Utility-Based Anonymization Using Local Recoding*

Jian Xu[1] Wei Wang[1] Jian Pei[2] Xiaoyuan Wang[1] Baile Shi[1] Ada Wai-Chee Fu[3]
[1]Fudan University, China, {xujian, weiwang1, xy_wang, bshi}@fudan.edu.cn
[2]Simon Fraser University, Canada, jpei@cs.sfu.ca
[3]The Chinese University of Hong Kong, adafu@cse.cuhk.edu.hk

ABSTRACT

Privacy becomes a more and more serious concern in applications involving microdata. Recently, efficient anonymization has attracted much research work. Most of the previous methods use global recoding, which maps the domains of the quasi-identifier attributes to generalized or changed values. However, global recoding may not always achieve effective anonymization in terms of discernability and query answering accuracy using the anonymized data. Moreover, anonymized data is often for analysis. As well accepted in many analytical applications, different attributes in a data set may have different utility in the analysis. The utility of attributes has not been considered in the previous methods.

In this paper, we study the problem of *utility-based anonymization*. First, we propose a simple framework to specify utility of attributes. The framework covers both numeric and categorical data. Second, we develop two simple yet efficient heuristic local recoding methods for utility-based anonymization. Our extensive performance study using both real data sets and synthetic data sets shows that our methods outperform the state-of-the-art multidimensional global recoding methods in both discernability and query answering accuracy. Furthermore, our utility-based method can boost the quality of analysis using the anonymized data.

Categories and Subject Descriptors:H.2.8 [**Database Applications**]: [Data Mining]

General Terms: Security, Algorithms, Performance

keywords: Privacy preservation, data mining, k-anonymity, utility, local recoding

1. INTRODUCTION

To protect privacy against re-identifying individuals by joining multiple public data sources, k-anonymity was proposed [10, 13]. A data set is k-*anonymous* ($k \geq 1$) if each record in the data set is indistinguishable from at least $(k-1)$ other records within the same data set. The larger the value of k, the better the privacy is protected.

A few k-anonymization algorithms have been developed. Generally, to achieve k-anonymity, those methods generalize or suppress the *quasi-identifier attributes*, which are the minimal set of attributes in the table that can be joined with external information to re-identify individual records.

Information loss is an unfortunate consequence of anonymization. In order to make the anonymized data as useful as possible, it is required to reduce the information loss as much as possible. A few models have been proposed to measure the usefulness of anonymized data. For example, the discernability model [4] tries to minimize the number of tuples that are indistinguishable, as long as they satisfy the k-anonymity requirement.

In this paper, we study the problem of k-anonymization and focus on two interesting issues: *anonymization using heuristic local recoding* and *utility-based anonymization*.

1.1 Global and Local Anonymization

Many recent methods (e.g., [4, 6, 7]) use either *global recoding* or *constrained local recoding*. Global recoding maps a given value in a single domain to another one globally. In constrained local recoding, the data space is partitioned into a set of (non-overlapping) regions, and the anonymization maps all tuples in a region to the same generalized or changed tuple. For example, Figure 1(b) demonstrates a 3-anonymization using global recoding for the table in Figure 1(a), where (age, zipcode) is the quasi-identifier.

In contrast, *(fully) local recoding* maps (non-distinct) individual tuple to generalized tuples. For example, Figure 1(c) shows a 3-anonymization using local recoding of the same table in Figures 1(a). The two identical tuples, $R3$ and $R4$, are mapped to different generalized tuples in local recoding. Clearly, global recoding can be regarded as a specific type of local recoding.

Local recoding may achieve less information loss than global recoding. In our example, the two generalized tuples in global recoding have the sizes of intervals 8 and 5 in age, and 1 and 0 in zipcode, respectively. In local recoding, the sizes of intervals are 6 and 2 in age, and 1 and 2 in zipcode, respectively. By intuition, the smaller the sizes of intervals in the generalized tuples, the less information loss in the anonymization.

Can we use (fully) local recoding to achieve less information loss in anonymization effectively? Generally, optimal k-anonymity is NP-hard [8, 2]. In this paper, we propose two simple yet efficient heuristic algorithms using local recoding for k-anonymization. Our extensive empirical study on both real data sets and synthetic data sets show that our method outperforms the state-of-the-art global and constrained local recoding methods in both the discernability and the accuracy of query answering.

*This research was supported by the Shanghai Raising Star Program Grant 05QMX1405, the National Natural Science Foundation of China Grants 69933010 and 60303008, the National Basic Research Program of China (973) (2005CB321905), the NSERC Grants 312194-05 and 614067, the NSF Grant IIS-0308001, and the RGC Earmarked Research Grant of HKSAR CUHK 4120/05E. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

KDD'06, August 20--23, 2006, Philadelphia, Pennsylvania, USA.

Row-id	Age	Zipcode
$R1$	24	53712
$R2$	25	53711
$R3$	30	53711
$R4$	30	53711
$R5$	32	53712
$R6$	32	53713

(a) The original table.

Row-id	Age	Zipcode
$R1$	[24-32]	[53712-53713]
$R2$	[25-30]	53711
$R3$	[25-30]	53711
$R4$	[25-30]	53711
$R5$	[24-32]	[53712-53713]
$R6$	[24-32]	[53712-53713]

(b) 3-anonymization by global recoding.

Row-id	Age	Zipcode
$R1$	[24-30]	[53711-53712]
$R2$	[24-30]	[53711-53712]
$R3$	[24-30]	[53711-53712]
$R4$	[30-32]	[53711-53713]
$R5$	[30-32]	[53711-53713]
$R6$	[30-32]	[53711-53713]

(c) 3-anonymization by local recoding.

Figure 1: Global recoding and local recoding. The row-ids are for reference only and are not released with the data. Thus, the row-ids are not part of the quasi-identifier.

1.2 Utility-Based Anonymization

Anonymized data is often for analysis and data mining. As well recognized in many data analysis applications, different attributes may have different utility. For example, consider anonymizing a data set about patients for disease analysis. Suppose in order to achieve k-anonymity, we can generalize from a five-digit full zipcode to a four-digit prefix (e.g., from 53712 to $5371*$). Alternatively, we can also generalize attribute age to age groups (e.g., from 23 to $[20, 30]$). In many cases, the age information is critical to disease analysis, while the information loss on the accurate location is often acceptable (a four digit prefix in fact still identifies a relatively local region). Thus, the age attribute has more utility than the zipcode attribute, and should be retained as accurate as possible in anonymization.

Can we make the anonymization utility aware? Utility of attributes has not been considered by previous anonymization methods. In this paper, we propose a model for *utility-based anonymization*. We consider both numeric data and categorical data with and without hierarchies. We present a simple method to specify utility of attributes and push them into the heuristic local recoding anonymization methods. Our experimental results show that the utility-based anonymization improves the accuracy in answering targeted queries substantially.

The rest of the paper is organized as follows. In section 2, we recall the notions related to anonymization, and review the related work. We present our utility specification framework in Section 3. Our heuristic local recoding methods are developed in Section 4. An extensive performance study on both real data sets and synthetic data sets is reported in Section 5.

2. K-ANONYMITY AND RELATED WORK

Consider a table $T = (A_1, \ldots, A_n)$. A *quasi-identifier* is a minimal set of attributes $(A_{i_1}, \ldots, A_{i_l})$ $(1 \leq i_1 < \cdots < i_l \leq n)$ in T that can be joined with external information to re-identify individual records. In this paper, we assume that the quasi-identifier is specified by the administrator based on the background knowledge. Thus, we focus on how to anonymize T to satisfy the k-anonymity requirement.

A table T is said *k-anonymous* given a parameter k and the quasi-identifier $(A_{i_1}, \ldots, A_{i_l})$ if for each tuple $t \in T$, there exist at least another $(k-1)$ tuples $t_1, \ldots, t_{k-1}$ such that those k tuples have the same projection on the quasi-identifier, i.e., $t_{(A_{i_1}, \ldots, A_{i_l})} = t_{1_{(A_{i_1}, \ldots, A_{i_l})}} = \cdots = t_{k-1_{(A_{i_1}, \ldots, A_{i_l})}}$. Tuple t and all other tuples indistinguishable from t on the quasi-identifier form an *equivalence class*. We call the class the *group* that t is generalized.

Given a table T with the quasi-identifier and a parameter k, the problem of *k-anonymization* is to compute a view T' that has the same attributes as T such that T' is k-anonymous and as close to T as possible according to some quality metric. We shall discuss the quality metrics soon.

Since the attributes not in the quasi-identifier do not need to be changed, to keep our discussion simple but without loss of generality, hereafter we consider only the attributes in the quasi-identifier. That is, for table $T(A_1, \ldots, A_n)$ in question, we assume that $(A_1, \ldots, A_n)$ is the quasi-identifier.

Generally, data items are recoded in anonymization. Here, we regard suppression as a specific form of recoding that recodes a data item to null value (i.e., unknown).

Two types of recoding can be used [14]: global recoding and local recoding, as described and demonstrated in Section 1.1. Many previous methods use global recoding and constrained local recoding. In [9, 11], *full-domain generalization*, a type of constrained local recoding, was developed, which maps the whole domain of each quasi-identifier attribute to a more general domain in the domain generalization hierachy. Full-domain generalization guarantees that all values of a particular attribute still belong to the same domain after generalization.

To achieve full-domain generalization, two types of partitioning can be applied. First, single-dimensional partitioning [4, 5] divides an attribute into a set of non-overlapping intervals, and each interval will be replaced by a summary value (e.g., the mean, the median, or the range). On the other hand, (strict) multidimensional partitioning [7] divides the domain into a set of non-overlapping multidimensional regions, and each region will be generalized into a summary tuple.

Generally, anonymization is accompanied by information loss. Various models have been proposed to measure the information loss. For example, the *discernability model* [4] assigns to each tuple t a penalty based on the size of the group that t is generalized, i.e., the number of tuples equivalent to t on the quasi-identifier. That is, $C_{DM} = \sum_{E \in \text{group-bys on quasi-identifier}} |E|^2$.

Alternatively, the *normalized average equivalence class size metric* was given in [7]. The intuition of the metric is to measure how well the partitioning approaches the best case where each tuple is generalized in a group of k indistinguishable tuples. That is, $C_{AVG} = \frac{\text{number of tuples in the table}}{\text{number of group-bys on quasi-identifier} \cdot k}$.

The ideal anonymization should minimize the penalty. However, theoretical analysis [2, 8, 7, 3, 1] indicates that the problem of optimal anonymization under many non-trivial quality models is NP-hard. A few approximation methods were developed [3], such as datafly [12], annealing [15], and Mondrian multidimensional k-anonymity [7]. Some optimal methods [4, 6] with exponential cost in the worst case were shown feasible and often of good performance in practice.

To the best of our knowledge, none of the previous studies address the concern on utilities of attributes systematically.

3. UTILITY-BASED ANONYMIZATION

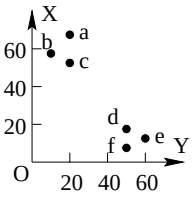

Figure 2: The six tuples in Example 1.

Without loss of generality, in this paper we assume that generalization is used in anonymization. That is, when a tuple is generalized, the ranges of the group of tuples that are generalized are used to represent the generalization, as illustrated in Figure 1. If other representations such as mean or median are used, the definitions can be revised straightforwardly and our methods still work.

In previous methods, the quality metrics, such as the discernability metric and the normalized average equivalence class size metric discussed in Section 2, mainly focus on the size of groups in anonymization. In an anonymized table, when each group of tuples sharing the same projection on the quasi-identifier has k tuples, the penalty metrics are minimized. However, such metrics may not lead to high quality anonymization.

EXAMPLE 1 (QUALITY METRICS). Consider 2-anonymizations for the six tuples shown in Figure 2. (X, Y) is the quasi-identifier. The six tuples can be anonymized in three groups: $\{a, b\}$, $\{c, d\}$, and $\{e, f\}$. In this anonymization scheme, both the discernability metric C_{DM} and the normalized average equivalence class size metric C_{AVG} are minimized.

Suppose each group is generalized using the range of the tuples in the group. That is, a and b are generalized to $([10, 20], [60, 70])$; c and d are generalized to $([20, 50], [20, 50])$; and e and f are generalized to $([50, 60], [10, 15])$.

In order to measure how well the generalized tuples approximate the original ones, for each tuple we can use the sum of the interval sizes on all attributes of the generalized tuple to measure the uncertainty of the generalized tuples. That is, $U(a) = U(b) = 10 + 10 = 20$. Similarly, we get $U(c) = U(d) = 60$ and $U(e) = U(f) = 15$. The total uncertainty of the anonymized table is the sum of the uncertainty of all tuples, i.e., $U(T) = \sum_{t \in T} U(t) = 20 + 20 + 60 + 60 + 15 + 15 = 190$. By intuition, the uncertainty reflects the information loss. The less the uncertainty, the less information is lost.

On the other hand, we may anonymize the tuples in two groups: $\{a, b, c\}$ are generalized to $([10, 20], [50, 70])$, and $\{d, e, f\}$ are generalized to $([50, 60], [10, 20])$. In fact, the data set is 3-anonymous, which is better than 2-anonymous in terms of privacy preservation. Moreover, the total uncertainty in this anonymization is 150, lower than the 2-anonymity scheme.

However, this anonymization scheme has a higher penalty than the 2-anonymous scheme in both the discernability metric C_{DM} and the normalized average equivalence class size metric C_{AVG}. In other words, *optimizing the quality metrics on group size may not always minimize information loss.* ■

A utility-based metric should capture two aspects: *the information loss caused by the anonymization* and *the importance of attributes.* Such utility-aware anonymization may help to improve the quality of analysis afterwards. We introduce the concept of certainty penalty.

First, let us consider the case of numeric attributes. Let T be a table with quasi-identifier $(A_1, \ldots, A_n)$, where all attributes are numeric. Suppose a tuple $t = (x_1, \ldots, x_n)$ is generalized to tuple $t' = ([y_1, z_1], \ldots, [y_n, z_n])$ such that $y_i \leq x_i \leq z_i$ $(1 \leq i \leq n)$. On attribute A_i, the *normalized certainty penalty* is defined as $NCP_{A_i}(t) = \frac{z_i - y_i}{|A_i|}$, where $|A_i| = \max_{t \in T}\{t.A_i\} - \min_{t \in T}\{t.A_i\}$ is the range of all tuples on attribute A_i.

Let each attribute A_i be associated with a weight w_i to reflect its utility in the analysis on the anonymized data. Then, the *weighted certainty penalty* of a tuple is given by $NCP(t) = \sum_{i=1}^{n}(w_i \cdot NCP_{A_i}(t)) = \sum_{i=1}^{n}(w_i \cdot \frac{z_i - y_i}{|A_i|})$.

Clearly, when all weights are set to 1 and all attributes have range $[0, 1]$, the weighted certainty penalty is the L_1 norm distance between points $(\max_{t \in G}\{t.A_1\}, \ldots, \max_{t \in G}\{t.A_n\})$ and $(\min_{t \in G}\{t.A_1\}, \ldots, \min_{t \in G}\{t.A_n\})$, where G is the equivalence group that t belongs to.

Our utility-based metric is given by the total weighted certainty penalty on the whole table. That is, $NCP(T) = \sum_{t \in T} NCP(t)$.

For categorical attributes, distance is often not well defined. That makes measuring utility on categorical attributes difficult. In some previous methods (e.g., [6, 7]), it is assumed that a total order exists on all values in a categorical attribute. In many applications, such an order may not exist. For example, sorting all zipcodes in their numeric order may not reflect the utility properly. Two regions may be adjacent but their zipcodes may not be consecutive.

More often than not, hierarchies exist in categorical attributes. For example, zipcodes can be organized into hierarchy of regions, cities, counties, and states. Let $v_1, \ldots, v_l$ be a set of leaf nodes in a hierarchy tree. Let u be the node in the hierarchy on the attribute such that u is an ancestor of $v_1, \ldots, v_l$, and u does not have any descendant that is still an ancestor of $v_1, \ldots, v_l$. u is called the *closest common ancestor* of $v_1, \ldots, v_l$, denoted by $ancestor(v_1, \ldots, v_l)$. The number of leaf nodes that are descendants of u is called the *size* of u, denoted by $size(u)$.

Suppose a tuple t has value v on a categorical attribute A. When it is generalized in anonymization, the value will be replaced by a set of values $\{v_1, \ldots, v_l\}$, where $v_1, \ldots, v_l$ are the values of tuples on the attribute in the same generalized group. We define the *normalized certainty penalty* of t as $NCP_A(t) = \frac{size(u)}{|A|}$, where $|A|$ is the number of distinct values on attribute A. Here, we assume that each leaf node is of the same importance. The definition can be straightforwardly extended by assigning weights to internal nodes to capture the more important leaf nodes and internal hierarchical structures. Limited by space, we omit the details here.

Putting things together, for a table consisting of both numeric and categorical attributes, the total weighted normalized certainty penalty is the sum of the weighted normalized certainty penalty of all tuples. That is, $NCP(T) = \sum_{t \in T}\sum_{i=1}^{n}(w_i \cdot NCP_{A_i}(t))$, where $NCP_{A_i}(t)$ should be computed according to whether A_i is a numeric or categorical attribute.

Given a table T, a parameter k, the weights of attributes and the hierarchies on categorical attributes, the *problem of optimal utility-based anonymization* is to compute a k-anonymous table T' such that the weighted normalized certainty penalty on T' is minimized.

The previous studies show that the problem of optimal k-anonymity is NP-hard under various quality models. The utility-based model we propose here is a generalization of the suppression model. Thus, the optimal utility-based anonymization is also NP-hard.

4. GREEDY METHODS

We propose two greedy algorithms. The first method conducts a bottom-up search, while the second one works top-down.

4.1 The Bottom-Up Method

To maximize the utility of the anonymization of a tuple, we may "cluster" the tuples locally according to the weighted certainty penalty. Those compact clusters having at least k tuples can be gen-

Input: a table T, parameter k, weights of attributes, and hierarchies on categorical attributes;
Output: a k-anonymous table T';
Method:
1: Initialization: create a group for each tuple;
2: `WHILE` there exists some group G such that $|G| < k$ `DO` {
3: `FOR` each group G such that $|G| < k$ `DO` {
4: scan all other groups once to find group G' such that $NCP(G \cup G')$ is minimized;
5: merge groups G and G';
}
6: `FOR` each group G such that $|G| \geq 2k$ `DO`
7: split the group into $\lfloor \frac{|G|}{k} \rfloor$ groups such that each group has at least k tuples;
}
8: generalize and output the surviving groups;

Figure 3: The bottom-up algorithm.

Input: a table T, parameter k, weights of attributes, hierarchies on categorical attributes;
Output: a k-anonymous table T';
Method:
1: `IF` $|T| \leq k$ `THEN RETURN`;
2: `ELSE` {
3: partition T into two exclusive subsets T_1 and T_2 such that T_1 and T_2 are more local than T, and either T_1 or T_2 have at least k tuples;
4: `IF` $|T_1| > k$ `THEN` recursively partition T_1;
5: `IF` $|T_2| > k$ `THEN` recursively partition T_2;
}
6: adjust the groups so that each group has at least k tuples;

Figure 4: The top-down greedy search method.

eralized. This idea leads to our bottom-up method.

At the beginning, we treat each tuple as an individual group. In each iteration, for each group whose population is less than k, we merge the group with the other group such that the combined group has the smallest weighted certainty penalty. The iteration goes on until every group has at least k tuples. The algorithm is shown in Figure 3.

The bottom-up algorithm is a greedy method. In each round, it merges groups such that the resulted weighted certainty penalty is locally minimized. In one iteration, if one group is merged with multiple groups, it is possible that the group becomes larger than k. In order to avoid over-generalization, if a group has more than $2k$ tuples, then the group should be split. It is guaranteed that in the resulted table, each group has up to $(2k-1)$ tuples.

Please note that, unlike many previous methods that try to minimize the average number of tuples per group, our algorithms try to reduce the weighted certainty penalty, which reflects the utility of the anonymized data. At the same time, they also keep the number of tuples per group small.

After the k-th round, the number of tuples in a group is at least 2^k. Therefore, by at most $\lceil \log_2 k \rceil$ iterations, each group has at least k tuples, and thus the generalized groups satisfy the k-anonymity requirement. The complexity of the algorithm is $O(\lceil \log_2 k \rceil |T|^2)$ on table T.

The bottom-up method is a local recoding method. It does not split the domain. Instead, it only searches the tuples. Different groups may have overlapping ranges. Moreover, in the step of splitting, several tuples with the identical quasi-identifier may be split into different groups.

4.2 A Top-Down Approach

The major cost in the bottom-up method is to search for the closest groups (Step 4 in Figure 3). In the bottom-up method, we have to use a two-level loop to conduct the search. We observe, if we can partition the data properly so that the tuples in each partition are local, then the search of the nearest neighbors can be sped up. Motivated by this observation, we develop the top-down approach.

The general idea is as follows. We partition the table iteratively. A set of tuples is partitioned into subsets if each subset is more local. That is, likely they can be further partitioned into smaller groups that reduce the weighted certainty penalty. After the partitioning, we merge the groups that are smaller than k to honor the k-anonymity requirement.

The algorithm framework is shown in Figure 4. To keep the algorithm simple, we consider binary partitioning. That is, in each round, we partition a set of tuples into two subsets. We adopt the following heuristic. We form two groups using the two seed tuples that cause the highest certainty penalty if they are put into the same group, and assign the other tuples into the two groups according to the two seed tuples.

Technically, we want to find tuples $u, v \in T$ that maximize $NCP(u, v)$. u and v become the seed tuple of groups G_u and G_v, respectively.

The cost of finding u, v such that $NCP(u, v)$ is maximized is $O(|T|^2)$. To reduce the cost, we propose a heuristic method here. We randomly picks a tuple u_1. By scanning all tuples once, we can find tuple v_1 that maximizes $NCP(u_1, v_1)$. Then, we scan all tuples again, find tuple u_2 that maximizes $NCP(u_2, v_1)$. The iteration goes on a few rounds until $NCP(u, v)$ does not increase substantially. Our experimental results on both the real data sets and the synthetic data sets show that the maximal weighted certainty penalty converges quickly. By up to 3 rounds, we can achieve 97% of the maximal penalty. By up to 6 rounds, we can achieve more than 98.75% of the maximal penalty. In practice, we can choose a small integer as the number of rounds to find the seed tuples.

Once the two seed tuples are determined, two groups G_u and G_v are created. Then, we assign other tuples to the two groups one by one in a random order. For tuple w, the assignment depends on $NCP(G_u, w)$ and $NCP(G_v, w)$, where G_u, G_v are the groups formed so far. Tuple w is assigned to the group that leads to a lower uncertainty penalty.

If a resulting group has k or more tuples, then the partitioning is conducted recursively on the group. A postprocessing step adjusts for those groups with less than k tuples. If one group G has less than k tuples, we apply the local greedy adjustment similar to the bottom-up approach. That is, we consider two alternatives. First, we can find a set G' of $(k-|G|)$ tuples in some other group that has more than $(2k-|G|)$ tuples such that $NCP(G \cup G')$ is minimized. Second, we compute the increase of penalty by merging G with the nearest neighbor group of G. By comparing the two penalty measures, we decide whether G' is moved to G or G is combined with its nearest neighbor group. Such adjustments should be done until every group has at least k tuples.

In the worst case, the overall partitioning cost is $O(|T|^2)$, and we may have to adjust $\lfloor \frac{|T|}{2k} \rfloor$ groups each having less than k tuples. However, in practice, the number of groups that are smaller than k is much less than the worst case. As shown in our experiments, the top-down method is clearly faster than the bottom-up method.

The top-down method is also a local recoding method. Two tuples identical in the quasi-identifier may be assigned to two different groups.

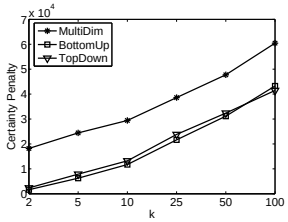

Figure 6: Certainty penalty on data set Adults.

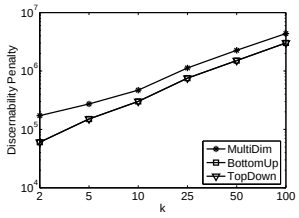

Figure 7: Discernability penalty on data set Adults.

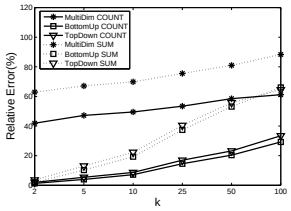

Figure 8: Query answering error rate on data set Adults.

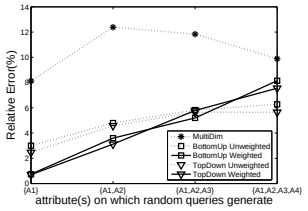

Figure 9: Utility in query answering, on uniform synthetic data sets (dimensionality=4, $k = 10$).

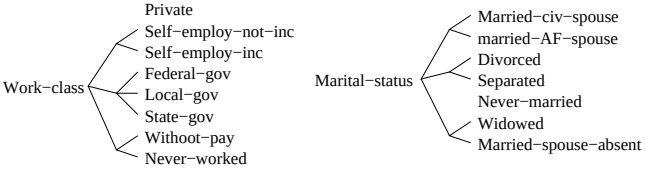

Figure 5: The hierarchies on attributes work-class and marital-status.

5. EXPERIMENTAL RESULTS

We compare three methods: the mondarian multidimensional k-anonymization method [7], the bottom-up method and the top-down method developed in this paper. According to [7], the mondarian multidimensional k-anonymization method (called MultiDim for short hereafter) is so far the best method in both quality (measured by the discernability penalty) and efficiency.

We measure the quality of the anonymization using three criteria: the certainty penalty, the discernability penalty, and the error rate in query answering. The certainty penalty proposed in this paper measures the utility of the anonymization. The discernability penalty is a de facto standard measure on anonymization quality used in many previous studies. The error rate measures how effective the anonymized data sets are in query answering.

All our experiments were conducted on a PC with a Pentium P4 2.0 GHz CPU and 512 MB main memory, running Microsoft Windows XP. All the algorithms were implemented by us in Microsoft Visual C++ version 6.0.

5.1 Results on Real Data Set Adults

The Adults census data set from the UC Irvine machine learning repository has become a de facto benchmark for k-anonymization. The data set was configured as described in [4]. The salary class attribute was dropped, and the tuples with missing values were removed. The resulting data set contains $30,162$ tuples.

Since the MultiDim method does not handle hierarchies on categorical attributes but treats a categorical attribute as a discrete numeric attribute, we configured the data set for MultiDim as it was used in [7]. For the bottom-up method and the top-down method proposed in this paper, we used age and education levels as numeric data, and use the other attributes as categorical attributes. We used the two hierarchies in Figure 5 on attributes work-class and marital-status. On other categorical attributes, a simple two-level hierarchy is applied: the values are the leaf nodes and the root is ALL (i.e., suppression). All weights were set to 1.

Figure 6 shows the certainty penalty of the anonymization of the three methods with respect to different k values. As expected, since the bottom-up method and the top-down method focus on the certainty penalty, but the MultiDim method does not, the anonymization generated by the bottom-up method and the top-down method has a clearly lower certainty penalty. The gap is stable, about 2×10^4.

Figure 7 compares the discernability penalty (drawn in the logarithmic scale) of the anonymization generated by the three methods with respect to different values of k. Interestingly, although the bottom-up and the top-down methods do not explicitly focus on reducing the discernability penalty, they outperform the MultiDim method. The results show that optimizing the utility and the reducing the discernability are not conflicting with each other. In fact, the two methods also try to keep the size of groups small when they reduce the certainty penalty. Grouping tuples locally can bring us benefit on reducing both the certainty penalty and the discernability penalty.

To test the effectiveness of query answering using the anonymized data, we generate workloads using `SUM` and `COUNT` aggregate queries, respectively. Each workload has $1,000$ random queries. Each `COUNT` query involves all the attributes, and each `SUM` query involves all but the age attribute that is used to compute the sum. The ranges of the attributes are selected randomly. For a categorical attribute, a query carries either a random categorical value, or a set of values that are summarized by an internal node in the hierarchy as the range. This is consistent with the settings in [7].

Figure 8 shows the results on two workloads of aggregate functions `COUNT` and `SUM`, respectively, with respect to different k values. The bottom-up method and the top-down method outperform the MultiDim method substantially, which can be explained in two aspects. First, the utility-driven anonymization put tuples that are similar to each other into groups. The generalized groups often have small ranges, and can answer queries more accurately. Second, our methods handle categorical attributes better than the MultiDim method. The hierarchies are considered in the anonymization. This contributes to the query answering quality strongly.

The runtime of the three methods on the Adult data set is not sensitive to k, about 10 seconds (MultiDim), 60 seconds (the top-down method) and 200 seconds (the bottom-up method). The top-down method is about 5-6 times slower than MultiDim, and is much faster than the bottom-up method. The difference in the efficiency can be explained by their complexity. While the MultiDim method has the complexity $O(|T| \log |T|)$, the bottom-up and the top-down methods have complexity $O(|T|^2)$.

5.2 Results on Synthetic Data Sets

To test the performance of the three methods more thoroughly, we generated synthetic data sets in two types of distributions: uniform distribution and Gaussian distribution. The dimensionality and the number of tuples may vary according to the needs of experiments. By default, a data set has $10,000$ tuples and each attribute

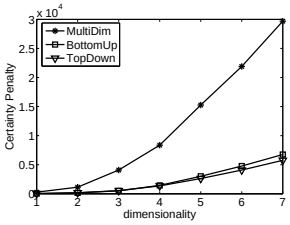

Figure 10: Certainty penalty with respect to dimensionality, on synthetic data sets with Gaussian distribution ($\sigma = 1.0$, $k = 10$).

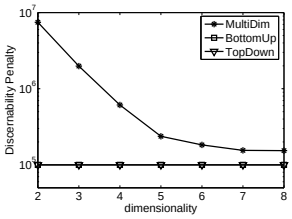

Figure 11: Discernability penalty with respect to dimensionality, on synthetic data sets with Gaussian distribution ($\sigma = 1.0$, $k = 10$).

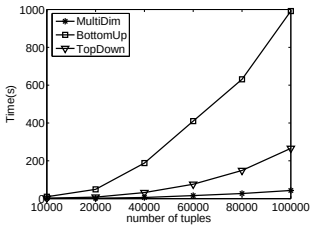

Figure 12: Scalability with respect to database size, on synthetic data sets with uniform distribution (dimensionality=4, $k = 10$).

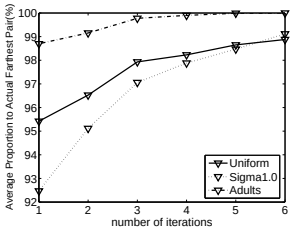

Figure 13: The effectiveness of the seed tuple choice heuristic in the top-down method, on real data set Adults, and synthetic data sets with uniform and Gaussian distribution (dimensionality=4).

is in the domain of integer with range $[1, 16]$. Again, by default the weights are set to 1.

We generate 4 groups of random queries on attribute combinations A_1, A_1A_2, $A_1A_2A_3$, and $A_1A_2A_3A_4$, respectively. The average error rates of the queries in each group is shown in Figure 9. For comparison, we also conduct the same queries on anonymization that do not consider the weights. As can be seen, the effect of utility-based anonymization is significant. The anonymization using the weighted top-down or bottom-up methods answers the queries on A_1, A_1A_2, and $A_1A_2A_3$ more accurately than the non-weighted methods. When all attributes are involved in a query, the weighted methods may lose some accuracy as the trade-off.

Figure 10 shows the certainty penalty with respect to various dimensionality. The top-down method and the bottom-up method are comparable, and the top-down method is slightly better. The MultiDim method has a high certainty penalty in high dimensional data. Please note that, as the dimensionality increases, the certainty penalty generally increases accordingly since each attribute contributes to the certainty penalty. The bottom-up and the top-down methods try to reduce the penalty in the anonymization procedure and thus may achieve good results. Figure 11 shows the results using the discernability penalty measure, which are consistent with the results reported in [7]. We can observe that the bottom-up method and the top-down method have similar performance, and achieve less discernability penalty than the MultiDim method. This is consistent with the results on the real Adults data set.

From this set of experiments, we conclude that the bottom-up and the top-down methods often have similar performance in anonymization quality, measured by both the certainty penalty and the discernability. The anonymization quality using those two methods are often better than the MultiDim method.

The advantages of the bottom-up and the top-down methods in anonymization quality do not come for free. The trade-off is the longer computation time. Figure 12 shows the results on scalability. The complexity of the MultiDim method is $O(|T| \log |T|)$, lower than that of the bottom-up and the top-down methods. Thus, the MultiDim method is more scalable. However, since anonymization is typically an offline, one-time task, quality can be a more important concern than the runtime. On the other hand, the difference between the top-down method and the MultiDim method is not dramatic. In our experiments, even when the data set scales up to $100,000$ tuples, the runtime of the top-down approach is just less than 6 times slower than that of the MultiDim method.

The top-down method is substantially faster than the bottom-up method. As analyzed in Section 4, splitting in the top-down method is much faster than merging in the bottom-up method.

A critical step in the top-down method is to choose two seed tuples. We used a heuristic method as described in Section 4. Figure 13 shows the effectiveness of the heuristic. We used a thorough method to compute the pair of tuples of the largest certainty penalty. Then, we used the heuristic method to compute seed tuples that are far away, and compare their certainty penalty with the maximum. As shown, with a small number of iterations, our heuristic gives very good approximation to the maximum. Thus, in our implementation, we conduct 3 iterations to obtain the seed tuples.

To summarize, the extensive experiments using both real data sets and synthetic data sets show that, in terms of utility and discernability, the bottom-up method and the top-down method developed in this paper often achieve better anonymization in quality than the MultiDim method, the state-of-the-art approach. The top-down method is better than the bottom-up method.

The trad-off of high anonymization quality is the runtime. The MultiDim method is more efficient. However, the runtime of the top-down method is not far away from that of the MultiDim method in practice. Moreover, for anonymization, the computation time is often a secondary consideration yielding to the quality.

6. REFERENCES

[1] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB '05*.

[2] G. Aggarwal, et al. Anonymizing tables. In *ICDT'05*.

[3] G. Aggarwal, et al. Approximation algorithms for k-anonymity. *Journal of Privacy Technology*, (2005112001), 2005.

[4] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE'05*.

[5] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *KDD'02*.

[6] K. LeFevre, et al. Incognito: Efficient full-domain k-anonymity. In *SIGMOD'05*.

[7] K. LeFevre, et al. Mondrian multidimensional k-anonymity. In *ICDE'06*.

[8] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *PODS'04*.

[9] P. Samarati. Protecting respondents' identities in microdata release. *IEEE TKDE*, 13(6):1010–1027, 2001.

[10] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *PODS'98*.

[11] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. In *Technical Report SRI-CSL-98-04*, 1998.

[12] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness, and Knowledge-based Systems*, 10(5):571–588, 2002.

[13] L. Sweeney. K-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness, and Knowledge-based Systems*, 10(5):571–588, 2002.

[14] L. Willenborg and T. deWaal. *Elements of Statistical Disclosure Control*. Lecture Notes in Statistics. Springer Verlag, 2000.

[15] W. E. Winkler. Using simulated annealing for k-anonymity. In *Technical Report Statistics 2002-7, U.S. Census Bureau, Statistical Research Division*, 2002.