(α, k)-anonymity Based Privacy Preservation by Lossy join

Raymond Chi-Wing Wong[1], Yubao Liu[2], Jian Yin[2], Zhilan Huang[2], Ada Wai-Chee Fu[1], and Jian Pei[3]

[1] Department of Computer Science and Engineering, the Chinese University of Hong Kong, Hong Kong, cwwong,adafu@cse.cuhk.edu.hk

[2] Department of Computer Science, Zhongshan University, China, liuyubao,issjyin@mail.sysu.edu.cn, santahzl@gmail.com,

[3] School of Computing Science, Simon Fraser University, Canada, jpei@cs.sfu.ca

Abstract. Privacy-preserving data publication for data mining is to protect sensitive information of individuals in published data while the distortion to the data is minimized. Recently, it is shown that (α, k)-anonymity is a feasible technique when we are given some sensitive attribute(s) and quasi-identifier attributes. In previous work, generalization of the given data table has been used for the anonymization. In this paper, we show that we can project the data onto two tables for publishing in such a way that the privacy protection for (α, k)-anonymity can be achieved with less distortion. In the two tables, one table contains the undisturbed non-sensitive values and the other table contains the undisturbed sensitive values. Privacy preservation is guaranteed by the lossy join property of the two tables. We show by experiments that the results are better than previous approaches.

1 Introduction

Privacy-preserving data mining is about preserving the individual privacy and retaining as much as possible the information in a dataset to be released for mining. The perturbation approach [2] and the k-anonymity model [14, 13, 4, 1] are two major techniques for this goal.

The k-anonymity model assumes a **quasi-identifier** (QID), which is a set of attributes that may serve as an identifier in the data set. In the simplest case, it is assumed that the dataset is a table and that each tuple corresponds to an individual. For example, in Table 1, attributes Job, Birth and Postcode form a quasi-identifier, where attribute Illness is a sensitive attribute. The privacy may be violated if some quasi-identifier values are unique in the released table. The assumption is that an attacker can have the knowledge of another table where the quasi-identifier values are linked with the identities of individuals. Therefore, a join of the released table with this background table will disclose the sensitive data of individuals. A real example is found in the voter registration records in the United States, where the attributes of name, gender, zip code and date of

Job	Birth	Postcode	Illness
clerk	1975	4350	HIV
manger	1955	4350	flu
clerk	1955	5432	flu
factory worker	1955	5432	fever
factory worker	1975	4350	flu
technical supporter	1940	4350	fever

Table 1. Raw medical data set

Job	Birth	Postcode	Illness
*	*	4350	HIV
*	*	4350	flu
*	*	5432	flu
*	*	5432	fever
*	*	4350	flu
*	*	4350	fever

Table 2. A $(0.5, 2)$-anonymous table of Table 1 by full-domain generalization

Job	Birth	Postcode	Illness
white-collar	*	4350	HIV
white-collar	*	4350	flu
*	1955	5432	flu
*	1955	5432	fever
blue-collar	*	4350	flu
blue-collar	*	4350	fever

Table 3. An (0.5,2)-anonymous data set of Table 1 by local recoding

birth are recorded. It is found that a high percentage of the population can be uniquely identified by the gender, date of birth and the zip code [12].

Let Q be the quasi-identifier (QID). An **equivalence class** set, called a QID-EC, for the same QID value of a table with respect to Q is a collection of all tuples in the table containing identical values of Q. For instance, Table 2 contains two QID-EC's. The first QID-EC contains the first two and the last two tuples because these tuples contain identical values of Q. Similarly, the second QID-EC contains the third and the fourth record.

A data set D is *k-anonymous* with respect to Q if the size of every QID-EC with respect to Q is k or more. As a result, it is less likely that any tuple in the released table can be linked to an individual and thus personal privacy is preserved. For example, each QID-EC in Table 2 has a size equal to or greater than 2. If $k = 2$, the data set in Table 2 is said to be k-anonymous.

We assume that each attribute follows a generalization hierarchy. In this hierarchy, a value in a lower level has a more specific meaning compared with a value in a higher level. For instance, Figure 1 shows a generalization hierarchy of attribute job.

In order to achieve k-anonymity, we *generalize* some values in some attributes in the quasi-identifier by replacing the values in a lower level by the values in a higher level according to the generalization hierarchy. Table 2 is a generalization of Table 1.

2 (α, k)-anonymity

The k-anonymity model is proposed in order to prevent the re-identification of individuals in the released data set. However, it does not consider the *inference*

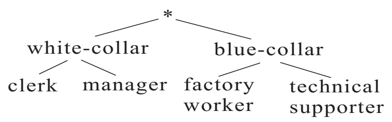


Fig. 1. Generalization hierarchy of attribute job

relationship from the quasi-identifier to some sensitive attribute. We assume for simplicity that there is only one sensitive attribute and that some values of this attribute are sensitive values. Suppose all tuples in a QID-EC contain the same sensitive value in the released data set, even though the size of the QID-EC is greater than or equal to k, all tuples in this QID-EC are linked to this sensitive value in the released data set. Therefore, each individual that has the corresponding QID value will be linked to the sensitive value. Let us call such an attack an inference attack.

In order to overcome this attack, [9, 17] proposed some privacy models and methods. [9] and [17] proposed an l-diversity model and an (α, k)-anonymity model, respectively, where α is a real number $\in [0, 1]$ and k and l are positive integers. As discussed in [17], it is difficult for users to set the parameters in the l-diversity model. In this paper, we focus on the (α, k)-anonymity model, which generates publishable data that is free from the inference attack. In addition to k-anonymity, this model requires that the value of the frequency (in fraction) of any sensitive value in any QID-EC is no more than α after anonymization.

There are two possible schemes of generalizations: *global recoding* and *local recoding*. With **global recoding** [13, 8, 3, 11, 7, 16, 4] all values of an attribute come from the same domain level in the hierarchy. In other words, all values come from the values in the same level in the generalization hierarchy. For example, all values in attribute job are in the lowest level (i.e. clerk, manager, factory worker and technical supporter), or all are in the top level (i.e. *). For example, a global recoding of Table 1 is Table 2. One advantage is that an anonymous view has uniform domains. But, it may lose more information, compared with local recoding (which will be discussed next), because it suffers from *over-generalization*.

Under the scheme of **local recoding** [14, 13, 1, 10, 6, 5, 19], values may be generalized to different levels in the domain. For example, Table 3 is a (0.5, 2)-anonymous table by local recoding. In fact, one can say that local recoding is a more general model and global recoding is a special case of local recoding. Note that, in the example, known values are replaced by unknown values (*). This is called *suppression*, which is one special case of generalization, which is in turn one of the ways of recoding.

It is easy to check that generalizing data to form QID-EC's in a released table is one possible way to achieve (α, k)-anonymity. However, it is not the only possible way, and we shall describe another method in the next section.

3 The lossy join approach

In recent work, it has been found that lossy join of multiple tables is useful in privacy-preserving data publishing [18, 15]. The idea is that if two tables with

Job	Birth	Postcode	Illness	ClassID
clerk	1975	4350	HIV	1
manager	1955	4350	flu	1
clerk	1955	5432	flu	2
factory worker	1955	5432	fever	2
factory worker	1975	4350	flu	3
technical supporter	1940	4350	fever	3

Table 4. Temp table

Job	Birth	Postcode	ClassID
clerk	1975	4350	1
manager	1955	4350	1
clerk	1955	5432	2
factory worker	1955	5432	2
factory worker	1975	4350	3
technical supporter	1940	4350	3

Table 5. NSS Table

ClassID	Illness
1	HIV
1	flu
2	flu
2	fever
3	flu
3	fever

Table 6. SS Table

a join attribute are published, the join of the two tables can be lossy and this lossy join helps to conceal the private information.

In this paper, we make use of the idea of lossy join to derive a new mechanism for achieving a similar privacy preservation target as (α, k)-anonymization. Let us take a look at an example in Table 1. A (0.5, 2)-anonymization is given in Table 3. From this table, we can generate a table Temp as shown in Table 4. For each equivalence class E in the anonymized table, we assign a unique identifier (ID) to E and also to all tuples in E. Then, we attach the correspondence ID to each tuple in the original raw table and form a new table Temp. From the Temp table, we can generate two separate tables, Tables 5 and 6. The two tables share the attribute of ClassID. If we join these two tables by the ClassID, it is easy to see that the join is lossy and it is not possible to derive the table Temp after the join. The result of joining the two tables is given in Table 7. From the lossy join, each individual is linked to at least 2 values in the sensitive attribute. Therefore, the required privacy of individual can be guaranteed. Also, in the joined table, for each individual, there are at least 2 individuals that are linked to the same bag B of sensitive values, such that in terms of the sensitive values, they are not distinguishable. For example, the first record in the raw table (QID = (clerk, 1975, 4350)) is linked to bag {HIV,flu}. We find that the second individual (QID = (manager, 1955, 4350)) is also linked to the same bag B of sensitive values. This is the goal of k-anonymity for the protection of sensitive values.

3.1 Contribution

[17] proposed to generate *one generalized* table which satisfies (α, k)-anonymity. Since the table is generalized, the data in the table is distorted. In this paper, we generalize the definition of (α, k)-anonymity to allow for the generation of *two* tables instead of *one generalized* table. In this way, the privacy protection for (α, k)-anonymity can be achieved with less distortion. In the two tables, one

Job	Birth	Postcode	Illness	ClassID
clerk	1975	4350	HIV	1
manager	1955	4350	HIV	1
clerk	1975	4350	flu	1
manager	1955	4350	flu	1
clerk	1955	5432	flu	2
factory worker	1955	5432	flu	2
clerk	1955	5432	fever	2
factory worker	1955	5432	fever	2
factory worker	1975	4350	flu	3
technical supporter	1940	4350	flu	3
factory worker	1975	4350	fever	3
technical supporter	1940	4350	fever	3

Table 7. Join result table

table contains the *undisturbed* non-sensitive values and the other table contains the *undisturbed* sensitive values. The privacy preservation is by the *lossy join* property of the two tables. We show that the results are better than previous approaches [17, 18] in the experiments.

The rest of the paper is organized as follows. In Section 4, we re-visit (α, k)-anonymity and propose a genearlization model of (α, k)-anonymity. In Section 5, we describe how the lossy join can be adapted to the generalized (α, k)-anonymity model. We propose an algorithm which generates two tables satisfying (α, k)-anonymity in Section 6. A systematic performance study is reported in Section 7. The paper is concluded in Section 8.

4 Generalized (α, k)-anonymity

Let us re-examine the objectives of (α, k)-anonymity. With k-anonymity, we want to make sure that when an individual is mapped to some sensitive values, at least $k-1$ other individuals are also mapped to the same sensitive values. Let B be a bag of these sensitive values. For example, consider an individual with QID=(clerk, 1975, 4350) in Table 1. With 2-anonymity, since s/he is mapped to the first and the second tuple in Table 3, s/he is mapped to a bag $B = \{HIV, flu\}$. There is another individual with QID=(manager, 1955, 4350) in Table 1 that also is mapped to the same bag $B = \{HIV, flu\}$ in Table 3. (α, k)-anonymity further ensures that no sensitive value is sufficiently dominating in B so that an individual cannot be linked to any sensitive value in B with a high confidence. For instance, with $\alpha = 0.5$, since B contains HIV and flu, the frequency (in fraction) of each value in B is at most 0.5. Based on this observation, we generalize the definition of (α, k)-anonymity as follows:

Definition 1 (Generalized (α, k)-anonymity). *Consider a dataset D in which a set of attributes form the QID. We assume that the adversary only has the knowledge of an external table where the QID's are linked to individuals. A released data set D' generated from D satisfies generalized k-anonymity if, whenever an individual is linked to a bag B of sensitive values, at least $k-1$ other* *individuals are also linked to B. In addition, if the frequency (in fraction) of any sensitive value in B is no more than α, then the released data satisfies generalized (α, k)-anonymity.*

5 Generalized (α, k)-anonymity by Lossy Join

Suppose we form an anonymized table in which some QID values are generalized. In the anonymized table, each set of tuples with the same QID values forms a QID-EC. However, instead of publishing one single table A with the generalized values, there is the possibility of separating the sensitive attribute from the non-sensitive attributes and generate two tables by projecting these two sets of attributes. Tuples in the two tables are linked if they belong to the same QID-EC in A. Hence we can publish two tables: (1) one table, called *non-sensitive table (NSS table)*, containing all the non-sensitive attributes together with QID equivalence class (QID-EC in A) IDs, and (2) the other table, called *sensitive table (SS table)*, containing the QID-EC ID and the sensitive attributes. The released tables are annotated with the remark that each tuple in each of the two published table corresponds to one record in the original single table. This is to ensure that a user will not mistakenly join the two tables and assume that the join result corresponds to the original table.

The schema of the non-sensitive table (NSS table) is shown as follows, where Class ID corresponds to QID-EC ID.

Original QID attributes	Class ID

The schema of the sensitive table (SS table) is shown as follows.

Class ID	Sensitive attribute

Let us consider the example in Table 1 again. We propose that Table 5 (NSS) and Table 6 (SS) can be published as the anonymized data.

Theorem 1. *The resulting published tables NSS and SS satisfy generalized (α, k)-anonymity.*

Proof: Given the QID information of individuals in a table T_I (which we assume that an attacker may possess) and the anonymized Table T_A (e.g. Table 3), we can "join" the two tables by matching each QID in T_A to its anonymized equivalence class and obtain a table T_{IA}. Since T_A satisfies (α, k)-anonymity, when the QID of an individual is linked to a bag B of values in the sensitive attribute, at least $k-1$ other QID's of other individuals are also linked to B. In addition, the frequency (in fraction) of any sensitive value in B is no more than α.

Now, suppose the adversary is given tables NSS and SS. Equipped with only table T_I, an adversary must join the tables NSS and SS on their common attribute in an attempt to link the QID's to the sensitive values. Let the join result be table T'_A, such as Table 7. Consider any QID-EC with class ID X. Let B_X be the bag of sensitive values that X is linked to in T_A and suppose there are a tuples in T_A belonging to X. In Table T'_A, there will be a^2 tuples generated for X. In Table T'_A, B_X becomes B'_X and each entry in B_X is duplicated a times in B'_X. In the a^2 tuples in T'_A, each original QID value in the given table T will now be linked to the bag B'_X. Besides, a individuals are involved in X, and each is linked to B'_X. The frequency of each sensitive value in B'_X is the same as that in B_X in T_{IA}. Hence, the tables NSS and SS release no more information as the table T_A in terms of the linkage of an individual to a bag B of sensitive values and in terms of the percentage of each sensitive value in B.

This shows that the privacy protection provided by the single anonymized table T_A is no stronger than that provided by the NSS and SS tables in terms of (α, k)-anonymity. Since T_A satisfies (α, k)-anonymity, tables NSS and SS also satisfy (α, k)-anonymity. ∎

The example shown in Tables 3 to 7 demonstrates the ideas in the proof above. If we publish Tables 5 and 6, we can achieve similar privacy preservation objectives as if we publish Table 3 only.

6 Algorithm

Our method includes the following steps.

1. Construct an (α, k)-anonymous table $T*$ from the given raw table (which will be described in Algorithm 1), and assign each equivalence class in the resulting table a class ID.
2. Add a column for the class ID of the equivalence class in the original raw table, such that, for each tuple, the class ID is the ID of the equivalence class that the tuple belongs in $T*$. Call this new table the Temp table. Hence the Temp table contains the raw table plus one extra column.
3. Project the Temp table on the QID attributes and the Class ID column. The resulting table is the NSS table.
4. Project the Temp table on the sensitive attributes and the Class ID column. This results in the SS table.

The top-down approach has been found to be highly effective in k-anonymization [4]. In this approach, the table is first totally anonymized to the unknown values, and then attributes are specialized one at a time until we hit a point where the resulting table violates (α, k)-anonymity. We shall adopt the top-down approach in [17] to tackle the first step of (α, k)-anonymization in the above. The idea of the algorithm is to first generalize all tuples *completely* so that, initially, all tuples are generalized to one equivalence class. Then, some values in the dataset are *specialized* in iterations. During the specialization, we must maintain (α, k)-anonymity. The process continues until we cannot specialize the tuples anymore without violating (α, k)-anonymity. The pseudo-code of the top-down approach is shown in Algorithm 1.

Algorithm 1 Top-Down Approach for Single Attribute

```
1: fully generalize all tuples such that all tuples are equal
2: let $P$ be a set containing all these generalized tuples
3: $S \leftarrow \{P\}$; $O \leftarrow \emptyset$
4: repeat
5:   $S' \leftarrow \emptyset$
6:   for all $P \in S$ do
7:     specialize all tuples in $P$ one level down in the generalization hierarchy such
       that a number of specialized child nodes are formed
8:     unspecialize the nodes which do not satisfy $(\alpha, k)$-anonymity by moving the
       tuples back to the parent node
9:     if the parent $P$ does not satisfy $(\alpha, k)$-anonymity then
10:      unspecialize some tuples in the remaining child nodes so that the parent $P$
         satisfies $(\alpha, k)$-anonymity
11:    for all non-empty branches $B$ of $P$, do $S' \leftarrow S' \cup \{B\}$
12:    $S \leftarrow S'$
13:    if $P$ is non-empty then $O \leftarrow O \cup \{P\}$
14: until $S = \emptyset$
15: return $O$
```

7 Experimental Results

The system platform we used is: Windows XP OS, Microsoft SQL Server 2000, Intel Celeron CPU 2.66GHz, 256MB Memory, 80G Hard disk. We implemented our proposed algorithm, the (α, k)-anonymity based privacy preservation by lossy join, in C/C++ language. Let us denote it by *Alpha*(*Lossy*). We compared the proposed lossy-join algorithm with two algorithms in the literature. One is the original algorithm of (α, k)-anonymity [17] which generalizes the QID and forms one generalized table only. Let us denote the algorithm by *Alpha*. The other algorithm is the anatomy algorithm which makes use of the lossy join for the anonymization [18]. Let us denote the algorithm by *Anatomy*. Anatomy also generates two tables with a similar strategy of separating the sensitive data and the QID data. However, the goal of Anatomy is to create QID-EC's which satisfy the l-diversity requirement, without precaution in creating QID-EC's that also minimizes the effective distortion to the QID values. In other words, Anatomy does not consider the minimization of the variations in the QID values in each QID-EC when two tables are released. *Alpha*(*Lossy*) takes care of this issue by the top-down anonymization algorithm and therefore results in less data distortion.

The source code of this algorithm can be obtained from the author's website http://www.cs.cityu.edu.hk/∼taoyf/paper/vldb06.html. In our experiments, we make some modifications on the ST files generated by the original anatomy algorithm such that ST table can be loaded into the Microsoft SQL Server 2000.

Similar to [4, 8, 17], we adopted the adult data set for the experiment, which can be downloaded in the UCIrvine Machine Learning Repository (http://www.ics.uci.edu/∼mlearn/MLRepository.html). We eliminated the records

	Attribute	Distinct Values	Generalizations	Height
1	Age	74	5-, 10-, 20-year ranges	4
2	Work Class	7	Taxonomy Tree	3
3	Education	16	Taxonomy Tree	4
4	Martial Status	7	Taxonomy Tree	3
5	Race	5	Taxonomy Tree	2
6	Sex	2	Suppression	1
7	Native Country	41	Taxonomy Tree	3
8	Salary Class	2	Suppression	1
9	Occupation	14	Taxonomy Tree	2

Table 8. Description of Adult Data Set

with unknown values in this data set. The resulting data set contains 45,222 tuples. Nine of the attributes were chosen in our experiments, as shown in Table 8. By default, we set $k = 2$ and $\alpha = 0.33$. In Table 8, we set the first eight attributes and the last attribute as the quasi-identifer and the sensitive attribute, respectively.

We compare the algorithms in terms of effectiveness for aggregate queries. Similar to [18], the effectiveness of aggregate query is defined to be its average relative error in answering a query of the following form.

SELECT COUNT(*)
FROM Unknown-Microdata
WHERE $pred(A_1^{qi})$ AND ... AND $pred(A_{qd}^{qi})$ AND $pred(A^s)$

In the above query, Unknown-Microdata is an original data set or an anonymized data set. qd denotes the number of QID attributes to be queried and A^s denotes the sensitive attribute. For any attribute A, the predicate $pred(A)$ has the form

$$(A = x_1 \text{ OR } A = x_2 \text{ OR ... OR } A = x_b)$$

where x_i is a random value in the domain of A, for $1 \leq i \leq b$. The value of b depends on the *expected query selectivity* s

$$b = \lceil |A| \cdot s^{1/(qd+1)} \rceil$$

where $|A|$ is the domain size of A. If the value of s is set higher, the selection conditions in $pred(A)$ will be more.

We compare the anonymized tables generated by different algorithms in terms of *average relative error*, which is defined as follows. We perform the aggregate query with the original data set, called *Original*. That is,

SELECT COUNT(*)
FROM Original
WHERE $pred(A_1^{qi})$ AND ... AND $pred(A_{qd}^{qi})$ AND $pred(A^s)$

Let us call the count obtained above *act*. We execute the aggregate query with the anonymized data set as follows. As algorithm *Alpha(Lossy)* and algorithm *Anatomy* generates two tables, namely NSS and SS, we perform the query as follows.

```
SELECT COUNT(*)
FROM SS
WHERE SS.ClassID in (SELECT NSS.ClassID FROM NSS
                WHERE $pred(A_1^{qi})$ AND ... AND $pred(A_{qd}^{qi})$) AND $pred(A^s)$)
```

Let us call the count obtained above *est*. As algorithm *Alpha* generates one anonymized table, we perform the first query by replacing `Unknown-Microdata` with the anonymized or generalized data. Then, we define the relative error to be $|act - est|/act$, where *act* is its actual count derived from the original, and *est* the estimated count computed from the anonymized table. In our experiments, we compare all algorithms by varying the following factors: (1) the number of QID-attributes d; (2) query dimensionality qd; (3) selectivity s and (4) dataset cardinality n.

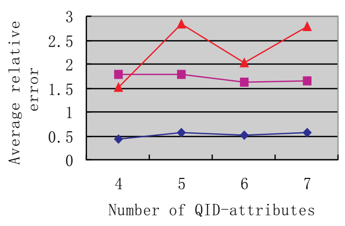
For each setting, we performed 1000 queries on the anonymized tables and then reported the average query accuracy. By default, we set $qd = 4, s = 0.05$ and $n = 45222$. As we adopt the first eight attributes in Table 8 as the quasi-identifier, the default value of d is 8.

We study the effect of the number of QI-attributes as shown in Figure 2. The average relative error remains unchanged. Also, algorithm *Alpha(Lossy)* gives a lower average relative error compared with algorithm *Anatomy* and algorithm *Alpha*. This is because algorithm *Alpha(Lossy)* considers the minimization step of the distortion for the anonymization but algorithm *Anatomy* does not. Also, algorithm *Alpha(Lossy)* does not generalize the table but algorithm *Alpha* generalize the table, which makes the average relative error higher. On average, algorithm *Anatomy* gives lower average relative error compared with algorithm *Alpha*. The reason is similar. Algorithm *Alpha* generalizes the table, which distort the data much, but algorithm *Anatomy* does not.

We also studied the effect of query dimensionality qd as shown in Figure 3. Similarly, even though the average relative error of algorithm *Alpha(Lossy)* is smaller than that of algorithm *Anatomy* and algorithm *Alpha*, qd had little effect on the average relative error.

We also varied the selectivity s as shown in Figure 4 and found that the average relative error of all algorithms decreases when s increases. This is because, when s is larger, each attribute in the aggregate query involves more value matches. That means the actual count is larger. Note that the actual count is the denominator of the average relative error. Besides, if the generalized values in the anonymized table match more aggregate values in the query, the estimated count will be more accurate. Thus, the overall average relative error decreases when s increases.

Figure 5 shows the average relative error against the data set cardinality n. We found that the average relative error of all algorithms decreases slightly when n increases. This is because, when n is larger, there is more chance that a tuple can be matched with an existing tuple in the data without much generalization.



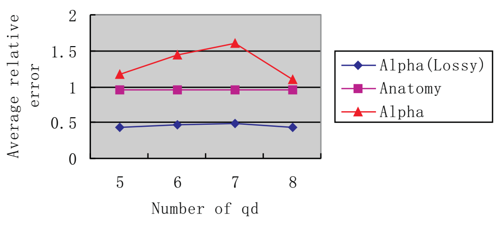
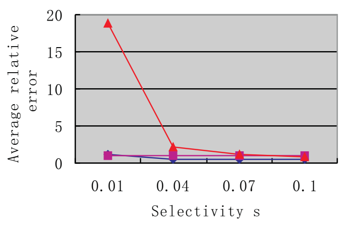


Fig. 2. Query accuracy vs. the number of QID-attributes d

Fig. 3. Query accuracy vs. query dimensionality qd



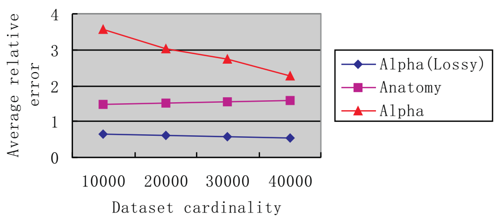


Fig. 4. Query accuracy vs. selectivity s

Fig. 5. Query accuracy vs. dataset cardinality

Similarly, algorithm *Alpha(Lossy)* gives a lower average relative error compared with algorithm *Anatomy* and algorithm *Alpha*.

8 Conclusion

In this paper, we proposed an (α, k)-anonymity based privacy preservation mechanism that reduce information loss by the use of lossy join. Instead of one generalized table, we generate two tables with a sharing attribute called ClassID, which corresponds to a unique identifier of an "equivalence class". One table contains the detailed information of the quasi-identifier and ClassID, and the other table contains ClassID and the sensitive attribute. By avoiding the generalization of the quasi-identifier in the first table, we achieve less information loss. We conducted some experiments and verified the improvement on information loss.

Acknowledgements: This paper is in part supported by the National Natural Science Foundation of China (60573097), Natural Science Foundation of Guangdong Province (05200302), Research Foundation of Science and Technology Plan Project in Guangdong Province (2005B10101032), and Research Foundation of Disciplines Leading to Doctorate degree of Chinese Universities (20050558017). This research was also supported by the RGC Earmarked Research Grant of HKSAR CUHK 4120/05E.

References

1. G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Anonymizing tables. In *ICDT*, pages 246–258, 2005.
2. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *SIGMOD*, pages 439–450, May 2000.
3. R. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE*, pages 217–228, 2005.
4. B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *ICDE*, pages 205–216, 2005.
5. A. Hundepool. The argus software in the casc-project: Casc project international workshop. In *Privacy in Statistical Databases*, volume 3050 of *Lecture Notes in Computer Science*, pages 323–335, Barcelona, Spain, 2004. Springer.
6. A. Hundepool and L. Willenborg. μ-and τ- argus: software for statistical disclosure control. In *Third international seminar on statsitcal confidentiality*, Bled, 1996.
7. V. S. Iyengar. Transforming data to satisfy privacy constraints. In *SIGKDD*, pages 279–288, 2002.
8. K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *SIGMOD*, pages 49–60, 2005.
9. A. Machanavajjhala, J. Gehrke, and D. Kifer. l-diversity: privacy beyond k-anonymity. In *ICDE*, 2006.
10. A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *PODS*, pages 223–228, 2004.
11. P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.
12. L. Sweeney. Uniqueness of simple demographics in the u.s. population. *Technical Report, Carnegie Mellon University*, 2000.
13. L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International journal on uncertainty, Fuzziness and knowldege based systems*, 10(5):571 – 588, 2002.
14. L. Sweeney. k-anonymity: a model for protecting privacy. *International journal on uncertainty, Fuzziness and knowldege based systems*, 10(5):557 – 570, 2002.
15. K. Wang and B. Fung. Anonymizing sequential releases. In *SIGKDD*, 2006.
16. K. Wang, P. S. Yu, and S. Chakraborty. Bottom-up generalization: A data mining solution to privacy protection. In *ICDM*, pages 249–256, 2004.
17. R. Wong, J. Li, A. Fu, and K. Wang. (alpha, k)-anonymity: An enhanced k-anonymity model for privacy-preserving data publishing. In *SIGKDD*, 2006.
18. X. Xiao and Y. Tao. Anatomy: Simple and effective privacy preservation. In *VLDB*, 2006.
19. J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. W.-C. Fu. Utility-based anonymization using local recoding. In *SIGKDD*, 2006.