH tree). For example, $LM(\texttt{EU}) = \frac{|\Delta^{-1}(\texttt{EU})|-1}{|\Delta^{-1}(*)|-1} = \frac{3-1}{6-1}$. LM for a dataset normalizes the total cost to get a number between 0 and 1.

Despite its vulnerabilities, in Section 4 we start our analysis with k-anonymity instead of ℓ-diversity. There are two reasons for this. First, k-anonymity has a simple definition and instant k-anonymization is a simpler subproblem of instant ℓ-diversity. Second, k-anonymity is still used for creating ℓ-diverse anonymizations that are resistant to minimality attacks [Wong et al. 2007]. Such attacks are carried out by adversaries who know the underlying anonymization algorithm and enable adversary to violate ℓ-diversity conditions even though the released anonymization is ℓ-diverse. The basic idea to create resistant algorithms is to group the tuples without considering sensitive attributes as in k-anonymity, then enforce ℓ-diversity in each equality group by distortion if necessary. In Section 5, we continue with the problem of achieving instant ℓ-diversity.

2.2. Anonymization-based Privacy Protection

Single dimensional generalizations have been proposed in Samarati [2001] and LeFevre et al. [2005] and have been adopted by many works [Machanavajjhala et al. 2006; Nergiz et al. 2007; Li and Li 2007, 2006; Fung et al. 2005; Nergiz and Clifton 2009]. Samarati [2001] observes that all possible single dimensional mappings create a lattice over the subset operation. The proposed algorithm finds an optimal k-anonymous generalization (optimal in minimizing a utility cost metric) by performing a binary search over the lattice. LeFevre et al. [2005] improve this technique with a bottom-up pruning approach and finds all optimal k-anonymous generalizations. Fung et al. [2005] show that a top-down approach can better speed up the k-anonymity algorithm. Bayardo and Agrawal [2005] introduce more flexibility by relaxing the constraint that every value in the generalization should be in the same generalized domain. Machanavajjhala et al. [2006], Nergiz et al. [2007], and Li and Li [2007] adopt previous single dimensional algorithms for other privacy notions such as ℓ-diversity, t-closeness, and δ-presence.

There have also been other algorithms that output heterogeneous generalizations. Nergiz and Clifton [2007], Byun et al. [2007], and Agrawal et al. [2006] use clustering techniques to provide k-anonymity. LeFevre et al. [2006] and Hore et al. [2007] partition the multidimensional space to form k-anonymous and ℓ-diverse groups of tuples. Ghinita et al. [2007] make use of space filling curves to reduce the dimensionality of the database and provides k-anonymity and ℓ-diversity algorithms that work in one dimension. To the best of our knowledge, there are two works that address the scalability problem in multidimensional anonymization algorithms. The first work in Iwuchukwu and Naughton [2007] proposes a k-anonymization algorithm that makes use of R-tree indexes. The algorithm partitions the space in a bottom-up manner and reports execution times in seconds on synthetic data. Unfortunately, their approach cannot easily be extended for optimal single-dimensional algorithms.

The second work [LeFevre et al. 2008] is of more interest to us. The first proposed algorithm Rothko-T is based on extracting frequency sets (statistics necessary to decide on a split point) from the dataset and splitting the multidimensional domain with respect to a chosen attribute. Statistics are updated whenever a split happens. Rothko-T does not make statistical analysis or predictions. The second proposed algorithm Rothko-S is based on predictions and is showed to be more efficient than Rothko-T; thus is more relevant to us. Rothko-S extracts a sample (that fits in memory) from the dataset and uses the sample to identify split points within the Mondrian algorithm. The paper analytically shows how to calculate the confidence on the selected split

points for k-anonymity but mentions the difficulty of doing a similar analysis for ℓ-diversity. Another sample is re-extracted in case the confidence for a smaller partition drops below a threshold. Experiments on synthetic data identify cases in which the sampling-based approach decreases the number of scans to three while returning the same output as Mondrian returns. A similar approach can be followed for single-dimensional algorithms. Even without an empirical analysis of such an extension, we can state the following differences with our approach. First, both approaches differ in the way the probabilistic analysis is carried out. Given a sample, LeFevre et al. [2008] bound the probability that a split of space will violate the anonymity requirements. They use first order Bonferroni bounds which is faster to compute but which tends to give wide bounds [Schwager 1984]. We approximate this probability given a summary structure by using a more direct approach. Second, we provide a formal analysis also for ℓ-diversity (given some statistics on data). As mentioned in LeFevre et al. [2008] and as we shall see in Section 5 compared to k-anonymity, the probabilistic analysis of ℓ-diversity is harder and computing confidence on ℓ-diversity is much less efficient. Thus we also propose optimizations to speed up the analysis.

Due to the large space of heterogeneous generalizations, none of the above mentioned algorithms guarantees optimality in their domain.

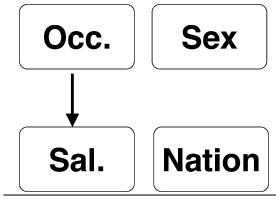
To the best of our knowledge, only LeFevre et al. [2005] report results regarding the efficiency of the single-dimensional anonymity algorithm when implemented over a real database. The time required for the single dimensional algorithm to run over a moderate database is in hours. By using the techniques given in the rest of this paper, we will reduce the execution time to seconds with little or no overhead on the utility.

Besides those already mentioned, there has been other work on anonymization of datasets: Xiao and Tao [2006a] pointed out that if the sole purpose for anonymization is to protect against sensitive information disclosure, we can avoid generalizing quasi-identifiers for maximum utilization and achieve the same privacy guarantees by associating sensitive values with equality groups rather than individual tuples. In Jiang and Clifton [2006] and Zhong et al. [2005], anonymity was achieved in a distributed system by the use of secure multi party computations. In Xiao and Tao [2006b], privacy requirements for anonymizations were personalized based on individual preferences on sensitive attributes. Kifer and Gehrke [2006] showed releasing marginal count tables along with anonymizations can increase utilization without violating k-anonymity privacy constraints. Wong et al. [2007] showed optimality with respect to a cost metric can be exploited to recover sensitive attributes from an anatomization. Some studies [Nergiz et al. 2009a; Bonchi et al. 2008; Nergiz et al. 2009c; Terrovitis et al. 2008; Hay et al. 2008; Cormode et al. 2008; Aggarwal and Yu 2007] extend anonymity definitions for relational, spatio-temporal, transactional, graph, and string data. And recently, there has been work on modeling adversary background knowledge in a variety of privacy settings [Martin et al. 2007; Chen et al. 2007; Du et al. 2008; Li et al. 2009].

2.3. Selectivity Estimation and Multivariate Statistics

In order to come up with an execution plan, most current DBMS estimate the result size of the queries. For this, the DBMS constructs and maintains a summary model that predicts the joint probability of the attribute values. Such a summary model is freely available as a database resource. Many types of summary models have been proposed. Kooi [1980] and Poosala et al. [1996] are just two among many that use histograms on each distinct attribute and assume attribute independence to predict joint distributions. While many real DBMS maintain histograms, it has been shown that such first order histograms cannot accurately describe a moderately skewed data. This led to many other approaches [Matias et al. 2000, 1998; Bruno et al. 2001;

Table II.



$\mathcal{P}(\texttt{Prof}) = .5$ $\mathcal{P}(\texttt{H}|\texttt{Prof}) = .75$
$\mathcal{P}(\texttt{Pdoc}) = .25$ $\mathcal{P}(\texttt{H}|\texttt{Pdoc}) = .5$
$\mathcal{P}(\texttt{Grad}) = .25$ $\mathcal{P}(\texttt{H}|\texttt{Grad}) = 0$

$\mathcal{P}(\texttt{M}) = .5$ $\mathcal{P}(\texttt{US}) = .25$
$\mathcal{P}(\texttt{F}) = .5$ $\mathcal{P}(\texttt{It}) = .25$
$\vdots$

Lee et al. 1999; Markl et al. 2007; Aboulnaga and Chaudhuri 1999; Poosala and Ioannidis 1997; Getoor et al. 2001] in which more complex summary models are maintained to capture multivariate statistics. One such technique supported by Oracle is dynamic sampling that involves scanning a small random sample of data blocks to extract more accurate statistics [Oracle 2009].

As histograms are the most widely used resources for query selectivity, we adopt them in our analysis. As an example for complex summary models, we also adopt Bayesian networks (BN) [Getoor et al. 2001].[2] BNs are basically directed graphs in which each vertex corresponds to an attribute and each edge shows a dependency relation between the connected attributes (see Table II). Attribute pairs not connected by an edge are assumed to be conditionally independent. Conditional probabilities on the connected attributes are supplied with the BN structures.

We show, in Table II, the BN structure for table T of Table I. The structure implies correlation between the attributes occupation and salary, but shows independence for the rest of the attributes. For the attributes in the root nodes, first-order distributions are released. For the salary attribute with an incoming edge from occupation, conditionals of the form $\mathcal{P}(\texttt{Sal}|\texttt{Occ})$ are released. Note that BNs most often are not 100% accurate. There may exist relatively small dependencies between attributes that are not captured by the graph.

Even though we adopt histograms and BNs in our experiments, it should be noted that the methodology given in this article is independent of the summary model being offered by the DBMS. All we need is the joint probability distribution for attributes. Thus, from now on we stop referring to BNs and assume we have access to a summary function F as a database resource:

Definition 2.9. A summary function F on a dataset T, when given an anonymized tuple t^* returns an approximation to the probability that a randomly selected tuple $t \in T$ will satisfy $t^* \in \Delta(t)$.

If we use a histogram over T in Table I as a summary function; by attribute independence, $F(\texttt{<M,US,Prof,H>}) = \mathcal{P}(\texttt{M})\mathcal{P}(\texttt{US})\mathcal{P}(\texttt{Prof})\mathcal{P}(\texttt{H}) = 0.5 \cdot 0.25 \cdot 0.5 \cdot 0.5 = 0.03125$. If we use the BN structure from Table II, we have $F(\texttt{<M,US,Prof,H>}) = \mathcal{P}(\texttt{M})\mathcal{P}(\texttt{US})\mathcal{P}(\texttt{Prof})$ $\mathcal{P}(\texttt{H}|\texttt{Prof}) = 0.5 \cdot 0.25 \cdot 0.5 \cdot 0.75 = 0.046875$.

Similarly, $F(\texttt{<M,AM,*,H>}) = \mathcal{P}(\texttt{M}) \sum_{v \in \Delta^{-1}(\texttt{AM})} \mathcal{P}(\texttt{v}) \sum_{v \in \Delta^{-1}(*)} \mathcal{P}(\texttt{v})\mathcal{P}(\texttt{H}|\texttt{v}) = 0.5 \cdot 0.5 \cdot (0.375 + 0.125 + 0) = 0.125$. If we were to use sampling, we would calculate $F(\texttt{<M,US,Prof,H>})$ by looking at the sample size and the frequency of <M,US,Prof,H> in the sample.

[2]We are not aware of a real system that utilizes BNs for selectivity. But in the light of recent research and developments in the market [Oracle 2009], we believe there is strong motivation in using a complex summary model capturing higher order statistics. Thus, we include BNs ,in addition to histograms, in our discussion to evaluate our approach with respect to a complex summary structure.

Table III. Notations

$T[c][r], t[c]$	value of column c, row r in table T and attribute c in tuple t
T^*, t^*, v^*	any generalization of table T, tuple t, and value v
$\Delta_\mu(T), \Delta_\mu(t), \Delta_\mu(v)$	generalizations of table T, tuple t and value v with respect mapping μ
$\Delta(v)$	set of all possible generalizations of value v
$\Delta^{-1}(v^*)$	set of all atomic values that generalized value v^* stands for
$\mathcal{P}_{\mu_k}, \mathcal{P}_{\mu_\ell}$	Given a summary structure and private table T, probability that $\Delta_\mu(T)$ is k-anonymous (ℓ-diverse)
$\mathcal{E}_{\mu_k}, \mathcal{E}_{\mu_\ell}$	Given a summary structure and private table T, expected number of tuples violating k-anonymity (ℓ-diversity) in $\Delta_\mu(T)$

3. INSTANT ANONYMIZATION ALGORITHMS

3.1. μ-Probability and μ-Expectation

Given a summary function F on a table T and the size of T, our aim is to find a mapping μ that will *likely* make T k-anonymous or *close* to k-anonymous. Most anonymity algorithms search a fixed space of generalization mappings and check for each mapping to see if the anonymity constraints are satisfied. Thus, the following two definitions play a crucial role in designing instant anonymization algorithms:

Definition 3.1 (μ_k-*Probability* $\mathcal{P}_{\mu_k}$). Given F on T, a mapping μ and the size of T, μ_k-probability is the probability that $\Delta_\mu(T)$ is k-anonymous.

Definition 3.2 (μ_k-*Expectation* $\mathcal{E}_{\mu_k}$). We say outliers for an anonymization T^* are those tuples in T^* that violate k-anonymity. Given F on T, a mapping μ and the size of T, μ_k-expectation is the expected number of outliers in $\Delta_\mu(T)$.

Both definitions are useful for our purpose. μ_k-Probability is our confidence to get a k-anonymization when we apply the mapping, however this does not say anything about the number of tuples violating the condition in the worst cases. There might just be only one outlier in the dataset violating k-anonymity and yet μ_k-probability tells nothing about it. Note that data releaser has the option to fully suppress an outlier to enforce k-anonymity, so a mapping producing a small number of outliers is still an alternative to the data releaser. μ_k-Expectation on the other hand identifies the expected number of outliers, however does not say anything about the distribution. A mapping with a good expectation can very well result in a huge number of outliers with unacceptably high probability. We evaluate the effectiveness of both notions in Section 7.

In an ℓ-diversity framework, the notions of μ-probability and μ-expectation have slightly different meanings. We redefine both notions for ℓ-diversity:

Definition 3.3 (μ_ℓ-*Probability* $\mathcal{P}_{\mu_\ell}$). Given F on T, a mapping μ and the size of T, μ_ℓ-probability is the probability that $\Delta_\mu(T)$ is ℓ-diverse.

Definition 3.4 (μ_ℓ-*Expectation* $\mathcal{E}_{\mu_\ell}$). We say diversity outliers for an anonymization T^* are those tuples in T^* whose equality group violates ℓ-diversity. Given F on T, a mapping μ and the size of T, μ_ℓ-expectation is the expected number of diversity outliers in $\Delta_\mu(T)$.

The notation is summarized in Table III. We now prove that higher-level generalizations have higher μ_k-probabilities:

THEOREM 3.5. *Given* $\mu^1 \subset \mu^2$, $\mathcal{P}_{\mu_k^1} \geq \mathcal{P}_{\mu_k^2}$ *and* $\mathcal{E}_{\mu_k^1} \leq \mathcal{E}_{\mu_k^2}$.

PROOF. Let $A_{\mu,T}$ be the event that $\Delta_\mu(T)$ is k-anonymous. Thus, by Definition 3.1, μ-probability for μ^1 is given by,

$$\begin{aligned}\mathcal{P}_{\mu_k^1} &= \mathcal{P}(A_{\mu^1,T} \mid F) \\ &= \sum_{T_i} \mathcal{P}(A_{\mu^1,T} \mid T = T_i, F) \cdot \mathcal{P}(T = T_i \mid F) \\ &= \sum_{T_i} \mathcal{P}(A_{\mu^1,T} \mid T = T_i) \cdot \mathcal{P}(T = T_i \mid F)\end{aligned}$$

Since given a table T_j, we can check for k-anonymity, by antimonotonicity, we have $\mathcal{P}(A_{\mu^1,T_j} \mid T_j) \geq \mathcal{P}(A_{\mu^2,T_j} \mid T_j)$. Thus;

$$\begin{aligned}\mathcal{P}_{\mu_k^1} &\geq \sum_{T_i} \mathcal{P}(A_{\mu^2,T} \mid T = T_i) \cdot \mathcal{P}(T = T_i \mid F) \\ \mathcal{P}_{\mu_k^1} &\geq \mathcal{P}_{\mu_k^2}\end{aligned}$$

□

We skip the proof for μ_k-expectation as it is similar. The theorem and the proof can trivially again be extended for μ_ℓ-probability and μ_ℓ-expectation.

In Sections 4 and 5, we probabilistically analyze k-anonymity and ℓ-diversity given a generalization mapping μ and show how to calculate μ-probability and μ-expectation of achieving k-anonymity or ℓ-diversity given the summary structure F and μ. But first, we show how these two notions can be used to create an instant algorithm

3.2. Single-Dimensional Instant Anonymization Algorithm: S-INSTANT

In this section, we present a single-dimensional algorithm that traces the whole space of single-dimensional mappings and returns a suitable mapping based on our previous analysis. The algorithm has two phases:

(1) Without looking at the data, the algorithm takes the summary structure, the data size, and an anonymity parameter as an input and returns an ordered set of candidate mappings based on either μ-probability or μ-expectation.
(2) The algorithm then applies the generalization mappings in the candidate set to the dataset until it finds one mapping satisfying the anonymity constraints.

The first phase is an in-memory calculation and is faster than the second phase which requires database access. The algorithm can be modified to return both k-anonymous and ℓ-diverse outputs To achieve k-anonymity or ℓ-diversity, we use the calculations in Section 4 or 5 respectively. Without loss of generality, in this section, we assume we want to achieve ℓ-diversity. We now give the details of the anonymization algorithm.

Algorithm S-INSTANT searches the whole domain of single dimensional mappings in a top-down manner (see Definition 2.3 and Figure 4a) to find those mappings with a μ-probability (or μ-expectation) bigger than a user threshold. Since the full domain of such mappings is quite large, S-INSTANT makes use of the anti-monotonicity property to prune the search space.

The skeleton of S-INSTANT is given in Algorithm 1:

First Phase. In this phase, we do not look at the data but just make use of the summary structure available to us. In lines 1–11, we first construct a candidate set of mappings such that each mapping μ in the list has a μ-probability $\mathcal{P}_\mu$ higher than a given threshold. To do that we need to traverse the whole space of single dimensional mappings and calculate $\mathcal{P}_\mu$ for each mapping. Fortunately, the possible single-dimensional mappings over a table domain form a lattice on the $\subset$ relation (see Figure 4a). In lines 3–10, we

ALGORITHM 1: S-INSTANT

Require: a private table T from domain D, a summary structure F on T, privacy parameter ℓ, a utility cost metric CM, a user threshold th;
Ensure: return a minimum cost ℓ-diverse full domain generalization of T.
1: let the candidate mapping set C be initially empty.
2: create lattice lat for all possible generalization mappings for D. Let n be the maximum level of mappings in lat.
3: **for all** level i from n to 0 **do**
4: **for all** mapping μ of level i in lat **do**
5: calculate $\mathcal{P}_\mu$ by using F
6: **if** $\mathcal{P}_\mu < th$ **then**
7: delete node μ and all children and grandchildren of μ from lat.
8: **else**
9: calculate CM that would result from applying μ.
10: $C+=\mu$
11: sort C in ascending order with respect to CM values.
12: **for all** $\mu \in C$(starting from the first element) **do**
13: create $T^* = \Delta_\mu(T)$.
14: **if** T^* is ℓ-diverse **then**
15: return T^*
16: return null

traverse the lattice in a top-down manner. In lines 6–7, we use the anti-monotonicity property of ℓ-diversity to prune the lattice, thus reduce the search space. In line 9–10, we collect those mappings that are not pruned in the candidate set and in line 11, we sort the mappings with respect to the utility metric. The LM cost metric is a good candidate here mainly because we can calculate the LM cost of a mapping without accessing data as long as we can get marginal distributions of attributes from the summary structure (this is the case for summary structures such as Bayesian networks and histograms). But other cost metrics could also be used in the form of expected costs.

Second Phase. When we apply the mappings in the candidate set to the private table, we do not necessarily get ℓ-diverse anonymizations. Thus, in lines 12–15, we test whether the resulting anonymizations satisfy ℓ-diversity. We start testing the lowest cost mappings and continue until we find an ℓ-diverse anonymization. Note that this phase of the algorithm requires data access. Also note that depending on the user supplied threshold, the resulting anonymization is not necessarily optimal in minimizing the utility cost metric.

The selection of the threshold value is crucial to the execution of the algorithm. A too small threshold would cause the algorithm to prune too little thus the algorithm would possibly require many data accesses in the second phase but most likely find the optimal anonymization. A too big threshold would cause the algorithm to prune too much; thus, the algorithm would return a high cost anonymization but would most likely make only one data access. In Section 7, we show experimentally that this really is not a problem in practice.

The effectiveness of the first phase depends on the accuracy of the summary structure. For Bayesian networks, the accuracy drops as the number of outliers in the dataset increases. Fortunately, most real datasets inherit enough patterns for BNs to capture the underlying distribution. Again fortunately, the first phase of S-INSTANT is not too sensitive to outliers. An outlier in a more specific domain may not be an outlier in a generalized domain (If we consider an outlier as a random point, it is likely that some will follow the same distribution as the other non-outliers in the new generalized domain.) In most cases, even a small amount of generalization can wipe out most of the previously existing outliers. S-INSTANT becomes sensitive to outliers only when

ALGORITHM 2: M-INSTANT

Require: a private table T from domain D, a summary structure F on T, privacy parameter ℓ, a user threshold th;
Ensure: return a multidimensional ℓ-diverse generalization of T.
1: let Q be an empty queue of partitions (subdomains).
2: let M be an empty stack of mappings.
3: push a mapping μ respecting domain D into M.
4: enqueue D into Q.
5: **while** there exists at least one subdomain unmarked in Q **do**
6: dequeue subdomain D' from Q.
7: **if** D' is marked **then**
8: continue.
9: **for all** attribute a **do**
10: partition D' with respect to a into subdomains D_1 and D_2.
11: create mapping μ respecting subdomains in $\{D_1, D_2\} \cup Q$.
12: calculate $\mathcal{P}_\mu$ by using F.
13: **if** $\mathcal{P}_\mu > th$ **then**
14: enqueue D_1, D_2 into Q; push μ into M.
15: break.
16: **if** a is the last attribute **then**
17: mark D', enqueue D'.
18: **while** M is not empty **do**
19: pop μ from M.
20: create $T^* = \Delta_\mu(T)$.
21: **if** T^* is ℓ-diverse **then**
22: further partition big segments in T^* and return T^*.
23: return null.

we go down on the generalization lattice to more specific domains, and this happens, if happens at all, towards the end of its execution. Inaccuracy in prediction in specialized domains is not as costly as that in generalized domain. As we shall see in Section 7, even histograms are accurate enough for S-INSTANT to make an accurate prediction at the end of the first phase.

Algorithm S-INSTANT can easily be modified to prune with respect to μ-expectation. We change line 6 to check against $\mathcal{E}_\mu > th$. This is especially useful when we allow the algorithm to suppress some portion of the tuples (specified by the user as the acceptable suppression rate) to minimize the effect of outliers on utility [LeFevre et al. 2005]. In such a case, acceptable suppression rate is a good candidate for a threshold value.

As we mentioned before, μ-expectation is a better approach than μ-probability for algorithms with suppression tolerance. However, this does not mean we cannot use μ-probability for such algorithms. μ-Probability would return lower probabilities than the actual one, but this difference can be offset by decreasing the threshold value. Since it is not easy to find such a suitable threshold analytically, we do not elaborate on this issue.

3.3. Multidimensional Instant Anonymization Algorithm: M-INSTANT

In this section, we present a multidimensional instant algorithm by modifying the Mondrian algorithm with median partitioning proposed by LeFevre et al. [2006]. The structure of the algorithm is similar to the one given in LeFevre et al. [2008].

Without loss of generality, assume we want to create an ℓ-diverse anonymization. We present the pseudocode in Algorithm 2. As in Section 3.2, in the first phase (lines 3–17), we create several partitions of the multidimensional domain and calculate the probability of ℓ-diversity given the partitions and the summary structure. The way

we create partitions as follows: We first partition the whole domain into two segments by splitting with respect to the first dimension. Then we continue partitioning each segment by splitting with respect to the other dimensions. For each partitioning, we create a generalization mapping respecting the subdomains (e.g., a mapping that maps data values to subdomains). We continue splitting the segments recursively until the probability of ℓ-diversity (given the associated mapping) goes below a certain threshold. We start the second phase with the final partitioning of the domain. Again note that we do not look at the data at this phase.

In the second phase (lines 18–22), we check whether applying the final partitioning to the dataset produces an ℓ-diverse anonymization. If not, we merge the segments that violate ℓ-diversity with their siblings until the final segments all satisfy ℓ-diversity. Furthermore, we try to partition the relatively big segments (if any exists) to create a better utilized anonymization.

In Section 7, we experimentally evaluate M-INSTANT and show that the M-INSTANT algorithm produces anonymizations in seconds or few minutes while the database implementation of Mondrian takes around 30 minutes to output an anonymization of similar utility.

4. INSTANT *K*-ANONYMITY

In this section, we look at the instant k-anonymity problem. Without loss of generality, we assume, for only this section, every table has only QI attributes (e.g., we ignore the salary attribute in Table I).

4.1. Deriving μ_k-Probability

Calculating the μ_k-probability is a computationally costly operation. To overcome this challenge, we make the following assumption in our probabilistic model:

Tuple Independence. When we compute μ_k-probability, we assume distinct tuples are drawn from the same distribution (F) but are independent from each other. Meaning for any two tuples $t_1, t_2 \in T$, $\mathcal{P}(t_1[i] = v_j) = \mathcal{P}(t_1[i] = v_j \mid t_2[i] = v_k)$ for all possible i, v_j, and v_k. Such equality does not necessarily hold given F. But for large enough data, independence is a reasonable assumption. To demonstrate this, consider that we sample from a census dataset of size n in which we know (from the histograms or bayesian network) exactly $n \cdot p$ of the members are female. If we know that a particular individual t_1 is female, for $p = 0.5$ the probability that another individual t_2 being also female is $\frac{(0.5 \cdot n)-1}{n-1}$. By tuple independence, we approximate this probability as 0.5. Note that as n goes to infinite, the difference between the two quantities becomes 0. For $n = 100$ (which can be considered a small number compared to the average sizes of current databases), the error is around 0.005. Lahiri et al. [2007] present a more extensive analysis on the error showing that the approximation is accurate for large n and for p not too close to 0 or 1.

Definition 4.1 (*Bucket Set*). A bucket set for a set of attributes C, and a mapping μ, is given by $B = \{\text{tuple } b \mid \text{there exists at least one tuple } t \text{ from the domain of } C \text{ such that } b = \Delta_\mu(t)\}$

In table T of Table I, for the mapping [0,1,1], the bucket set is given by {<M,AM,*>, <M, EU,*>, <F,AM,*>, <F,EU,*>}. When we refer to this bucket set, we will index the elements: $\{b_1, b_2, b_3, b_4\}$. (For the multidimensional mapping used in T_1^*, the bucket set would be {<M,*,Grad>, <F,*,Grad>, <M,*,Pdoc>, <F,*,Pdoc>, <*,AM,Prof>, <*,EU, Prof>}).

Generalization of any table T with a fixed mapping μ can only contain tuples drawn from the associated bucket set $B = \{b_1, \ldots, b_n\}$. Since we do not access T at this time,

the cardinality of the buckets acts as a random variable. However, we know the size of T. Letting X_i be the random variable for the cardinality of b_i, and assuming T has size N, we have the constraint $\sum_i X_i = N$.

In Table I, $N = |T| = 8$. So for the buckets $\{b_1, b_2, b_3, b_4\}$; we have $X_1 + X_2 + X_3 + X_4 = 8$.

A generalization T^* satisfies k-anonymity if each bucket (generalized tuple) in T^* has cardinality of either 0 or at least k. Using the notation $X \geq^0 k$ for $(X \geq k) \vee (X = 0)$, μ_k-probability takes the following form:

$$\mathcal{P}_{\mu_k} = \mathcal{P}\left(\bigcap_i X_i \geq^0 k \middle| \sum_i X_i = N, F \right).$$

By definition, the summary function F determines the probability that a random tuple $t \in T$ will be generalized to a bucket b_i:

$$\ell_i = F(b_i), \tag{1}$$

which we all the *likelihood* of bucket b_i.

From the BN structure in Table II, the likelihood of $\ell_1 = F(\texttt{<M,AM,*>}) = .5 \cdot .5 \cdot 1 = 0.25$. Similarly $\ell_2 = F(\texttt{<M,EU,*>}) = \ell_3 = F(\texttt{<F,AM,*>}) = \ell_4 = F(\texttt{<F,EU,*>}) = .25$.

Without tuple independence assumption, each X_i behaves like a hypergeometric[3] random variable with parameters $(N, N\ell_i, N)$. However, hypergeometrics are slow to compute. With tuple independence, we can model X_i as a binomial random variable[4] $\mathcal{B}$ with parameters (N, ℓ). Such an assumption is reasonable for big N and moderate ℓ values [Lahiri et al. 2007]. So the μ_k-probability can be written as:

$$\mathcal{P}_{\mu_k} = \mathcal{P}\left(\bigcap_i X_i \geq^0 k \middle| \sum_i X_i = N, X_i \sim \mathcal{B}(N, \ell_i) \right). \tag{2}$$

Equation (2) resembles a multinomial cumulative, except we constraint over the condition $\geq^0$ rather than $\leq$. Nergiz et al. [2009b] show that the exact calculation of Equation (2) is infeasible and propose an approximation which is based on the approximation of multinomial cumulatives given in Levin [1981]. We use their approximation to calculate $\mathcal{P}_{\mu_k}$, and refer to the Appendix for the derivation of it:

Let Y_i be a truncated binomial; $Y_i \sim (X_i | X_i \geq^0 k)$. Let $(\bar{X}_i, \bar{Y}_i)$ is the mean and $(\sigma^2_{X_i}, \sigma^2_{Y_i})$ is the variance of (X_i, Y_i) respectively, and $\mathcal{N}(m, \sigma^2)$ be the normal distribution with mean m and variance σ^2, then

$$\mathcal{P}_{\mu_k} \simeq \frac{\mathcal{P}(|\mathcal{N}_Y - N| \leq 0.5)}{\mathcal{P}(|\mathcal{N}_X - N| \leq 0.5)} \cdot \prod \mathcal{P}(X_i \geq^0 k),$$

where $\mathcal{N}_X \sim \mathcal{N}(\sum \bar{X}_i, \sum \sigma^2_{X_i})$ and $\mathcal{N}_Y \sim \mathcal{N}(\sum \bar{Y}_i, \sum \sigma^2_{Y_i})$

In Algorithm 3, we show the pseudocode for calculating the μ_k-probability. In lines 1–8, the algorithm processes each bucket b_i and calculates the mean and the variance of the associated truncated variable. These statistics are derived from the first and the second moments of the variable. Note that the density function for the truncated variable $\mathcal{P}(X = a | X \geq^0 k)$ is given by $\frac{\mathcal{P}(X=a)}{\mathcal{P}(X \geq^0 k)}$ for $a \geq k$. As there is no closed form for the cumulative function of a binomial distribution, unfortunately there is no closed form for the moments of the truncated binomial. Thus, algorithm in lines 3–6,

[3] $\mathcal{H}$(x;N,M,n):A sample of n balls is drawn from an urn containing M white and $N - M$ black balls *without replacement*. $\mathcal{H}$ gives the probability of selecting exactly x white balls.

[4] $\mathcal{B}$(x;n,p):A sample of n balls is drawn from an urn of size N containing Np white and $N(1-p)$ black balls *with replacement*. $\mathcal{B}$ gives the probability of selecting exactly x white balls.

ALGORITHM 3: μ_k-Probability

Require: Given a bucket set $B = \{b_1, \ldots, b_n\}$, the associated likelihood set $L = \{\ell_1, \ldots, \ell_n\}$ and the number of tuples N; return the μ_k-probability. Let $X_i \sim \mathcal{B}(N, \ell_i)$

1: **for all** $b_i \in B$ **do**
2: $cml_i = 0$, $Ex_i = 0$, $Ex_i^2 = 0$
3: **for all** $x \geq^0 k$ **do**
4: $cml_i += P(X_i = x)$
5: $Ex_i += x \cdot P(X_i = x)$
6: $Ex_i^2 += x^2 \cdot P(X_i = x)$
7: calculate the mean mu_i and the variance σ_i^2 of the variable X_i
8: calculate the mean $mu_i^{(t)}$ and the variance $\sigma^{2_i^{(t)}}$ of the truncated variable $X_i | X_i \geq^0 k$ from cml, Ex, and Ex^2.
9: $mu = \sum mu_i$, $\sigma^2 = \sum \sigma_i^2$
10: $mu^{(t)} = \sum mu_i^{(t)}$, $\sigma^{2^{(t)}} = \sum \sigma^{2_i^{(t)}}$
11: $num1 = \mathcal{P}(|\mathcal{N}(mu, \sigma^2) - N| \leq 0.5)$
12: $num2 = \prod(cml_i)$
13: $den = \mathcal{P}(|\mathcal{N}(mu^{(t)}, \sigma^{2^{(t)}}) - N| \leq 0.5)$
14: $num1 \cdot num2/den$

calculates the density function one by one at each point satisfying the k-anonymity constraint. The first and the second moments are extracted during the process. In lines 9–10, the algorithm sums up the expectations and the variances of all the binomials and the truncated variables to get the statistics for the sum of the random variables.

Lines 3–6 can be rewritten to iterate over the x values with $1 \leq x \leq k-1$. Thus, the in memory time complexity of calculating μ_k-probability is $O(n \cdot k)$. Note that the time complexity does not depend on the data size but the generalized domain size.

4.2. Deriving μ_k-Expectation

μ_k-Expectation is an easier problem which can be solved without the tuple independence assumption. Let random variable Z_i be defined as;

$$Z_i = \begin{cases} 0, & X_i \geq^0 k; \\ X_i, & \text{otherwise.} \end{cases}$$

In other words, Z_i holds the number of outliers in bucket b_i. Then the total number of outliers is the sum of all Z_i. $\mathcal{E}_\mu$ takes the following form:

$$\mathcal{E}_\mu = E\left(\sum_i Z_i\right) = \sum_i E(Z_i),$$

where

$$E(Z_i) = \sum_{j=1}^{k-1} j \cdot \mathcal{B}(j, N, \ell_i).$$

We show, in Algorithm 4, the pseudocode to calculate μ_k-expectation. In lines 3-4, the algorithm calculates the expected number of outliers in bucket b_i. The expectations for all buckets are summed up and returned as the μ_k-expectation. The time complexity of Algorithm 4 is also $O(n \cdot k)$.

ALGORITHM 4: μ_k-Expectation

Require: Same as in Algorithm 3.
1: $Ex = 0$.
2: **for all** $b_i \in B$ **do**
3: **for all** $x \in [1, (k-1)]$ **do**
4: $Ex+ = x \cdot P(X_i = x)$
5: return Ex.

5. INSTANT ℓ-DIVERSITY

5.1. Deriving μ_ℓ-Probability

Definition 5.1 (*Sensitivity Likelihood*). Sensitivity likelihood ℓ'_i for a sensitive value s_i and a bucket b with set of quasi identifier values $q_1^*, \ldots, q_a^*$ is the probability that a random tuple t will have the sensitive value s_i given that t has quasi-identifiers $q_1^*, \ldots q_a^*$. Given F, sensitivity likelihood ℓ'_i for s_i and b is calculated as;

$$\ell'_i = \frac{F(\langle q_1^*, \ldots, q_a^*, s_i \rangle)}{\sum_j F(\langle q_1^*, \ldots, q_a^*, s_j \rangle)}$$

There is one set of sensitivity likelihoods for each given bucket and they can be constructed from the given summary function. For instance, given the BN structure in Table II, Sex, Nation and Salary are independent thus sensitivity likelihood ℓ' for the sensitive value `H` and tuple `<M,US,Prof>` is $\mathcal{P}(\texttt{H}|\texttt{Prof}) = 1$. Similarly for tuple `<M,AM,*>` and value $s_1 = \texttt{H}$, $\ell'_1 = \frac{F(\texttt{<M,AM,*,H>})}{F(\texttt{<M,AM,*,*>})} = \frac{0.15625}{0.25} = 0.625$. Again for tuple `<M,AM,*>` and value $s_2 = \texttt{L}$, $\ell'_2 = 0.375$. Note that for all the buckets b_1, b_2, b_3, b_4, we have the same set of sensitivity likelihoods: $\{\ell'_1, \ell'_2\}$.

To achieve ℓ-diversity, we need to enforce constraints on the sensitive attribute of each bucket. In order to do that, we introduce another indicator random variable I_i for bucket b_i as follows;

$$I_i = \begin{cases} 1, & b_i \text{ satisfies diversity;} \\ 0, & \text{otherwise.} \end{cases}$$

Thus, given $X_i \sim \mathcal{B}(N, \ell_i)$, we are interested in the following probability:

$$\begin{aligned} \mathcal{P}_{\mu_\ell} &= \mathcal{P}\left(\bigcap_i I_i \,\middle|\, \sum_i X_i = N \right) \\ &= \frac{\mathcal{P}(\sum_i X_i = N \mid \bigcap_i I_i)}{\mathcal{P}(\sum_i X_i = N)} \cdot \mathcal{P}\left(\bigcap_i I_i \right) \\ &= \frac{\mathcal{P}(\sum_i (X_i|I_i) = N)}{\mathcal{P}(\sum_i X_i = N)} \cdot \prod_i \mathcal{P}(I_i). \end{aligned} \tag{3}$$

As we did before, we approximate the first numerator and the denominator with normal distribution. Calculation of the denominator is the same as before. For the first numerator, we need to calculate the mean (and the variance) of $X_i|I_i$ which is given by;

$$E(X_i|I_i) = \sum_x x \cdot \frac{\mathcal{P}(I_i|X_i = x) \cdot \mathcal{P}(X_i = x)}{\mathcal{P}(I_i)}.$$

Calculation of $\mathcal{P}(I_i)$ is also required in Equation (3). By conditioning on the variable X_i, we calculate $\mathcal{P}(I_i)$ as $\mathcal{P}(I_i) = \sum_x \mathcal{P}(I_i|X_i = x)$. Thus,

$$E(X_i|I_i) \;=\; \sum_x x \cdot \frac{\mathcal{P}(I_i|X_i = x) \cdot \mathcal{P}(X_i = x)}{\sum_x \mathcal{P}(I_i|X_i = x) \cdot \mathcal{P}(X_i = x)}. \tag{4}$$

All we need to show is how to calculate $\mathcal{P}(I_i|X_i = x)$. Informally, we are interested in the probability that a bucket of size x will satisfy ℓ-diversity given a set of sensitivity likelihoods. From now on we name this probability as the *conditional ℓ-diversity probability*. Note that we need to calculate the conditional ℓ-diversity for all possible x values and for each bucket. Following the example above, conditional ℓ-diversity problem for the bucket $b_1 =$ <M,AM,*> can be restated as the following: Given $\{\ell'_1, \ell'_2\}$, and $|b_1| = x$, what is the probability that b_1 satisfies ℓ-diversity?

Let U_j be the frequency of sensitive value s_j in b_i. By tuple independence, each $U_j \sim \mathcal{B}(x, \ell'_j)$. By the definition of ℓ-diversity, we are interested in the following probability;

$$\mathcal{P}(I_i|X_i = x) = \mathcal{P}\left(\bigcap_j \left(U_j \leq \frac{x}{\ell}\right) \middle| \sum_j U_j = x\right). \tag{5}$$

Interestingly, this is nearly the same problem as the k-anonymity problem we addressed in Section 4.1 (see the resemblance of Equation (2) and Equation (5)). The only difference is the constraint on the random variable. We can use the same technique which is based on the approximation of multinomial distribution to approximate the probability:

Let V_j be a truncated binomial; $V_j \sim (U_j|U_j \leq \frac{x}{\ell})$. Let $(\bar{U}_j, \bar{V}_j)$ be the mean and $(\sigma^2_{U_j}, \sigma^2_{V_j})$ is the variance of (U_j, V_j) respectively, then

$$\mathcal{P}(I_i|X_i = x) \simeq \frac{\mathcal{P}(|\mathcal{N}_V - N| \leq 0.5)}{\mathcal{P}(|\mathcal{N}_U - N| \leq 0.5)} \cdot \prod_j \mathcal{P}\left(U_j \leq \frac{x}{\ell}\right), \tag{6}$$

where $\mathcal{N}_U \sim \mathcal{N}(\sum \bar{U}_j, \sum \sigma^2_{U_j})$ and $\mathcal{N}_V \sim \mathcal{N}(\sum \bar{V}_j, \sum \sigma^2_{V_j})$.

In Algorithm 5, we show the pseudocode for calculating the μ_ℓ-probability of a given bucket set. Note the similarity to Algorithm 3. The key difference is that in lines 3–7, we calculate the moments of the conditional $X_i|I_i$ (the size of the bucket given the bucket is ℓ-diverse) rather than $X_i|X_i \geq^0 k$. To this end, as also mentioned in Equation (4), the algorithm, in line 4, calculates the conditional ℓ-diversity probability $\mathcal{P}(I_i|X_i = x)$ by calling Algorithm 3 with a slight modification. Specifically, the input to Algorithm 3 is now the associated sensitivity likelihood set and the condition in line 3 is changed in order to calculate the moments of $X_j|X_j \leq \frac{x}{\ell}$. Note that if we fix the bucket size as x, $\frac{x}{\ell}$ is the maximum frequency that a sensitive value can appear in an ℓ-diverse bucket.

Calculation of μ_ℓ-probability with Algorithm 5 requires two dimensional approximation of multinomial cumulatives. Cumulatives regarding the distribution in the sensitive attributes (see line 4) are faster to compute since the sizes of the buckets and the domain of sensitive attributes tend to be smaller compared to the size of the dataset and the domain of the QI attributes. For each bucket and for each $x \leq N$, we call Algorithm 3 which has a complexity of $o(m \cdot \frac{x}{\ell})$ where m is the domain size of the sensitive attribute. Thus, the algorithm has an in-memory complexity of $O(n \cdot m \cdot \frac{N^2}{\ell})$. When N is large, the calculation of μ_ℓ-probability is costly. In Section 6, we present several optimizations that speed up the algorithm significantly.

ALGORITHM 5: μ_ℓ-Probability for ℓ-Diversity

Require: Given a bucket set $B = \{b_1, \ldots, b_n\}$, the associated likelihood set $L = \{\ell_1, \ldots, \ell_n\}$, the number of tuples N, and for each b_i the associated sensitivity likelihood set $L'^{(i)} = \{\ell_1'^{(i)}, \ldots, \ell_m'^{(i)}\}$; return the μ_ℓ-probability. Let I_i be the event that b_i satisfies ℓ-diversity and X_i be the random variable for the size of b_i ($X_i \sim \mathcal{B}(N, \ell_i)$).

1: **for all** $b_i \in B$ **do**
2: $cml_i = 0$, $Ex_i = 0$, $Ex_i^2 = 0$
3: **for all** $x \in [1, N]$ **do**
4: calculate $\mathcal{P}(I_i|X_i = x)$ by calling Algorithm 3 with $X_j \sim \mathcal{B}(x, \ell_j'^{(i)})$ and truncation $X_j|X_j \leq \frac{x}{\ell}$.
5: $cml_i += \mathcal{P}(I_i|X_i = x) \cdot P(X_i = x)$
6: $Ex_i += x \cdot \mathcal{P}(I_i|X_i = x) \cdot P(X_i = x)$
7: $Ex_i^2 += x^2 \cdot \mathcal{P}(I_i|X_i = x) \cdot P(X_i = x)$
8: calculate the mean mu_i and the variance σ_i^2 of the variable X_i
9: calculate the mean $mu_i^{(t)}$ and the variance $\sigma^{2}{}_i^{(t)}$ of the conditional variable $X_i|I_i$ from cml, Ex, and Ex^2 by using Equation 4.
10: $mu = \sum mu_i$, $\sigma^2 = \sum \sigma_i^2$
11: $mu^{(t)} = \sum mu_i^{(t)}$, $\sigma^{2^{(t)}} = \sum \sigma^{2}{}_i^{(t)}$
12: $num1 = \mathcal{P}(|\mathcal{N}(mu, \sigma^2) - N| \leq 0.5)$
13: $num2 = \prod(cml_i)$
14: $den = \mathcal{P}(|\mathcal{N}(mu^{(t)}, \sigma^{2^{(t)}}) - N| \leq 0.5)$
15: return $num1 \cdot num2/den$

5.2. Deriving μ_ℓ-Expectation

Define random variable W_i as follows;

$$W_i = \begin{cases} 0, & I_i = 1; \\ X_i, & \text{otherwise.} \end{cases}$$

In other words, W_i holds the number of diversity outliers in bucket b_i. Then the total number of outliers is the sum of all W_i. $\mathcal{E}_{\mu_\ell}$ takes the following form:

$$\mathcal{E}_{\mu_\ell} = E\left(\sum_i W_i\right) = \sum_i E(W_i).$$

Again $E(W_i)$ can be calculated by conditioning on X_i:

$$\begin{aligned} E(W_i) &= \sum_x x \cdot \mathcal{P}(X_i = x; I_i = 0) \\ &= \sum_x x \cdot \mathcal{P}(X_i = x) \cdot (1 - \mathcal{P}(I_i|X_i = x)). \end{aligned} \tag{7}$$

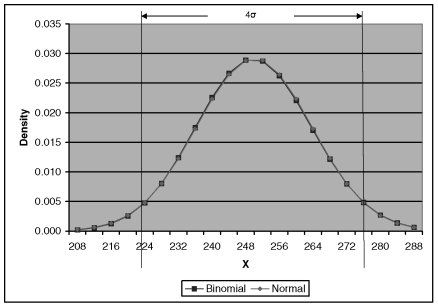
The right-hand side of Equation (7) involves the conditional ℓ-diversity probability and can be calculated from Equation (6). Note that unlike k-anonymity, μ_ℓ-expectation for ℓ-diversity is an approximation. In Algorithm 6, we show the pseudocode for the computation of μ_ℓ-expectation. The algorithm makes a call to Algorithm 3 in each iteration. The time complexity is also $O(n \cdot m \cdot \frac{N^2}{\ell})$.

6. OPTIMIZATIONS

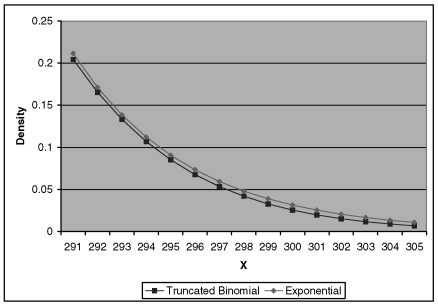
As mentioned before, calculating μ_ℓ-probability and μ_ℓ-expectation of ℓ-diversity through Algorithm 5 is quadratic in the size of the dataset. This may not be acceptable for large datasets. In this section, we present several optimizations that have a huge

ALGORITHM 6: μ_ℓ-Expectation for ℓ-Diversity

Require: Same as in Algorithm 5.
1: **for all** $b_i \in B$ **do**
2: $Ex = 0$.
3: **for all** $x \in [1, N]$ **do**
4: use Algorithm 3 to calculate $\mathcal{P}(I_i|X_i = x)$ as in line 4 of Algorithm 5.
5: $Ex+ = x \cdot P(X_i = x) \cdot (1 - \mathcal{P}(I_i|X_i = x))$
6: return Ex.



(a) Normal Approximation to Binomial



(b) Exponential Approximation to Truncated Binomial

Fig. 2. Approximations to binomial.

effect on the execution time of Algorithm 5. Note that Algorithm 5 makes $N \cdot n$ calls to Algorithm 3 which approximates μ_k-probability of k-anonymity. Some of the optimizations in this section are directed to Algorithm 3 thus are also applicable in k-anonymity domain.

6.1. Approximating the Truncated Binomials

In Algorithm 3, the inner-most loop (lines 3–6) calculates the mean and the variance of the truncated binomial $X|X > k$. We require such a loop because the moments of the truncated binomial have no closed forms. Thus, we introduce an overhead of k binomial computations each of which is not efficient to compute. Next two subsections cover how we can avoid this overhead.

6.1.1. Normal Approximation to Binomial Distribution. Fortunately, when the binomial density function is not too skewed, $\mathcal{B}(N, p)$ can successfully be approximated by the normal distribution $\mathcal{N}(mu, \sigma^2)$ where $mu = N \cdot p$ and $\sigma^2 = N \cdot p \cdot (1 - p)$. Specifically, we can almost safely use a normal approximation when $(mu + 3\sigma) \in [0, N]$. The advantage of using a normal approximation is two fold. First, calculating the normal cumulative functions are faster than calculating binomials. And more importantly, moments of the truncated normal distribution are very well tabulated, thus can be calculated without iterating a loop [Barr and Sherrill 1999].

We show in Figure 2(a) the approximation of the binomial variable $X \sim \mathcal{B}(1000, 0.25)$. We experimentally show in Section 7 that the normal approximation is one of the major factors in making the proposed techniques practical.

6.1.2. Exponential Approximation to the Truncated Binomial. When we have a skewed density function, a normal approximation to a binomial does not guarantee accurate results. For these cases, we observe the following for a random variable $X \sim \mathcal{B}(N, p)$:

For $x \geq k \geq (N \cdot p)$;

$$\frac{\mathcal{P}(X = x+1 | X \geq k)}{\mathcal{P}(X = x | X \geq k)} = \frac{n-x}{x-1}\frac{p}{1-p}.$$

As we get far away from the mean, the density of the truncation diminishes with a variable factor. This very much resembles an exponential distribution. The difference is that for an exponential distribution, the diminishing factor is constant. Given that Z is an exponential variable with parameter λ, we have:

$$\frac{\mathcal{P}(x+1 \leq Z \leq x+2)}{\mathcal{P}(x \leq Z \leq x+1)} = \frac{-e^{-\lambda(x+2)} + e^{-\lambda(x+1)}}{-e^{-\lambda(x+1)} + e^{-\lambda x}} = e^{\lambda}.$$

To approximate $X|X \geq k$, we set $e^{\lambda} = \frac{n-k}{k-1}\frac{p}{1-p}$ and solve for λ which serves as the parameter for an exponential approximation. Note that this approximation is accurate for small x values (those x values for which the diminishing rate of the truncation is not too small compared to that of the approximation). This is generally good enough for an approximation. The reason is that the majority of the probability mass in a truncated binomial (or an exponential variable) is concentrated over small x values. Thus, deviation of the approximation from the truncation over big x values is not significant. We show in Figure 2(b) the approximation of the truncated variable $X|X \geq (mu + 3\sigma)$ where $X \sim \mathcal{B}(1000, 0.25)$.

As in the previous section, the moments of exponential random variables have closed forms and they are very efficient to compute without iterating over all possible x values. We experimentally see in Section 7 that the exponential approximation introduces little or no accuracy loss but provide a major speed-up.

6.2. Pruning the Long Tail of a Binomial

In lines 3–7 of Algorithm 5, we iterate over each value $x \in [1 - N]$, to update the moments of the random variable $X_i|I_i$ (see Section 5). However, each iteration makes a call to Algorithm 3 which has a complexity of $n \cdot N/\ell$. We can avoid most of these costly calls with the following observation.

Note that in lines 5, 6, and 7, the returned value from Algorithm 3 ($\mathcal{P}(I_i|X_i = x)$) is multiplied by $\mathcal{P}(X_i = x)$ where X_i is a binomial. However, no matter what the value of $\mathcal{P}(I_i|X_i = x)$ is, if $\mathcal{P}(X_i = x)$ is too small, the current iteration will contribute little to the moments. We can use the following inequalities to bound $\mathcal{P}(X_i = x)$.

By Hoeffding's inequality, for $X \sim \mathcal{B}(n, p)$ with $\mu = n \cdot p$, $\sigma^2 = n \cdot p \cdot (1 - p)$:

$$\mathcal{P}(|X - mu| \geq k\sigma) \leq 2e^{-2p(1-p)k^2}.$$

By the Chebyshev inequality,

$$\mathcal{P}(|X - mu| \geq k\sigma) \leq \frac{1}{k^2}.$$

Plugging the numbers in Hoeffding's, we see that a binomial $X \sim \mathcal{B}(N, 0.25)$ deviates 4σ away from the mean with less than probability 0.0045. And no matter what parameters we use, by Chebyshev, this probability is no bigger than 0.0625.

So skipping those points that are, say, 4σ away from the mean would have a negligible effect on the computation of the moments. By doing so however, we would achieve a major speed up. Instead of iterating the loop in lines 3–7 of Algorithm 5 N times, we would just perform $8\sigma = 8\sqrt{Np(1-p)} \leq 4\sqrt{N}$ iterations. For example, in Figure 2(a) where $N = 1000$, $p = 0.25$; we have $8\sigma \simeq 110$. This optimization effectively reduces the complexity of Algorithms 5 and 6 to $O(N\sqrt{N})$. In Section 7, we experimentally show that pruning the long tail of a binomial indeed provides a huge speed up without any major effect.

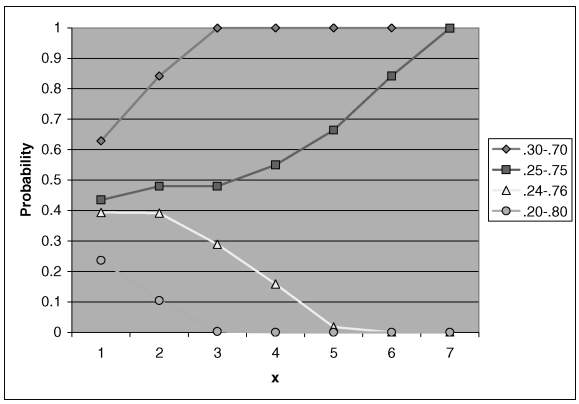


Fig. 3. Conditional ℓ-diversity probabilities of four different sets of likelihoods with varying data size $N = 10 \cdot 5^x$.

6.3. Limit of the Conditional ℓ-Diversity Probability

In lines 3–7 of Algorithm 5, for each x value, we calculate $\mathcal{P}(I_i|X_i = x)$; in words we calculate the conditional ℓ-diversity probability; the probability that a given equality group (i.e. bucket) of size x respects ℓ-diversity given a set of sensitivity likelihoods. As also mentioned above this is done by calling Algorithm 3 and is not a cheap operation. We now try to reduce the number of these calls by observing the limit of $\mathcal{P}(I_i|X_i = x)$.

THEOREM 6.1. *Let $S' = \{s'_1, \ldots, s'_n\}$ be the set of sensitive values and $L' = \{\ell'_1, \ldots, \ell'_n\}$ be the associated sensitivity likelihood set with the highest sensitivity likelihood ℓ'_{max}. As in Section 5.1, let X be the random variable for the total frequencies of s'_is in a given bucket (i.e., size of the bucket) and I be the event that bucket respects ℓ-diversity. Then, as x goes to infinity, the following holds:*

—If $\ell'_{max} \leq \frac{1}{\ell}$ then $\mathcal{P}(I|X = x)$ is monotonically increasing and $\mathcal{P}(I|X = x) \rightarrow 1$.
—Else $\mathcal{P}(I|X = x)$ is monotonically decreasing and $\mathcal{P}(I|X = x) \rightarrow 0$.

PROOF. Let U_i be the random variable for the frequency of sensitive value s'_i. Recall that by Equation 5, $\mathcal{P}(I|X = x) = \mathcal{P}(\bigcap_i (\frac{U_i}{x} \leq \frac{1}{\ell}) \mid \sum_i U_i = x)$. As also mentioned in Section 4.1, by the tuple independence assumption, each $U_i \sim \mathcal{B}(x, \ell'_i)$ with mean $x \cdot \ell'_i$ and variance $x \cdot \ell'_i \cdot (1 - \ell'_i)$. Thus, $\frac{U_i}{x}$ has mean ℓ'_i and variance $\frac{\ell'_i \cdot (1-\ell'_i)}{x}$. As x goes to infinity, variance becomes zero and the mean remains unchanged. In other words; in an infinitely large bucket, the rate of the frequency of each sensitive value s'_i over the total size of the bucket is *exactly* ℓ'_i. Thus, if $\ell'_{max} \leq \frac{1}{\ell}$, the bucket is ℓ-diverse with 100% confidence. Otherwise, the bucket is most definitely not ℓ-diverse. □

Theorem 6.1 basically states when we iterate over increasing x values, we can derive the limit of the conditional ℓ-diversity probability without calculating it. To do this, we compare the maximum sensitivity likelihood with the ℓ parameter. In Figure 3, we set $\ell = 1.33$ and show the conditional ℓ-diversity probabilities of four sets of likelihoods with increasing data size. We observe that even though the set of likelihoods (0.75,0.25) barely satisfy ℓ-diversity (i.e., $0.75 < \frac{1}{1.33}$), the conditional probability converges to one. Similarly, (0.76,0.24) barely violates ℓ-diversity (i.e., $0.76 > \frac{1}{1.33}$), the conditional probability converges to zero. How fast the probabilities converge depends on how much bigger the maximum sensitivity likelihood is than the other likelihoods.

We can exploit such a limit and reduce the number of calls to Algorithm 4 with the following approach. As we calculate the conditional ℓ-diversity probability in each iteration, if the probability ever becomes more than a threshold (i.e., 0.99) or less than a threshold (i.e., 0.01), we can stop further calls to Algorithm 4 and assume the threshold value as the conditional ℓ-diversity probability in future iterations. Doing so would introduce a bounded error (i.e., an error no more than 0.01 in this case) that can be managed through the selection of the threshold. We show in Section 7 that limiting the conditional probability has a considerable effect on the execution time.

6.4. Overwhelming Maximum Likelihood

Recall from Equation (5) that $\mathcal{P}(I|X=x) = \mathcal{P}(\bigcap_i (U_i \leq \frac{x}{\ell}) \mid \sum_i U_i = x)$. This probability is not easy to calculate because we deal with many U_i variables all of which are dependent through the conditional part. As also mentioned in Section 5.1, if we were to deal with only one variable, the conditional would reduce to a binomial cumulative ($\mathcal{P}(U_i \mid \sum_i U_i = x) \sim \mathcal{B}(x, \ell_i')$). The next theorems allow us to make such a reduction.

THEOREM 6.2. *Let X, Y be two random variables with $\mathcal{P}(X < a) \leq \varepsilon_1$ and $\mathcal{P}(Y > a) \leq \varepsilon_2$ where a is a fixed number and $0 \leq \varepsilon_1, \varepsilon_2 \leq 1$. Then $\mathcal{P}(Y > X) \leq \varepsilon_1 + \varepsilon_2$.*

PROOF.

$$\begin{aligned}\mathcal{P}(Y > X) &\leq \mathcal{P}(X < a \ \vee \ Y > a)\\ &= \mathcal{P}(X < a) + \mathcal{P}(Y > a) - \mathcal{P}(X < a \ \wedge \ Y > a)\\ &\leq \mathcal{P}(X < a) + \mathcal{P}(Y > a) \leq \varepsilon_1 + \varepsilon_2.\end{aligned}$$

□

Basically, if a binomial U_i rarely goes below a threshold a and another binomial U_j rarely goes above a, then $\mathcal{P}(U_i > U_j)$ should be big.

THEOREM 6.3. *Let $X, Y_1, \ldots, Y_n$ be $n+1$ random variables with $\mathcal{P}(Y_i > X) \leq \varepsilon_i$. Then $\mathcal{P}(\bigcap_i (X \geq Y_i)) \geq 1 - \sum_i \varepsilon_i$.*

PROOF.

$$\begin{aligned}\mathcal{P}\left(\bigcap_i (X \geq Y_i)\right) &= 1 - \mathcal{P}\left(\bigcup_i (Y_i > X)\right)\\ &= 1 - \left(\sum_i \mathcal{P}(Y_i > X\right) - \sum_{i,j} \mathcal{P}(Y_i > X \wedge Y_j > X)\\ &\quad + \sum_{i,j,k} \mathcal{P}(Y_i > X \wedge Y_j > X \wedge Y_k > X) + \cdots)\\ &\geq 1 - \sum_i \mathcal{P}(Y_i > X) \geq 1 - \sum_i \varepsilon_i\end{aligned}$$

□

In words, if a random variable X is bigger than all random variables Y_i ($i \in [1 - n]$) with overwhelming probability, then X is quite likely the maximum of the set $X, Y_1, \ldots, Y_n$.

We benefit from these two theorems as follows. Among the set of random variables $SU = \{U_{max}, U_1, \ldots, U_n\}$ that describe the frequency of sensitive values, suppose U_{max} has the highest sensitivity likelihood. We first check for all $i \in [1-n]$ and some negligible ε_1 and ε_2, if there exists a number th such that $\mathcal{P}(U_{max} < th) < \varepsilon_1$ and $\mathcal{P}(U_i > th) < \varepsilon_2$. We can perform such a check by making use of the discussion in Section 6.2. (I.e., th can be picked such that $th = mu - 4\sigma$ where mu and σ are the expectation and standard

deviation of U_{max} respectively.) If we can pass the check, by Theorem 6.2, we will have proved $\mathcal{P}(U_{max} < U_i) < \varepsilon_i$ for all $i \in [1-n]$ and some negligible ε_i. Then by Theorem 6.3, U_{max} is most likely to be the maximum frequency in the set. Thus, we can check the ℓ-diversity condition only on U_{max}: $\mathcal{P}(\bigcap_{U \in SU}(U \leq \frac{x}{\ell}) \mid \sum_{U \in SU} U = x) \simeq \mathcal{P}(U_{max} \leq \frac{x}{\ell} \mid \sum_{U \in SU} U = x)$. Note that $(U_{max} \mid \sum_{U \in SU} U = x) \sim \mathcal{B}(x, \ell'_{max})$ meaning we do not need to call Algorithm 3 to calculate the conditional ℓ-diversity problem.

As the conditional ℓ-diversity problem is no different then the μ_k-probability problem, the optimization given in this section can also be used to speed up the instant k-anonymity algorithm as well.

7. EXPERIMENTS

This section presents the experimental evaluation of the algorithm S-INSTANT. Recall from Section 3 that the algorithm first selects a candidate list of generalization mappings based on either μ-probability or μ-expectation. This phase is performed by utilizing a summary structure and does not require data access. Then the algorithm applies the mappings to the dataset one by one and validates if the privacy constraints are satisfied. The performance of the algorithm depends on how many validation steps need to be performed before a privacy preserving mapping is found. We also presented several optimizations for S-INSTANT in Section 6. In this section, we respectively use the names HYBRID, EXPO, SD_CUT, LIMIT, and MAX for the optimizations given in Sections 6.1.1, 6.1.2, 6.2, 6.3, and 6.4. We also present a brief analysis of the M-INSTANT algorithm.

During our experiments, we use the real datasets[5] that were extracted from CENSUS data and previously used by [Xiao and Tao 2006a; Ghinita et al. 2007]. The set consists of 5 similar datasets with varying cardinalities, same QI-attributes and the same sensitive attribute. QI-attributes in the datasets are *Age*, *Gender*, *Education*, *Marital Status*, *Race*, *Work Class*, *Native Country* and the sensitive attribute is *Salary*. We run S-INSTANT with both histograms and BN structures. While histograms are generated by the algorithm, we assume BN structures are available as a DBMS resource. Thus, execution times include the construction costs for histograms but not for BN structures. For each dataset, we create a BN structure by using the BANJO[6] tool and selecting the simulated annealing-based method. Resulting BN structure of the dataset with cardinality 100k can be seen in Figure 4(b). We conducted our tests on a Core2Duo 2.0GHz powered system with 2GB memory, running Linux and the algorithm is implemented in Java. Table IV summarizes the parameters of our experiments and their corresponding values.

We implemented two versions of S-INSTANT that searches the space of full domain generalizations (as in Incognito [LeFevre et al. 2005]) and single dimensional algorithms (as in k-Optimize and ℓ-Optimize [Bayardo and Agrawal 2005]). We compare S-INSTANT with the optimal full domain k-anonymity (or ℓ-diversity) algorithm, Incognito and a DGH version of the optimal single dimensional algorithm k-Optimize (or ℓ-Optimize). We use in-memory implementations of Incognito, k-Optimize and S-INSTANT. Among this algorithms, the original Incognito uses bottom-up pruning (checking starts at the bottom of the generalization lattice and upper nodes are pruned whenever k-anonymity is satisfied for the current generalization mapping). In order to optimize Incognito for an in-memory environment, we modify the algorithm so that top-down pruning is performed (see Section 3.2). As noted in Fung et al. [2005] and Nergiz et al. [2007], a top-down Incognito prunes much better than its bottom-up version. (As an example for the dataset we used, a top-down Incognito prunes around

[5]Available at www.cse.cuhk.edu.hk/∼taoyf/paper/vldb06.html
[6]http://www.cs.duke.edu/∼amink/software/banjo/

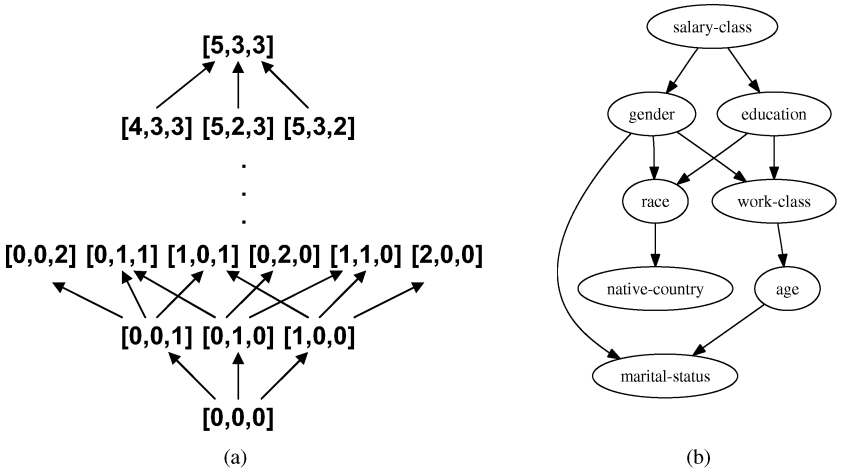


Fig. 4. (a) Generalization Lattice, (b) BN structure of the dataset with $N = \mathbf{100}$k.

Table IV. Parameters and Tested Values

Parameter	Values
k	20, 40, 60, 80, 100
ℓ	2, 3, 4, 5, 10
cardinality N	100k, 200k, 300k, 400k, 500k
number of QI-attributes d	3, 4, 5, 6, 7
suppression rate SR	1%, 2%, 3%, 4%, 5%
optimization OPT	MAX, LIMIT, EXPO
μ-Probability threshold th	0.1, 0.3, 0.5, 0.7, 0.9

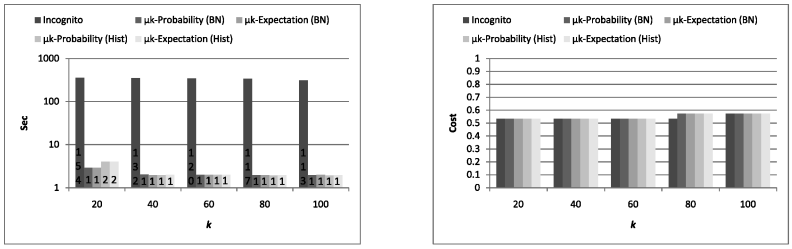
2000 of 2150 possible generalizations, while bottom-up version prunes only around 150.) The comparison is carried out in terms of both execution time and utility of the anonymization. We use the LM cost metric [Iyengar 2002] (see Section 2) to quantify utility. Note that S-INSTANT can be run to adopt either μ-probability or μ-expectation as the pruning criteria. In this section when we say μ-probability or μ-expectation, we refer to these two versions of S-INSTANT.

One important point to make is the following: As mentioned before, we use in-memory implementations of Incognito, k-Optimize and S-INSTANT. This unfairly favors Incognito and k-Optimize. As also reported in LeFevre et al. [2005], a database implementation of Incognito requires minutes/hours to execute as opposed to the seconds we report in the figures[7]. The relative performance of S-INSTANT we present in this section is certainly a lower bound on the actual performance we would see in a database implementation. Having said that, we also show the number of data scans required by both algorithms inside the bars in the figures.

7.1. *k*-Anonymity Experiments

For μ_k-probability experiments, we use the optimization HYBRID. We select 0.8 as the threshold value for S-INSTANT. Recall that S-INSTANT prunes any mapping in the generalization lattice with μ_k-probability (probability that mapping will produce a k-anonymous generalization) less than the threshold.

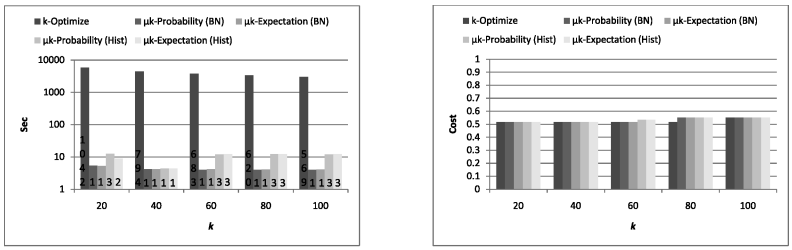
[7]The execution times in the figures belong to an in-memory implementation. However, S-INSTANT requires very few data scans in all experiments, therefore does not suffer much from additional IO costs. There is little to no difference in speed between an in-memory implementation of S-INSTANT and its database version.



(a) Execution Time vs k; $N = 500$k (b) Loss Metric vs k; $N = 500$k

(c) Execution Time vs N; $k = 80$ (d) Loss Metric vs N; $k = 80$

Fig. 5. k-Anonymity experiments: Incognito vs μ-Probability vs μ-Expectation when k and N vary.



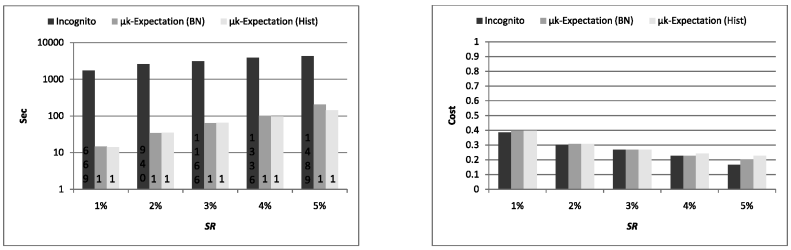
(a) Execution Time vs k; $N = 500$k (b) Loss Metric vs k; $N = 500$k

(c) Execution Time vs N; $k = 80$ (d) Loss Metric vs N; $k = 80$

Fig. 6. k-Anonymity experiments: k-Optimize vs μ-Probability vs μ-Expectation when k and N vary.

For μ_k-expectation, we use the only applicable optimization SD_CUT. We select 1 as the threshold value. That is, S-INSTANT prunes any mapping with μ_k-expectation (expected number of outliers violating k-anonymity) more than 1.

In Figures 5(a), 5(b), 6(a), and 6(b) we evaluate the effect of varying k. In terms of speed, we see that both μ_k-probability and μ_k-expectation based S-INSTANT outperform Incognito and k-Optimize by a factor of at the least 100. As expected, if we increase the value of k, Incognito and k-Optimize run faster since less number of nodes are being visited in the generalization lattice by the algorithm (search space of the



(a) Execution Time vs SR; $k = 80, N = 500$k (b) Loss Metric vs SR; $k = 80, N = 500$k

Fig. 7. k-Anonymity experiments: Incognito vs μ-Expectation when SR varies.

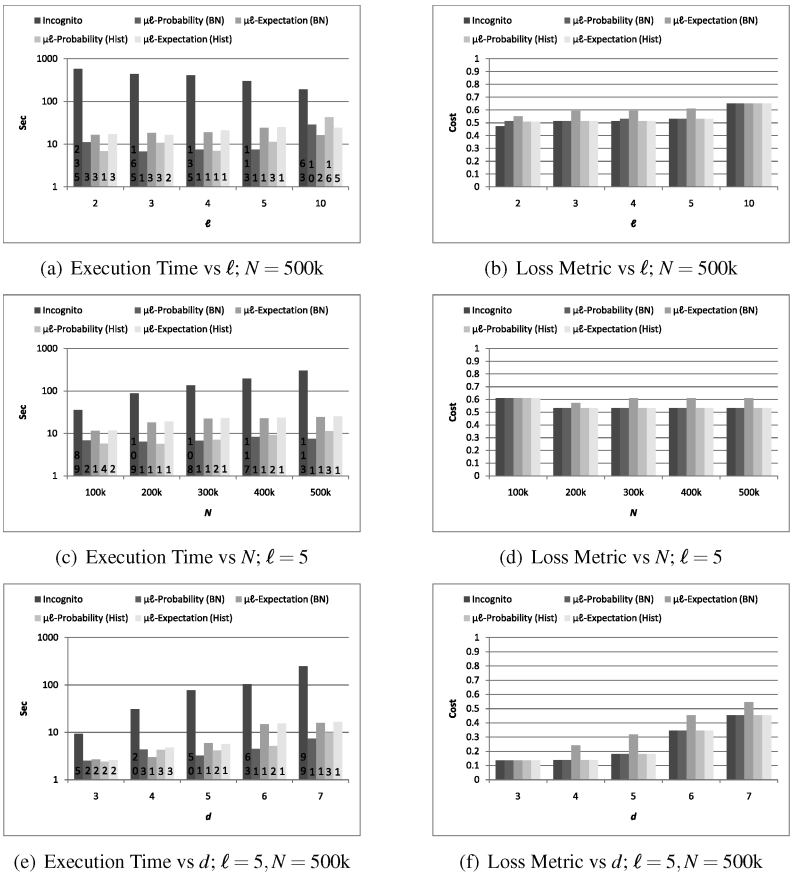
algorithm decreases). In Section 4.1, we showed that the execution time of μ_k-probability and μ_k-expectation correlates with k ($O(n \cdot k)$). However, as k increases, fewer calls to μ_k-probability and μ_k-expectation are made (more nodes are pruned) and execution times appear to be independent of k. In terms of data scans, both Incognito and k-Optimize check more than 100 mappings for k-anonymity, thus refer to data more than 100 times. Nearly all versions of S-INSTANT find the k-anonymous mapping in their first (or rarely third) trial. In terms of utility cost, μ_k-probability and μ_k-expectation based S-INSTANT differ from Incognito and k-Optimize by a slight margin only when $k = 60$ and $k = 80$. For other k values, all three algorithms perform the same This means that in most cases, S-INSTANT finds the optimal mapping by making only one scan of data. This surely is the best an anonymization algorithm can do.

In Figures 5(a), 5(d), 5(c), and 6(d) we perform a similar experiment and show the effect of varying dataset cardinality. In terms of speed, we observe that both μ-probability and μ-expectation based S-INSTANT outperform Incognito, k-Optimize by a factor of at the least 65. While all algorithms require more time as the cardinality increases, S-INSTANT scales much better than Incognito and k-Optimize. Note again that S-INSTANT finds the optimal mapping in most cases.

As mentioned in Nergiz and Clifton [2007], and LeFevre et al. [2005] and also in Section 2, allowing single dimensional algorithms to suppress some of the tuples reduces the negative effect of outliers on the utility. We also run both algorithms such that they tolerate a certain number of tuples to be fully suppressed. The effect of tuple suppression to the execution time and the utility cost of the anonymization is shown in Figures 7(a) and 7(b). Note that a μ_k-probability based S-INSTANT cannot capture suppression tolerance but μ_k-expectation does. So we include only the μ_k-expectation test results and set the threshold as the suppression tolerance rate. In terms of speed, we see that μ_k-expectation based S-INSTANT outperforms Incognito by a factor of 15 at the least. In terms of utility, both algorithms perform similar, the largest LM difference is 0.06 which occurs when suppression rate is 5% and when we use histograms. As was foreseen, both algorithms return more utilized (more specific) mappings. This increases the number of data accesses Incognito needs to make to thousands. This number remains to be one for S-INSTANT.

Even though we do not present the results, we also observed the effects of a varying threshold value. Interestingly, the k-anonymity algorithm S-INSTANT is insensitive to a wide range of changes in the threshold value. For example, we get the same utility and execution time with the thresholds 0.1 or 0.9. (For all generalizations in the lattice, we have either $\mathcal{P}_{\mu_k} < 0.1$ or $\mathcal{P}_{\mu_k} > 0.9$) This implies that the selection of the threshold value is not a real burden on the data anonymizer.[8]

[8]This may not be true for some datasets with large domains and with DGHs of large heights. In those cases, the search space is bigger and it is more likely to find generalizations with border line μ-probabilities.



(a) Execution Time vs ℓ; $N = 500$k

(b) Loss Metric vs ℓ; $N = 500$k

(c) Execution Time vs N; $\ell = 5$

(d) Loss Metric vs N; $\ell = 5$

(e) Execution Time vs d; $\ell = 5, N = 500$k

(f) Loss Metric vs d; $\ell = 5, N = 500$k

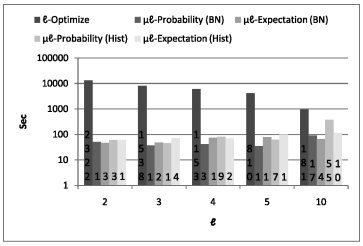
Fig. 8. ℓ-Diversity experiments: Incognito vs μ-Probability vs μ-Expectation when ℓ, N and d vary.

7.2. ℓ-Diversity Experiments

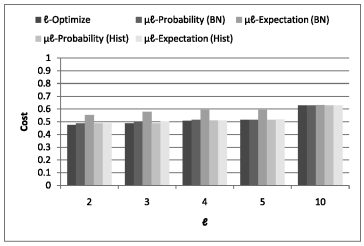
We compare the ℓ-diversity version of S-INSTANT with the optimal full domain and single-dimensional ℓ-diversity algorithms Incognito and ℓ-Optimize. Except one experiment where the effect of using different combinations of optimizations is observed, we use all five optimizations in both μ_ℓ-probability and μ_ℓ-expectation based S-INSTANT executions.

In Figures 8 and 9 we evaluate the effects of varying ℓ, N and d. In terms of execution time, we observe that S-INSTANT algorithms perform much better than Incognito and ℓ-Optimize algorithm in all three experiments (at times by a factor of 20). We see that μ_ℓ-probability based S-INSTANT is generally faster than μ_ℓ-expectation version. In terms of utility, μ_ℓ-probability generally performs very similar to Incognito and ℓ-Optimize whereas μ_ℓ-expectation slightly performs worse than the other two. One interesting observation is the following: Even though the complexity of μ_ℓ-probability and μ_ℓ-expectation depends on the data size, we see in Figure 8(c) that the execution times of S-INSTANT algorithms do not seem to change with increasing data size. This is the case even when the algorithm returns the same output. The reason for this is that the optimizations being used hide the effect of data size to time complexity.

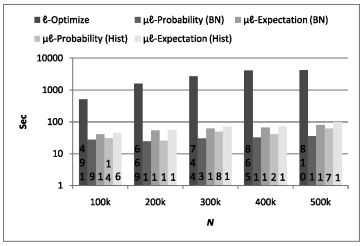
In Figures 10(a) and 10(b), we show how different combinations of optimizations affect execution time and utility of the anonymization. Even though we do not present



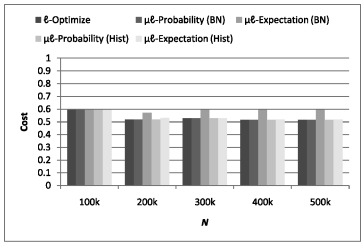
(a) Execution Time vs ℓ; $N = 500$k



(b) Loss Metric vs ℓ; $N = 500$k

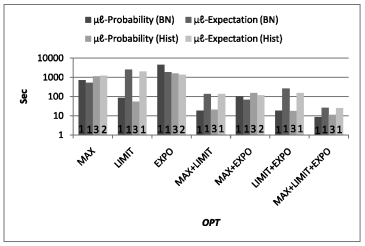


(c) Execution Time vs N; $\ell = 5$

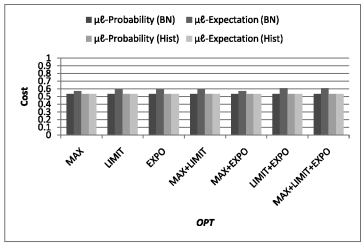


(d) Loss Metric vs N; $\ell = 5$

Fig. 9. ℓ-Diversity experiments: ℓ-Optimize vs μ-Probability vs μ-Expectation when ℓ and N vary.



(a) Execution Time vs OPT; $\ell = 5, N = 500$k



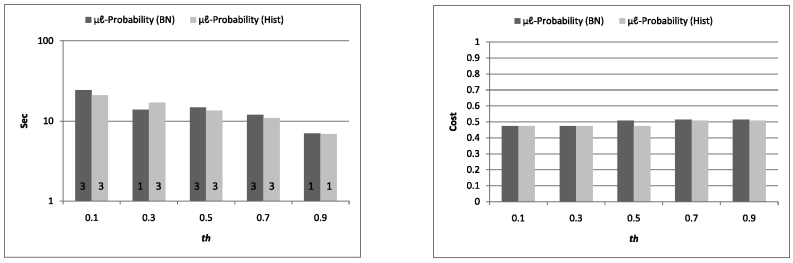
(b) Loss Metric vs OPT; $\ell = 5, N = 500$k

Fig. 10. ℓ-Diversity experiments: μ-Probability vs μ-Expectation when OPT varies.

results here, we observe that the optimizations HYBRID and SD_CUT have a huge effect on the execution time. Without any of these optimizations, S-INSTANT does not terminate in a reasonable time. So, we fix these optimizations, and try to evaluate the effects of the others. We see that S-INSTANT runs fastest when all optimizations are in place. When we remove any one of the optimizations, the execution time drops meaning that each optimization is effective independent of the others. We also observe for μ_ℓ-probability and μ_ℓ-expectation based S-INSTANT that using different combinations of optimizations result in little or no difference in utility. We conclude that optimizations greatly speed up the execution times at the cost of negligible utility loss.

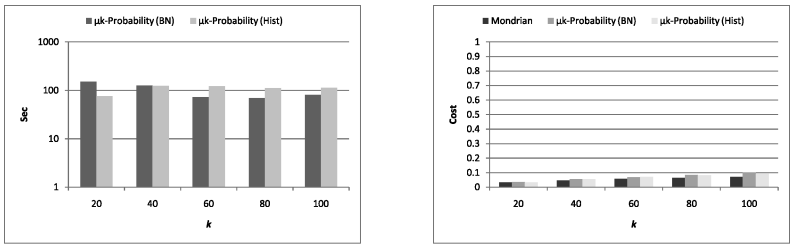
In Figures 11(a) and 11(b), we evaluate the effects of varying threshold value th. As noted before, a big th allows the algorithm to prune a lot but might cause the algorithm to miss the optimal mapping while the algorithm with a smaller th is more likely to find the optimal solution at the cost of higher execution times. While we see this effect in our experiments, the deviations in execution time and utility are not significant. In other words, the algorithm is not too sensitive to the selection of the threshold value.

Also note the number of data accesses required for the algorithms. Neither of the versions of S-INSTANT makes more than three data access with BN structures and more



(a) Execution Time vs th; $\ell = 2, N = 500$k (b) Loss Metric vs th; $\ell = 2, N = 500$k

Fig. 11. ℓ-Diversity experiments: μ-Probability when threshold th varies.



(a) Execution Time vs k; $N = 500$k (b) Loss Metric vs k; $N = 500$k

Fig. 12. k-Anonymity experiments: Mondrian vs μ-Probability when k varies.

than 55 data accesses with histograms. The previously proposed ℓ-diversity algorithm can go as high as thousands.

Lastly, we want to compare the two versions of S-INSTANT that use BN structures and histograms. Generally, BN structures are more accurate than histograms and as can be seen from k-anonymity and ℓ-diversity experiments, this slightly affects the performance of the algorithm S-INSTANT. However, in most cases the extra utility and efficiency we get by using BN structures are not very significant. Also note that constructing BNs from scratch is a costly operation, thus the algorithm can benefit from BN structures only if they are supplied by the DBMS. This makes the histogram version of S-INSTANT an attractive option for an efficient single-dimensional algorithm.

7.3. Experiments on Multidimensional Instant Algorithm

In this subsection, we compare a database implementation of M-INSTANT algorithm with a database implementation of the multidimensional Mondrian algorithm [LeFevre et al. 2006] in terms of efficiency and utility. Note again that the first phase of M-INSTANT algorithm does not access database. Mondrian algorithm and the second phase of M-INSANT algorithm use count queries on partitions of database to check for k-anonymity property. Using Oracle 10g as the DBMS, we perform the experiments on the same dataset with a threshold 0.95 and run M-INSTANT with both histograms and BN structures. We only use the version of M-INSTANT that is based on μ_k-probability. We leave a more detailed analysis of the approach as a future work.

For all k values listed in this section, a database implementation of Mondrian takes more than 30 minutes to execute.[9] As M-INSTANT runs much faster, in Figure 12(a),

[9]An in-memory implementation of Mondrian takes around 10-20 seconds to anonymize the same dataset. Note that M-INSTANT does not need to be faster than an in-memory version as it does not require that data fits in the memory.

Table V. k-Anonymity Experiments: μ-Probability when Threshold th Varies

th	0.1	0.3	0.5	0.7	0.9
LM Cost	0.084	0.084	0.084	0.084	0.084
Time (sec)	299	295	281	271	134

we include the execution times only for M-INSTANT. As can be seen from the figure, M-INSTANT is at least 25 times faster than Mondrian.

In Figure 12(b), we compare the multidimensional algorithms with respect to the utility of the anonymizations. While both M-INSTANT algorithms fail to find the same generalization as the Mondrian does, the difference in costs of both anonymizations is no bigger than 0.03. This implies that the multidimensional instant algorithms are reasonably successful. We also observe almost the same performance for the two versions of M-INSTANT algorithms that run with histograms and BN structures. Thus, even a simple summary structure is enough for a successful analysis of multidimensional k-anonymity. In Table V, we show the performance of M-INSTANT when threshold th varies. While utility is not affected, execution time seems to be decreasing with increasing threshold value. This is because a low threshold value results in smaller partitions at the end of the first phase. These partitions violate k-anonymity with high probability, triggering many merge operations in the second phase.

While considerably faster than the database implementation of Mondrian, the new algorithm M-INSTANT is slower than S-INSTANT. The main reason is that with extra flexibility, M-INSTANT generates generalizations with μ-probabilities very close to the threshold value. So it becomes more likely for the partitions in the final generalization to violate the k-anonymity property. In this case, the algorithm needs to access data more often in the second phase.

8. CONCLUSIONS AND FUTURE WORK

Many of the previous anonymization algorithms require a considerable number of data accesses. In this article, we addressed this problem by presenting a new single dimensional algorithm, S-INSTANT for both k-anonymity and ℓ-diversity. S-INSTANT makes use of the available DBMS resources to generate mappings with high μ-probabilities; probability that the mapping would produce a k-anonymous (or ℓ-diverse) generalization. We showed how to approximate such probabilities efficiently. We also presented several optimizations to make the proposed technique practical.

We experimentally showed that S-INSTANT algorithm greatly outperforms the previously proposed single dimensional algorithms in execution time at the cost of little or no utility loss. This is true for both k-anonymity and ℓ-diversity. In most cases, S-INSTANT requires only one scan of data, which is the best any algorithm can achieve. While S-INSTANT requires a threshold parameter to be specified, practically the performance of the algorithm is insensitive to the threshold.

An obvious future study would be to design instant algorithms for other database and data mining applications. As mentioned before, there is strong motivation for designing fast-response algorithms when we need to run them many times perhaps with different parameters (e.g., visualization). A strong candidate for such an application is perhaps clustering. Most often a good parameterization is required to get clusters that are visually relevant. An instant clustering technique (that interacts with a summary model instead of a huge database) would most certainly speed up discovering such parameters. Another future study is a comparative analysis of μ-probability and Bonferroni based predictions [LeFevre et al. 2008]. Bonferroni predictions can also be made based on histograms (or other summary structures) without requiring sampling. As mentioned before, the techniques given in this paper offer more accurate predictions

while Bonferroni based predictions are often more efficient. One can also consider a hybrid approach, in which we start with Bonferroni predictions for the mappings in the upper nodes of the lattice and switch to approximations whenever prediction falls under the threshold. Such an approach would likely benefit from the advantages of both approaches.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

REFERENCES

ABOULNAGA, A. AND CHAUDHURI, S. 1999. Self-tuning histograms: building histograms without looking at data. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data (SIGMOD'99)*. ACM, New York, NY, 181–192.

AGGARWAL, C. C. AND YU, P. S. 2007. On anonymization of string data. In *Proceedings of the SIAM International Conference on Data Mining (SDM'07)*.

AGRAWAL, G., FEDER, T., KENTHAPADI, K., KHULLER, S., PANIGRAHY, R., THOMAS, D., AND ZHU, A. 2006. Achieving anonymity via clustering. In *Proceedings of the 25th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'06)*. 153–162.

BARR, D. R. AND SHERRILL, E. T. 1999. Mean and variance of truncated normal distributions. *Amer. Statist. 53,* 4, 357–361.

BAYARDO, R. J. AND AGRAWAL, R. 2005. Data privacy through optimal k-anonymization. In *Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*. IEEE Computer Society, Los Alamitos, CA, 217–228.

BONCHI, F., ABUL, O., AND NANNI, M. 2008. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Proceedings of the 24nd International Conference on Data Engineering (ICDE'08)*.

BRUNO, N., CHAUDHURI, S., AND GRAVANO, L. 2001. Stholes: a multidimensional workload-aware histogram. *SIGMOD Rec. 30,* 2, 211–222.

BYUN, J.-W., KAMRA, A., BERTINO, E., AND LI, N. 2007. Efficient k-anonymization using clustering techniques. In *Proceedings of the 12th International Conference on Database Systems for Advanced Applications*.

CHEN, B.-C., LEFEVRE, K., AND RAMAKRISHNAN, R. 2007. Privacy skyline: Privacy with multidimensional adversarial knowledge. In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB'07)*. VLDB Endowment, 770–781.

CIRIANI, V., DI VIMERCATI, S. D. C., FORESTI, S., AND SAMARATI, P. 2007. *k*-anonymity. In *Secure Data Management in Decentralized Systems*, 323–353.

CORMODE, G., SRIVASTAVA, D., YU, T., AND ZHANG, Q. 2008. Anonymizing bipartite graph data using safe groupings. *Proc. VLDB Endow. 1,* 1, 833–844.

DOMINGO-FERRER, J. AND TORRA, V. 2005. Ordinal, continuous and heterogeneous k-anonymity through microaggregation. *Data Min. Knowl. Disc. 11,* 2, 195–212.

DU, W., TENG, Z., AND ZHU, Z. 2008. Privacy-MaxEnt: Integrating background knowledge in privacy quantification. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 459–472.

FELLER, W. 1968. *An Introduction to Probability Theory and Its Applications,* Vol. 1. Wiley.

FUNG, B. C. M., WANG, K., AND YU, P. S. 2005. Top-down specialization for information and privacy preservation. In *Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*. IEEE Computer Society, Los Alamitos, CA, 205–216.

GETOOR, L., TASKAR, B., AND KOLLER, D. 2001. Selectivity estimation using probabilistic models. *SIGMOD Rec. 30,* 2, 461–472.

GHINITA, G., KARRAS, P., KALNIS, P., AND MAMOULIS, N. 2007. Fast data anonymization with low information loss. In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB'07)*. VLDB Endowment, 758–769.

GIONIS, A., MAZZA, A., AND TASSA, T. 2008. k-anonymization revisited. In *Proceedings of the IEEE 24th International Conference on Data Engineering (ICDE'05)*. 744–753.

HAY, M., MIKLAU, G., JENSEN, D., TOWSLEY, D. F., AND WEIS, P. 2008. Resisting structural re-identification in anonymized social networks. *Proc. VLDB Endow. 1,* 1, 102–114.

HIPAA 2001. Standard for privacy of individually identifiable health information. *Fed. Register 66,* 40.

HORE, B., CH, R., JAMMALAMADAKA, R., AND MEHROTRA, S. 2007. Flexible anonymization for privacy preserving data publishing: A systematic search based approach. In *Proceedings of the SIAM International Conference on Data Mining*.

IWUCHUKWU, T. AND NAUGHTON, J. F. 2007. K-anonymization as spatial indexing: toward scalable and incremental anonymization. In *Proceedings of the 33rd International Conference on Very large Data Bases (VLDB'07)*. VLDB Endowment, 746–757.

IYENGAR, V. S. 2002. Transforming data to satisfy privacy constraints. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*. ACM, New York, NY, 279–288.

JIANG, W. AND CLIFTON, C. 2006. A secure distributed framework for achieving *k*-anonymity. *VLDB J. (Special Issue on Privacy-Preserving Data Management)*.

KIFER, D. AND GEHRKE, J. 2006. Injecting utility into anonymized datasets. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'06)*. ACM Press, New York, NY, 217–228.

KOOI, R. P. 1980. The optimization of queries in relational databases. Ph.D. thesis, Cleveland, OH.

LAHIRI, S. N., CHATTERJEEA, A., AND MAITI, T. 2007. Normal approximation to the hypergeometric distribution in nonstandard cases and a sub-gaussian berryÃ–esseen theorem. *J. Statist. Plann. Infere. 137,* 11, 3570–3590.

LEE, J.-H., KIM, D.-H., AND CHUNG, C.-W. 1999. Multi-dimensional selectivity estimation using compressed histogram information. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'99)*. ACM, New York, NY, 205–214.

LEFEVRE, K., DEWITT, D. J., AND RAMAKRISHNAN, R. 2005. Incognito: Efficient full-domain k-anonymity. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'05)*. ACM, New York, NY, 49–60.

LEFEVRE, K., DEWITT, D. J., AND RAMAKRISHNAN, R. 2006. Mondrian multidimensional k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*. 25–35.

LEFEVRE, K., DEWITT, D. J., AND RAMAKRISHNAN, R. 2008. Workload-aware anonymization techniques for large-scale datasets. *ACM Trans. Datab. Syst. 33,* 3, 1–47.

LEVIN, B. 1981. A representation for multinomial cumulative distribution functions. *Annals Statis. 9,* 5, 1123–1126.

LI, N. AND LI, T. 2007. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Proceedings of the 23nd International Conference on Data Engineering (ICDE'07)*.

LI, T. AND LI, N. 2006. Optimal k-anonymity with flexible generalization schemes through bottom-up searching. In *IEEE International Workshop on Privacy Aspects of Data Mining (PADM'06)*.

LI, T., LI, N., AND ZHANG, J. 2009. Modeling and integrating background knowledge in data anonymization. *Proceedings of the International Conference on Data Engineering*. 6–17.

MACHANAVAJJHALA, A., GEHRKE, J., KIFER, D., AND VENKITASUBRAMANIAM, M. 2006. ℓ-diversity: Privacy beyond *k*-anonymity. In *Proceedings of the 22nd IEEE International Conference on Data Engineering (ICDE'06)*.

MARKL, V., HAAS, P. J., KUTSCH, M., MEGIDDO, N., SRIVASTAVA, U., AND TRAN, T. M. 2007. Consistent selectivity estimation via maximum entropy. *VLDB J. 16,* 1, 55–76.

MARTIN, D. J., KIFER, D., MACHANAVAJJHALA, A., GEHRKE, J., AND HALPERN, J. Y. 2007. Worst-case background knowledge for privacy-preserving data publishing. In *Proceedings of the 23nd International Conference on Data Engineering (ICDE'07)*.

MATIAS, Y., VITTER, J. S., AND WANG, M. 1998. Wavelet-based histograms for selectivity estimation. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'98)*. ACM, New York, NY, 448–459.

MATIAS, Y., VITTER, J. S., AND WANG, M. 2000. Dynamic maintenance of wavelet-based histograms. In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB'00)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 101–110.

NERGIZ, M. E., ATZORI, M., AND CLIFTON, C. 2007. Hiding the presence of individuals in shared databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'07)*.

NERGIZ, M. E., ATZORI, M., SAYGIN, Y., AND GUC, B. 2009a. Towards trajectory anonymization: A generalization-based approach. *Trans. on Data Privacy 2,* 1.

NERGIZ, M. E., CICEK, E. A., AND SAYGIN, Y. 2009b. A look ahead approach to secure multi-party protocols. Tech. rep. 11593, Faculty of Engineering and Natural Sciences, Sabanci University.

NERGIZ, M. E. AND CLIFTON, C. 2007. Thoughts on k-anonymization. *Data Knowl. Engin. 63,* 3, 622–645.

NERGIZ, M. E. AND CLIFTON, C. 2009. δ-Presence without complete world knowledge. *IEEE Trans. Knowl. Data Engin.*

NERGIZ, M. E., CLIFTON, C., AND NERGIZ, A. E. 2009c. Multirelational k-anonymity. *IEEE Trans. Knowl. Data Engin. 99,* 1.

ØHRN, A. AND OHNO-MACHADO, L. 1999. Using boolean reasoning to anonymize databases. *Artif. Intell. Med. 15,* 3, 235–254.

ORACLE. 2009. Oracle database performance tuning guide, 11g release 2. Tech. rep. Part Number E10821-04, Oracle Corporation.

POOSALA, V., HAAS, P. J., IOANNIDIS, Y. E., AND SHEKITA, E. J. 1996. Improved histograms for selectivity estimation of range predicates. *SIGMOD Rec. 25,* 2, 294–305.

POOSALA, V. AND IOANNIDIS, Y. E. 1997. Selectivity estimation without the attribute value independence assumption. In *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB'97)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 486–495.

SAMARATI, P. 2001. Protecting respondent's identities in microdata release. *IEEE Trans. Knowl. Data Engin. 13,* 6, 1010–1027.

SCHWAGER, S. J. 1984. Bonferroni sometimes loses. *Amer. Statis. 38,* 3, 192–197.

SWEENEY, L. 2002. Achieving *k*-anonymity privacy protection using generalization and suppression. *Int. J. Uncert. Fuzz. Knowl.-Based Syst. 10,* 5.

TERROVITIS, M., MAMOULIS, N., AND KALNIS, P. 2008. Privacy-preserving anonymization of set-valued data. *Proc. VLDB Endow. 1,* 1, 115–125.

WONG, R. C.-W., FU, A. W.-C., WANG, K., AND PEI, J. 2007. Minimality attack in privacy preserving data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB'07)*. VLDB Endowment, 543–554.

WONG, R. C.-W., LI, J., FU, A. W.-C., AND WANG, K. 2006. (α, k)-anonymity: An enhanced k-anonymity model for privacy preserving data publishing. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06)*. ACM, New York, NY, 754–759.

XIAO, X. AND TAO, Y. 2006a. Anatomy: Simple and effective privacy preservation. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB'06)*.

XIAO, X. AND TAO, Y. 2006b. Personalized privacy preservation. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'06)*. ACM Press, New York, NY, 229–240.

ZHONG, S., YANG, Z., AND WRIGHT, R. N. 2005. Privacy-enhancing k-anonymization of customer data. In *Proceedings of the 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'05)*. ACM Press, New York, NY, 139–147.

Received July 2009; revised December 2009, March 2010, July 2010; accepted August 2010