Achieving k-Anonymity by Clustering in Attribute Hierarchical Structures

Jiuyong Li[1], Raymond Chi-Wing Wong[2], Ada Wai-Chee Fu[2], and Jian Pei[3]

[1] Department of Mathematics and Computing, The University of Southern Queensland, Australia
[2] Department of Computer Science and Engineering, The Chinese University of Hong Kong
[3] School of Computing Science, Simon Fraser University, Canada
jiuyong@usq.edu.au, cwwong, adafu@cse.cuhk.edu.hk, jpei@cs.sfu.ca

Abstract. Individual privacy will be at risk if a published data set is not properly de-identified. k-anonymity is a major technique to de-identify a data set. A more general view of k-anonymity is clustering with a constraint of the minimum number of objects in every cluster. Most existing approaches to achieving k-anonymity by clustering are for numerical (or ordinal) attributes. In this paper, we study achieving k-anonymity by clustering in attribute hierarchical structures. We define generalisation distances between tuples to characterise distortions by generalisations and discuss the properties of the distances. We conclude that the generalisation distance is a metric distance. We propose an efficient clustering-based algorithm for k-anonymisation. We experimentally show that the proposed method is more scalable and causes significantly less distortions than an optimal global recoding k-anonymity method.

1 Introduction

A vast amount of operational data and information has been stored at different vendors and organizations. Most of the stored data is useful only when the data is shared and analysed with other related data. However, this kind of data normally contains some personal details and sensitive information. The data can only be allowed to be released when the private information is protected.

More and more powerful data mining tools require a large amount of data from various sources to produce promising results. On the other hand, these powerful data mining tools may be maliciously used to uncover personal-related sensitive information in data. Therefore, privacy preservation becomes a fundamental issue in data mining.

Cryptographic technique is a choice since it can hide data from unauthorised users. However, cryptographic methods may restrict data access and exchange too much. Furthermore, cryptographic privacy-preserving methods [15, 22, 24] usually tailor some specific data mining tasks, and therefore lose generality.

Random perturbation can provide certain privacy protection [5, 4, 18], but they are suitable for data of numerical attributes. When data contains categorical values, the methods are not quite effective.

Data generalisation is applicable to both categorical and numerical data, and k-anonymity provides a practical model for privacy protection [21, 20, 19]. Since the k-anonymity model is simple and practical, it has been extensively studied in recent years [14, 6, 23, 10, 13]. A more general view of k-anonymisation is clustering with a constraint of the minimum number of objects in every cluster [3]. A number of methods approach identity protection by clustering [4, 1]. However, these methods are applicable to numerical attributes only. A recent work [9] extends a clustering-based method [8] to ordinal attributes, but it does not deal with attributes in hierarchical structures. Other works [2, 17] dealing with categorical attributes do not consider attribute hierarchies. In this paper, we focus our effort on achieving k-anonymity in hierarchical attribute structures. We define some general metrics in attribute hierarchies for measuring the quality of k-anonymous tables, and map them to generalisation distances which can be minimised in the process of k-anonymisation. This greatly facilitates achieving k-anonymity by local recoding via clustering. To the best of our knowledge, this work is the first work to do such mapping. We also present an efficient algorithm for this purpose, and demonstrate that our method causes less distortions than an optimal k-anonymity algorithm.

2 Preliminary Definitions

The objective of k-anonymisation is to make every tuple of privacy-related attributes in a published table identical to at least (k - 1) other tuples. As a result, no privacy-related information can be easily inferred.

For example, young people with stress and obesity are potentially identifiable by their unique combinations of gender, age and postcode attributes in Table 1a.

To preserve their privacy, we may generalise their gender and postcode attribute values such that each tuple in attribute set $\{$Gender, Age, Postcode$\}$ has two occurrences. The view after the generalisation is listed in Table 1b.

Gender	Age	Pcode	Problem
male	middle	4350	stress
male	middle	4350	obesity
male	young	**4351**	stress
female	young	**4352**	obesity
female	old	4353	stress
female	old	4353	obesity

Gender	Age	Pcode	Problem
male	middle	4350	stress
male	middle	4350	obesity
*****	young	**435***	stress
*****	young	**435***	obesity
female	old	4353	stress
female	old	4353	obesity

Gender	Age	Pcode	Problem
*	middle	435*	stress
*	middle	435*	obesity
*	young	435*	stress
*	young	435*	obesity
*	old	435*	stress
*	old	435*	obesity

Table 1. (a) Left: a raw table. (b) Middle: a 2-anonymous view by local recoding. (c) Right: a 2-anonymity view by global recoding.

In this paper, we adopt a simplified postcode scheme, where its hierarchy $\{$4201, 420*, 42**, 4***, *$\}$ corresponds to $\{$suburb, city, region, state, unknown$\}$, respectively.

Definition 1 (Quasi-identifier Attribute Set). *A* quasi-identifier attribute set *is a set of attributes in a table that potentially reveal private information, possibly by joining with other tables.*

For example, attribute set $\{$Gender, Age, Postcode$\}$ in Table 1a is a quasi-identifier. Table 1a potentially reveals private information of patients (e.g. young patients with stress and obesity). If the table is joined with other tables, it may reveal more information of patients' disease history. Normally, a quasi-identifier attribute set is understood by domain experts.

Definition 2 (equivalence class). *An* equivalence class *of a table with respect to an attribute set is the set of all tuples in the table containing identical values for the attribute set.*

For example, tuples 1 and 2 in Table 1a form an equivalence class with respect to attribute set $\{$Gender, Age, Postcode$\}$. Their corresponding values are identical.

Definition 3 (k-anonymity Property). *A table is* k-anonymous *with respect to a quasi-identifier if the size of every equivalence class with respect to the attribute set is k or more.*

k-anonymity requires that every occurrence within an attribute set has the frequency at least k. For example, Table 1a does not satisfy 2-anonymity property since tuples $\{$male, young, 4351$\}$ and $\{$female, young, 4352$\}$ occur once.

Definition 4 (k-anonymisation). *A view of a table is said to be a* k-anonymisation *of the table if the view modifies the table such that the view satisfies the k-anonymity property with respect to the quasi-identifier.*

For example, Table 1b is a 2-anonymous view of Table 1a since the size of all equivalence classes with respect to the quasi-identifier is 2.

A table may have more than one k-anonymous views, but some are better than others. For example, we may have another 2-anonymous view of Table 1a as in Table 1c. Table 1c loses more details than Table 1b. Another objective for k-anonymisation is to minimise distortions. We will give a definition of distortion later. Initially, we consider it as the number of cells being modified.

There are two ways to achieve k-anonymity, namely *global recoding* and *local recoding*. Another name for global recoding is *domain generalisation*. The generalisation happens at the domain level. When an attribute value is generalised, every occurrence of the value is replaced by the new generalised value. Most working models are global recoding models, such as [20, 14, 6, 19, 12, 23, 10] [1]. A global recoding method may *over-generalise* a table. An example of global recoding is given in Table 1c.

A local-recoding method generalises attribute values at cell level. A generalised attribute value co-exists with the original value. A local recoding method does not over-generalise a table and hence may minimise the distortion of an anonymous view. Sweeney studied a local recoding model, but did not present a working local recoding algorithm [21, 20]. Sweeney's MinGen algorithm is impractical and DataFly is a global recoding algorithm. Gagan Aggarwal et al. [2] and Adam Meyerson et al. [17] analysed a simplified local recoding model that does not involve hierarchical attributes. Both papers conclude that optimal k-anonymisation is NP-hard. μ- and τ-Argus methods [11], are two working local recoding methods, but μ-Argus does not guarantee k anonymity as discovered in [20]. τ-Argus works efficiently only on limited number of attributes. More recent work of local recoding k-anonymisation was reported in [13] by LeFevre et al. The method deals with numerical values, and does not involve attribute domain hierarchies. An example of local recoding is given in Table 1b.

A global-recoding method causes too much distortions to a table. It is preferable to use a local-recoding method. However, optimal local-recoding is NP-hard [2, 17] [2]. Therefore, good heuristic methods are required to achieve k-anonymisation by local recoding.

The objectives of k-anonymisation by local recoding is listed as follows.

- to modify a table to satisfy the k-anonymity property, and
- to minimise the distortion of the view from its original table.

3 Measuring the Quality of k-anonymisation

In this section, we discuss metrics for measuring the quality of generalisation.

A general criterion should be the distortion of a table. A simple measurement of distortion is the *modification rate*. For a k-anonymous view V of table T, the *modification rate* is the fraction of cells being modified within the quasi-identifier attribute set. For example, the modification rate from Table 1a to Table 1b is 22.2% and the modification rate from Table 1a to Table 1c is 66.7%.

This criterion does not consider hierarchical structures. For example, the distortion caused by the generalisation of birth date from D/M/Y to M/Y is significantly different from the distortion caused by the generalisation of gender from M/F to *. The former still

[1] [13] also considers global-recoding. However, the definition is different from our work and most previous work. Suppose there are three dimensions (A, B, C). In their global-recoding model, for each possible value (a, b, c) (where $a \in A, b \in B$ and $c \in C$), all tuples with this value in the data set should be generalised to the same value. However, this formulation is actually a local-recoding in our work and most previous work.

[2] To the best of our knowledge, the global local-recoding K-anonymity problem defined in this paper has not been shown to be NP-hard in the literature. As the definition of the global recoding in [13] is different from ours, the result of the NP-hardness shown in [13] can be regarded for the local-recoding problem in our work.

keeps most information of Birth Date but the latter loses all information for Gender. The modification rate is too simple to reflect such differences.

We calculate distortions of two tables based on distortions of their corresponding tuple pairs. We first define a metric measuring the distance between different levels in an attribute hierarchy.

Definition 5 (Weighted Hierarchical Distance). *Let h be the height of a domain hierarchy, and let levels $1, 2, \ldots, h-1, h$ be the domain levels from the most general to most specific, respectively. Let the weight between domain level j and $j-1$ be predefined, denoted by $w_{j,j-1}$, where $2 \leq j \leq h$. When a cell is generalised from level p to level q, where $p > q$. The weighted hierarchical distance of this generalisation is defined as*

$$\mathrm{WHD}(p,q) = \frac{\sum_{j=q+1}^{p} w_{j,j-1}}{\sum_{j=2}^{h} w_{j,j-1}}$$

The right part of Figure 1a shows the numbering methods of hierarchical levels and the left part of Figure 1a shows weights between hierarchical levels. Level 1 is always the most general level of a hierarchy and contains one value.

We can define weight $w_{j,j-1}$ to enforce a priority in generalisation. In the following, we discuss two simple but typical schemes.

1. Uniform Weight: $\boldsymbol{w_{j,j-1} = 1}$, where $2 \leq j \leq h$

This is the simplest scheme where all weights are equal to 1. In this scheme, WHD is the number of steps a cell being generalised over all possible generalisation steps, e.g. $h-1$. For example, let birth date hierarchy be {D/M/Y, M/Y, Y, 10Y, C/Y/M/O, *}, where 10Y stands for 10-year interval and C/Y/M/O for child, young, middle age and old age. WHD from D/M/Y to Y is $\mathrm{WHD}(6,4) = (1+1)/5 = 0.4$. In gender hierarchy, {M/F, *}, WHD from M/F to * is $\mathrm{WHD}(2,1) = 1/1 = 1$. This means that the distortion caused by the generalisation of five cells from D/M/Y to Y is equivalent to the distortion caused by the generalisation of two cells from M/F to *.

As this scheme is quite simple, this does not capture that the generalisations at different levels yield different distortions. It is expected that the generalisation near to the root should distort the data more compared with the generalisation far from the root. We take the address for illustration. Suppose the address contains three components - street no, street name and postcode. For example, the address is “20, Smith Street, Pcode 4351”. Let us consider two generalisations - the generalisation G_1 from “20, Smith Street, Pcode 4351” to “Smith Street, Pcode 4351” and the generalisation G_2 from “Pcode 4351” to “Pcode 435*”. It is obvious that G_1 (i.e. the removal of the street no) corresponds to a smaller distortion while G_2 (i.e. the removal of the suburb) corresponds to a larger distortion, because the area coverage by the suburb, of course, is larger than the area coverage by a housing (denoted by the street no). This example motivates us to propose another scheme.

2. Height Weight: $\boldsymbol{w_{j,j-1} = 1/(j-1)^{\beta}}$ where $2 \leq j \leq h$ and β is a real number ≥ 1 provided by a user.

For a fixed β, the intuition of this scheme is that the generalisation near to the top should give greater distortion compared with the generalisation far from the top. Thus, we formulate the height weight scheme, where the weight near to the top is larger and the weight far from the top is smaller. For example, consider a hierarchy: {D/M/Y, M/Y, Y, 10Y, C/Y/M/O, *} for birth date. Let $\beta = 1$. WHD from D/M/Y to M/Y is $\mathrm{WHD}(6,5) = (1/5)/(1/5+1/4+1/3+1/2+1) = 0.087$. In gender hierarchy {M/F, *}, WHD from M/F to * is $\mathrm{WHD}(2,1) = 1/1 = 1$. The distortion caused by the generalisation of one cell from M/F to * in gender attribute is more than the distortion caused by the generalisation of 11 cells from D/M/Y to M/Y in birth date attribute.

In some cases, users prefer that the weight near to the leaf node should be equal to a smaller value (compared with the case when $\beta = 1$). Then, in this model, we allow this requirement. In order to satisfy this kind of requirement, we simply set the β value with a larger value (e.g. 2) such that the weight near to the leaf node is smaller.

There are other possible other schemes for various applications.

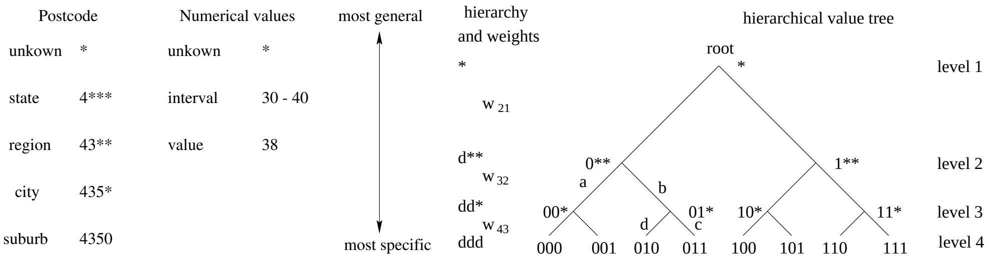


Fig. 1. (a) Left: Two examples of domain hierarchies - one for categorical values and one for numerical values. (b) Right: Depiction of weights between domain levels and a simplified hierarchical value tree.

In the following, we define distortions caused by the generalisation of *tuples* and *tables*.

Definition 6 (Distortions of Generalisation of Tuples). *Let $t = \{v_1, v_2, \ldots, v_m\}$ be a tuple and $t' = \{v'_1, v'_2, \ldots, v'_m\}$ be a generalised tuple of t. Let $\mathrm{level}(v_j)$ be the domain level of v_j in an attribute hierarchy. The distortion of this generalisation is defined as*

$$\mathrm{Distortion}(t, t') = \sum_{j=1}^{m} \mathrm{WHD}(\mathrm{level}(v_j), \mathrm{level}(v'_j))$$

For example, let the weights of WHD be defined by the uniform weight, attribute Gender be in hierarchy of $\{$M/F, * $\}$ and attribute Postcode be in hierarchy of $\{$dddd, ddd*, dd**, d***, * $\}$. Let t_3 be tuple 3 in Table 1a and t'_3 be tuple 3 in Table 1b. For attribute Gender, $\mathrm{WHD} = 1$. For attribute Age, $\mathrm{WHD} = 0$. For attribute Postcode, $\mathrm{WHD} = 1/4 = 0.25$. Therefore, $\mathrm{Distortion}(t_3, t'_3) = 1.25$.

Definition 7 (Distortions of Generalisation of Tables). *Let view D' be generalised from table D, t_i be the i-th tuple in D and t'_i be the i-th tuple in D'. The distortion of this generalisation is defined as*

$$\mathrm{Distortion}(D, D') = \sum_{i=1}^{|D|} \mathrm{Distortion}(t_i, t'_i)$$

where $|D|$ is the number of tuples in D.

From Table 1a and 1b, $\mathrm{WHD}(t_1, t'_1) = \mathrm{WHD}(t_2, t'_2) = \mathrm{WHD}(t_5, t'_5) = \mathrm{WHD}(t_6, t'_6)$ = 0 and $\mathrm{WHD}(t_3, t'_3) = \mathrm{WHD}(t_4, t'_4) = 1.25$. The distortion between the two tables is $\mathrm{Distortion}(D, D') = 1.25 + 1.25 = 2.5$.

4 Generalisation Distances

In this section, we map distortions to distances and discuss the properties of the mapped distances.

4.1 Distances between Tuples and equivalence classes

An objective of k-anonymisation is to minimise the overall distortions between a generalised table and the original table. We first consider how to minimise distortions when generalising two tuples into an equivalence class.

Definition 8 (Closest Common Generalisation). *All allowable values of an attribute form a hierarchical value tree. Each value is represented as a node in the tree, and a node has a number of child nodes corresponding to its more specific values. Let t_1 and t_2 be two tuples. t_{12} is the closest common generalisation of t_1 and t_2 for all i. The value of the closest common generalisation t_{12} is*

$$v_{12}^i = \begin{cases} v_1^i & \textit{if } v_1^i = v_2^i \\ \textit{the value of the closest common ancestor} & \textit{otherwise} \end{cases}$$

where, v_1^i, v_2^i, and v_{12}^i are the values of the i-th attribute in tuples t_1, t_2 and t_{12}.

For example, Figure 1b shows a simplified hierarchical value tree with 4 domain levels and $2^{(l-1)}$ values for each domain level l. Node 0** is the closest common ancestor of nodes 001 and 010 in the hierarchical value tree. Consider another example. Let $t_1 = \{male, young, 4351\}$ and $t_2 = \{female, young, 4352\}$. $t_{12} = \{*, young, 435*\}$.

Now, we define the distance between two tuples.

Definition 9 (Distance between Two Tuples). *Let t_1 and t_2 be two tuples and t_{12} be their closest common generalisation. The distance between the two tuples is defined as*

$$\mathrm{Dist}(t_1, t_2) = \mathrm{Distortion}(t_1, t_{12}) + \mathrm{Distortion}(t_2, t_{12})$$

For example, let the weights of WHD be defined by the uniform weights, attribute Gender be in hierarchy of {M/F, * } and attribute Postcode be in hierarchy of {dddd, ddd*, dd**, d***, * }. $t_1 = \{male, young, 4351\}$ and $t_2 = \{female,\ young,\ 4352\}$. $t_{12} = \{*, young, 435*\}$. Dist$(t_1, t_2) = \mathrm{Distortion}(t_1, t_{12}) + \mathrm{Distortion}(t_2, t_{12}) = 1.25 + 1.25 = 2.5$.

We discuss some properties of tuple distance in the following.

Lemma 1. *Basic properties of tuple distances*
(1) $\mathrm{Dist}(t_1, t_1) = 0$ *(i.e. a distance between two identical tuples is zero)*
(2) $\mathrm{Dist}(t_1, t_2) = \mathrm{Dist}(t_2, t_1)$ *(i.e. the tuple distance is symmetric), and*
(3) $\mathrm{Dist}(t_1, t_3) \leq \mathrm{Dist}(t_1, t_2) + \mathrm{Dist}(t_2, t_3)$ *(i.e. the tuple distance satisfies triangle inequality)*

Proof. The first two properties obviously follow Definition 9b. We prove property 3 here.

We first consider a single attribute. To make notions simple, we omit the superscript for the attribute. Let v_1 be the value of tuple t_1 for the attribute, v_{13} be the value of the generalised tuple t_{13} for the attribute from tuple t_1 and tuple t_3, and so forth.

Within a hierarchical value tree, $\mathrm{Dist}(t_1, t_3)$ is represented as the shortest path linking nodes v_1 and v_3 and $\mathrm{Dist}(t_1, t_2) + \mathrm{Dist}(t_2, t_3)$ is represented as the path linking v_1 and v_3 via v_2. Therefore, $\mathrm{Dist}(t_1, t_3) \leq \mathrm{Dist}(t_1, t_2) + \mathrm{Dist}(t_2, t_3)$. The two distances are equal only when v_2 is located within the shortest path between v_1 and v_3.

The overall distance is the sum of distances of all individual attributes. The above proof is true for all attributes. Therefore, the property 3 is proved.

An example of Property 3 can be found in the hierarchial value tree of Figure 1b. The distance between 00* and 011 is $(a + b + c)$, the distance between 00* and 010 is $(a + b + d)$, and the distance between 010 and 011 is $(c + d)$. Therefore, $\mathrm{Dist}(00*, 011) < \mathrm{Dist}(00*, 010) + \mathrm{Dist}(010, 011)$. In a special case, $\mathrm{Dist}(00*, 011) = \mathrm{Dist}(00*, 01*) + \mathrm{Dist}(01*, 011)$.

Now, we discuss distance between two groups of tuples.

Definition 10 (Distance between Two equivalence classes). *Let C_1 be an equivalence class containing n_1 identical tuples t_1 and C_2 be an equivalence class containing n_2 identical tuples t_2. t_{12} is the closest common generalisation of t_1 and t_2. The distance between two equivalence classes is defined as follows.*
$\mathrm{Dist}(C_1, C_2) = n_1 \times \mathrm{Distortion}(t_1, t_{12}) + n_2 \times \mathrm{Distortion}(t_2, t_{12})$

Note that t_{12} is the tuple that t_1 and t_2 will be generalised if two equivalence classes C_1 and C_2 are merged into one equivalence class. The distance is equivalent to the distortions of the generalisation and therefore the choice of merger should be those equivalence classes with the smallest distances.

5 Algorithm

In this section, we present an algorithm to implement k-anonymisation by local recoding.

The basic idea for the algorithm is finding an arbitrary equivalence class of size smaller than k and merging it with the closest equivalent classes to form a larger equivalent class with the smallest distortion. This process repeats recursively until each equivalent class contains at least k tuples.

We first discuss how to handle the situation that a small equivalent class (e.g. the class containing one tuple) merges to a large equivalent class (e.g. the class containing a hundred of tuples). Should we generalise the whole large equivalent class in order to absorb the small equivalent class? We should not. A better solution is to allocate a small number of tuples. For example, k-1 tuples from the large equivalent class are allocated to merge with the small equivalent class. As a result, information in most tuples of the larger equivalent class is preserved. the set of the tuples allocated in this way is called a *stub* and the set of the remaining tuples is called a *trunk*.

Definition 11 (Stub and Trunk of Equivalent Class). *Suppose a small equivalent class E_1 and a large equivalent class E_2 are to be generalised for k-anonymity. If $|E_1| < k$ and $|E_1| + |E_2| \geq 2k$, E_2 is split into two parts, a* stub *and a* trunk*. The stub contains $(k - |E_1|)$ tuples, and the trunk contains $(|E_1| + |E_2| - k)$ tuples.*

After this split, both the generalised equivalent class of E_1 with the stub and the remaining trunk of E_2 satisfy k-anonymity property. The detailed information in the trunk is preserved.

After this definition, we calculate the distance between two equivalent classes E_1 and E_2, where $|E_1| < k$, as follows.

- if $(|E_1| + |E_2| < 2k)$, calculate normal as in Definition 10.
- if $(|E_1| + |E_2| \geq 2k)$, calculate the distance between E_1 and the stub of E_2.

The pseudo code of the proposed algorithm is presented in Algorithm 1.

Algorithm 1 K-Anonymisation by Clustering in Attribute hierarchies (KACA)

```
1: form equivalence classes from the data set
2: while there exists an equivalence class of size $< k$ do
3:     randomly choose an equivalence class $C$ of size $< k$
4:     evaluate the pairwise distance between $C$ and all other equivalence classes
5:     find the equivalence class $C'$ with the smallest distance to $C$
6:     generalise the equivalence classes $C$ and $C'$
7: end while
```

Line 1 forms equivalent classes. Sorting data in a certain order will speed up the process. One tuple is also called an equivalent class. Normally, the number of equivalent classes is significantly less than the number of tuples in the data set.

The generalisation process continues in lines 2-6 when there is one or more equivalence classes whose size is smaller than k.

In each iteration, we randomly find an equivalence class C of size smaller than k in line 3. Then, we calculate the pairwise distances between C and all other equivalence classes in line 4. Line 5 finds the equivalence class C' with the smallest distance. Line 6 generalises the equivalence classes C and C'.

The above process terminates when there is no equivalent class whose size is smaller than k. The sizes of all equivalent classes are greater than or equal to k, and hence k-anonymity is achieved.

All tuples are sorted and only $O(n)$ passes is needed to find all equivalent classes. The complexity of this step is $O(nlog\ n)$. Let $|E|$ be the number of equivalent classes in line 2. Each iteration requires to choose an arbitrary equivalence class, which takes $O(1)$ time, evaluate the pairwise distance, which takes $O(|E|)$ time, find the equivalence class with the smallest distance, which takes $O(|E|)$ time, and finally generalise the equivalence class, which takes $O(1)$ time. Thus, the runtime of an iteration is $O(|E|)$. As there are $O(|E|)$ iterations, the overall runtime is $O(nlog\ n + |E|^2)$.

The above algorithm is easy to extend to handle outlier tuples, which are far away from all other tuples, by setting a minimum distance threshold in line 6 to avoid large distortions caused by generalising two distant equivalent classes. Outlier tuples are suppressed instead of generalised. We did not do this in this algorithm since in the next section we compare an optimal algorithm that does not suppress tuples.

6 Empirical Study

A Pentium IV 2.2GHz PC with 1GM RAM was used to conduct our experiments. The algorithm was implemented in C/C++. In our experiments, we adopted the publicly available data set, Adult Database, from the UCIrvine Machine Learning Repository [7]. This data set (5.5MB) was also adopted in [14, 16, 23, 10]. We also used a configuration similar to [14, 16]. We eliminated the records with unknown values. The resulting data set contains 45,222 tuples. Nine attributes were chosen as the quasi-identifier, as shown in Table 2.

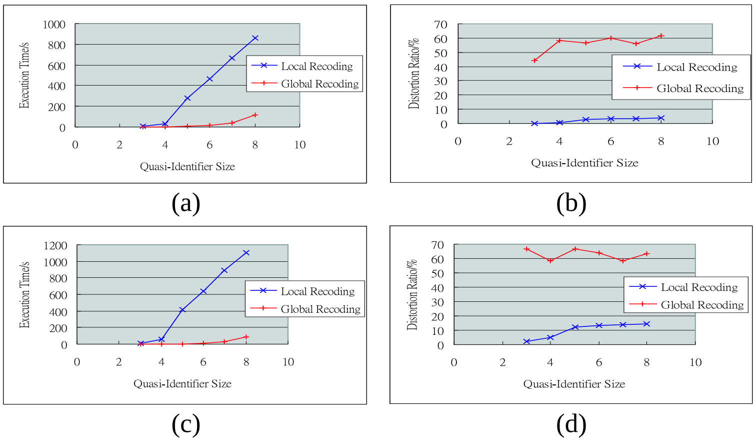
	Attribute	Distinct Values	Generalisations	Height
1	Age	74	5-, 10-, 20-year ranges	4
2	Work Class	7	Taxonomy Tree	3
3	Education	16	Taxonomy Tree	4
4	Martial Status	7	Taxonomy Tree	3
5	Occupation	14	Taxonomy Tree	2
6	Race	5	Taxonomy Tree	2
7	Sex	2	Suppression	1
8	Native Country	41	Taxonomy Tree	3
9	Salary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

Table 2. Description of Adult Data Set

We evaluated the proposed algorithm in terms of two measurements: execution time and distortion ratio. Let T be the original data set and T' be the data set generalised by an algorithm. Let T'' be the fully generalised data set, where all attributes of all tuples are generalised to the root of the hierarchy. Distortion ratio of a generalised data set T' is equal to the distortion of T' divided by the distortion of T''.

We conducted the experiments ten times and took the average execution time. We compared our algorithm KACA proposed with the best-known global recoding based algorithm Incognito [14].

We conducted the experiments with two types of distortion measures discussed in Section 3 - uniform weight and height weight. Figure 2 shows the results with uniform weight measurement.



(a)

(b)

(c)

(d)

Fig. 2. Execution Time and Distortion Ratio Versus Quasi-identifier Size (Uniform Weight) ($k = 2$ for (a) and (b) and $k = 10$ for (c) and (d))

Figure 2 shows that the execution time of both algorithms increases with the quasi-identifier size. On average, the execution time of the KACA algorithm is larger than that of the Incognito algorithm.

The distortion ratio increases with the quasi-identifer size. This is because it is less likely that two tuples in the original data set are equal to each other when the quasi-identifier size is greater. Thus, a larger distortion is needed. The distortion ratio of the KACA algorithm is 5.57 times lower than that of the Incognito algorithm on average. This is because, as we discussed before, the global recoding algorithm (Incognito algorithm) over-generalises the data set a lot while the KACA algorithm generalises the data set less extent for k-anonymity. When k increases, the distortion ratio of all algorithms increases. As we require more tuples to be identical for larger k, more distortions will be generated for larger k.

We have also conducted the experiments with height weight measurement. For the sake of space, we do not show the results as the results with height weight measurement are similar to Figure 2.

7 Conclusions

In this paper, we study how to achieve k-anonymity by clustering in attribute hierarchies. We define two general metrics of the generalised data sets to measure the quality of k-anonymisation. We define generalisation distances between tuples to characterise distortions of generalisations and discuss the properties of the distances. We conclude that the generalisation distance satisfies properties of metric distances. We propose an efficient algorithm to achieve k-anonymity by clustering in attribute hierarchical structures. We experimentally show that the proposed method causes significantly less distortions than an optimal global recoding k-anonymity method. The distortion ratio of our proposed algorithm is 5.57 times smaller on average.

Acknowledgement

This research was supported by the Innovation and Technology Fund (ITF) in the HKSAR [ITS/069/03] and the RGC Earmarked Research Grant of HKSAR CUHK 4120/05E. This research has also been partially supported by ARC Grant DP0559090, NSERC Grant 312194-05, and NSF Grant IIS-0308001.

References

1. C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 901–909. VLDB Endowment, 2005.
2. G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Anonymizing tables. In *ICDT05: Tenth International Conference on Database Theory*, pages 246–258, 2005.
3. G. Aggarwal, T. Feder, K. Kenthapadi, A. Zhu, R. Panigrahy, and D. Thomas. Achieving anonymity via clustering in a metric space. In *PODS '06: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 2006.
4. D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *PODS '01: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 247–255, New York, NY, USA, 2001. ACM Press.
5. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the ACM SIGMOD Conference on Management of Data*, pages 439–450. ACM Press, May 2000.
6. R. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE05: The 21st International Conference on Data Engineering*, pages 217–228, 2005.
7. E. K. C. Blake and C. J. Merz. UCI repository of machine learning databases, http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998.
8. J. Domingo-Ferrer and J. M. Mateo-Sanz. Practical data-oriented microaggregation for statistical disclosure control. *IEEE Transactions on Knowledge and Data Engineering*, 14(1):189–201, 2002.
9. J. Domingo-Ferrer and V. Torra. Ordinal, continuous and heterogeneous k-anonymity through microaggregation. *Data Mining and Knowledge Discovery*, 11(2):195–212, 2005.
10. B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *ICDE05: The 21st International Conference on Data Engineering*, pages 205–216, 2005.
11. A. Hundepool and L. Willenborg. μ-and τ- argus: software for statistical disclosure control. In *Third international seminar on statsitcal confidentiality*, Bled, 1996.
12. V. S. Iyengar. Transforming data to satisfy privacy constraints. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 279–288, 2002.
13. K. LeFevre, D. DeWitt, , and R. Ramakrishnan. Multidimensional k-anonymity. In *M. Technical Report 1521, University of Wisconsin*, 2005.
14. K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 49–60, 2005.
15. Y. Lindell and B. Pinkas. Privacy preserving data mining. *Journal of Cryptology*, 15(3):177–206, 2002.
16. A. Machanavajjhala, J. Gehrke, and D. Kifer. l-diversity: privacy beyond k-anonymity. In *To appear in the 22st International Conference on Data Engineering (ICDE06)*, 2006.
17. A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *PODS04: Proceedings of the twenty fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 223–228, 2004.
18. S. Rizvi and J. Haritsa. Maintaining data privacy in association rule mining. In *Proceedings of the 28th Conference on Very Large Data Base (VLDB02)*, pages 682–693. VLDB Endowment, 2002.
19. P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.
20. 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.
21. L. Sweeney. k-anonymity: a model for protecting privacy. *International journal on uncertainty, Fuzziness and knowldege based systems*, 10(5):557 – 570, 2002.
22. J. Vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 24–27, Washington, DC, 2003. ACM Press.
23. K. Wang, P. S. Yu, and S. Chakraborty. Bottom-up generalization: A data mining solution to privacy protection. In *ICDM04: The fourth IEEE International Conference on Data Mining*, pages 249–256, 2004.
24. R. Wright and Z. Yang. Privacy-preserving bayesian network structure computation on distributed heterogeneous data. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 713–718, New York, NY, USA, 2004. ACM Press.