# Anonymisation by Local Recoding in Data with Attribute Hierarchical Taxonomies

Jiuyong Li[+], Raymond Chi-Wing Wong[*], Ada Wai-Chee Fu[*] and Jian Pei[♯]

[+]School of Computer and Information Science, University of South Australia, Australia

[*]Department of Computer Science and Engineering, The Chinese University of Hong Kong

[♯]School of Computing Science, Simon Fraser University, Canada

jiuyong.li@unisa.edu.au, cwwong,adafu@cse.cuhk.edu.hk, jpei@cs.sfu.ca

March 3, 2008

**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. Among a number of $k$-anonymisation schemes, local recoding methods are promising for minimising the distortion of a $k$-anonymity view. This paper addresses two major issues in local recoding $k$-anonymisation in attribute hierarchical taxonomies. Firstly, we define a proper distance metric to achieve local recoding generalisation with small distortion. Secondly, we propose a means to control the inconsistency of attribute domains in a generalised view by local recoding. We show experimentally that our proposed local recoding method based on the proposed distance metric produces higher quality $k$-anonymity tables in three quality measures than a global recoding anonymisation method, Incognito, and a multidimensional recoding anonymisation method, Multi. The proposed inconsistency handling method is able to balance distortion and consistency of a generalised view.

**Key words:** $k$-anonymisation, local recoding, generalisation distance, inconsistency.

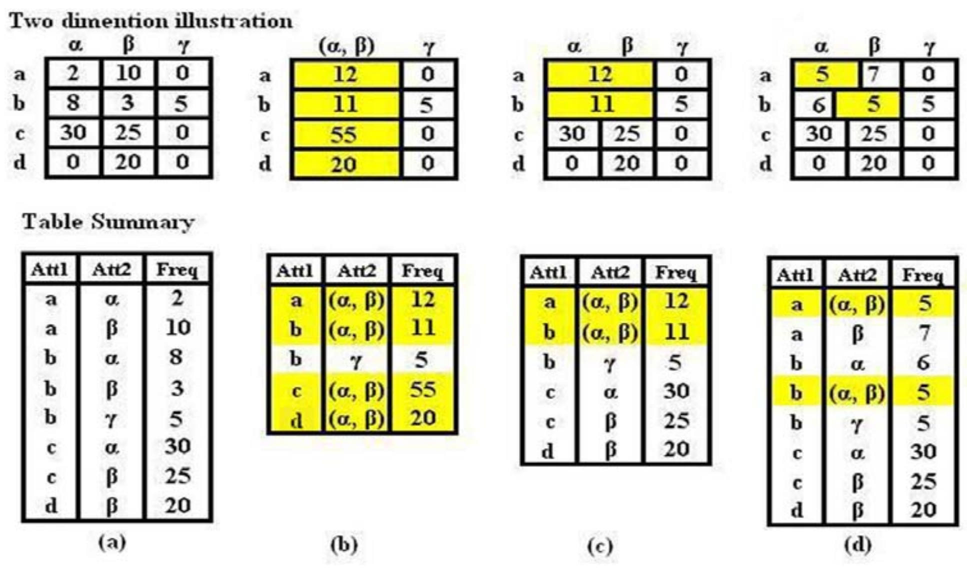
## 1 Introduction

A vast amount of operational data and information has been stored by different vendors and organizations. Most of the stored data is useful only when it is shared and analysed with other related data. However this kind of data often contains some personal details and sensitive information. The data can only be allowed to be released when individuals are unidentifiable. $k$-anonymity has emerged as an effective approach in anonymisation [18, 19, 20].

### 1.1 K-anonymisation and various methods

The key idea of $k$-anonymisation is to make individuals indistinguishable in a released table. A tuple representing an individual within the identifiable attributes has to be identical to at least $(k-1)$ other tuples. The larger the value of $k$ is, the better the protection. One way to produce $k$ identical tuples within the identifiable attributes is to generalise values within the attributes, for example, removing day and month information in a Date-of-Birth attribute. A general view of attribute generalisation is the aggregation of attribute values. $k$-anonymity has been extensively studied in recent years [4, 7, 9, 10, 22].

Various approaches for generalisation have been studied, such as global recoding generalisation [4, 7, 9, 18, 19, 22], multidimensional recoding generalisation [10], and local recoding generalisation [6, 15, 24]. Global recoding generalisation maps the current domain of an attribute to a more general domain. For example, ages are mapped from years to 10-year intervals. Multidimensional recoding generalisation (or multidimensional global recoding generalisation by LeFevre et al. [10]) maps a set


Two dimention illustration

(a)

| | α | β | γ |
|---|---|---|---|
| a | 2 | 10 | 0 |
| b | 8 | 3 | 5 |
| c | 30 | 25 | 0 |
| d | 0 | 20 | 0 |

(b)

| | (α, β) | γ |
|---|---|---|
| a | 12 | 0 |
| b | 11 | 5 |
| c | 55 | 0 |
| d | 20 | 0 |

(c)

| | α | β | γ |
|---|---|---|---|
| a | 12 | | 0 |
| b | 11 | | 5 |
| c | 30 | 25 | 0 |
| d | 0 | 20 | 0 |

(d)

| | α | β | γ |
|---|---|---|---|
| a | 5 | 7 | 0 |
| b | 6 | 5 | 5 |
| c | 30 | 25 | 0 |
| d | 0 | 20 | 0 |

Table Summary

(a)

| Att1 | Att2 | Freq |
|---|---|---|
| a | α | 2 |
| a | β | 10 |
| b | α | 8 |
| b | β | 3 |
| b | γ | 5 |
| c | α | 30 |
| c | β | 25 |
| d | β | 20 |

(b)

| Att1 | Att2 | Freq |
|---|---|---|
| a | (α, β) | 12 |
| b | (α, β) | 11 |
| b | γ | 5 |
| c | (α, β) | 55 |
| d | (α, β) | 20 |

(c)

| Att1 | Att2 | Freq |
|---|---|---|
| a | (α, β) | 12 |
| b | (α, β) | 11 |
| b | γ | 5 |
| c | α | 30 |
| c | β | 25 |
| d | β | 20 |

(d)

| Att1 | Att2 | Freq |
|---|---|---|
| a | (α, β) | 5 |
| a | β | 7 |
| b | α | 6 |
| b | (α, β) | 5 |
| b | γ | 5 |
| c | α | 30 |
| c | β | 25 |
| d | β | 20 |

Figure 1: An illustration of different methods to achieve $k$-anonymity. A table has two attributes Att1 and Att2. $\{a, b, c, d\}$ and $\{\alpha, \beta, \gamma\}$ are the original domains for Att1 and Att2. Top tables summarise the number of data points in separate regions of the two dimensional space. The bottom tables summarise frequencies of identical tuples. (a) the original data; (b) generalisation by a global recoding approach; (c) generalisation by a multidimensional recoding approach; and (d) generalisation by a local recoding approach.

of values to another set of values, some or all of which are more general than the corresponding pre-mapping values. For example, {male, 32, divorce} is mapped to {male, [30,40), unknown}. Local recoding generalisation modifies some values in one or more attributes to values in more general domains. We will illustrate differences between multidimensional recoding generalisation and local recoding generalisation in the following.

A general view of $k$-anonymity is clustering with the constraint of the minimum number of objects in every cluster. Data records are mapped to data points in a high dimensional space. When a region partitioned by attribute values has fewer than $k$ data points, individuals represented by data points are at risk of being identified. The region needs to be merged with other regions by generalising attribute values so that the merged region contains at least $k$ data points.

Global, multidimensional and local recoding generalisation can be explained in this way. Consider the two dimensional example in Figure 1(a) and let $k = 5$. Attribute values $(a, b, c, d)$ and $(\alpha, \beta, \gamma)$ partition the data space into 12 regions in Figure 1(a). Two regions, $[a, \alpha]$ and $[b, \beta]$, contain less than 5 but more than zero data points. Individuals in these two regions are likely to be identified. Therefore, they need to be merged with other regions to make the number of data points at least 5. In the global recoding generalisation scheme, a merged region stretches over the range of other attributes. For example, the merged rectangle in Figure 1(b) covers all values of Attribute 1 since all occurrences of $\alpha$ and $\beta$ in Attribute 2 have to be generalised. The merged regions and the summary of the corresponding generalised table are listed in Figure 1(b). In a table view, domain $(\alpha, \beta, \gamma)$ is mapped to domain $(\alpha, [\beta, \gamma])$. The global recoding generalisation causes some unnecessary mergers, for example, regions $[c, (\alpha, \beta)]$ and $[d, (\alpha, \beta)]$. This is the over generalisation problem of global recoding generalisation. For the multidimensional generalisation scheme, any two or more regions can be merged as long as the aggregated attribute value such as $[\beta, \gamma]$ makes sense. For example, regions $[a, \alpha]$ and $[a, \beta]$ merge into region $[a, (\alpha, \beta)]$, and regions $[b, \alpha]$ and $[b, \beta]$ merge into region $[b, (\alpha, \beta)]$. Regions $[c, \alpha]$, $[c, \beta]$, $[d, \alpha]$ and $[d, \beta]$ keep their original areas, see Figure 1(c). In a table view, all tuples $(a, \alpha)$ and $(a, \beta)$ are mapped to $(a, [\alpha, \beta])$ and all tuples $(b, \alpha)$ and $(b, \beta)$ are mapped to $(b, [\alpha, \beta])$, but tuples $(c, \alpha)$, $(c, \beta)$, and $(d, \beta)$ remain unchanged. A local recoding generalisation method is even more flexible, see Figure 1(d). It does not merge whole regions. A dense region can be split into two or more overlapping regions, and some merge with other regions. For example, region $[a, \beta]$ is split into two overlapping regions containing 3 and 7 data points each. The 3 data point region is merged with region $[a, \alpha]$ to form region $[a, (\alpha, \beta)]$ with 5 data points. Both multidimensional and local recoding approaches do not over generalise a table. In a table view, some tuples of $(a, \alpha)$ and $(a, \beta)$ are mapped to $(a, [\alpha, \beta])$, and some tuples of $(b, \alpha)$ and $(b, \beta)$ are mapped to $(b, [\alpha, \beta])$, but some remain unchanged in their original forms.

## 1.2 Existing problems and our contributions

Multidimensional and local recoding methods can improve the quality of anonymisation by reducing the amount of generalization. A number of research works have been conducted in this direction [1, 6, 10, 15, 24]. However, most works focus on numerical and ordinal attributes. Two works [1, 15] handle unordered categorical attributes, but both employ a simplified suppression model: values either exist or are unknown. They do not consider attribute hierarchical structures. Work in [24] touches upon attribute hierarchical structures, but the approach is fundamentally an numerical one. More discussions on the work are given in Section 2.

There is an opportunity for studying multidimensional and local recoding $k$-anonymisation in attribute hierarchies. When attributes are numerical or ordinal, their distances can be measured by the Euclidean distance or other similar metrics. However, not every attribute can be ordered. Attribute hierarchial taxonomies provide meaningful groups in a released table. Two immediate questions will be: How can we measure distances of data objects in attribute hierarchies? And how can we link the metric to the quality objective of $k$-anonymisation? This paper will discuss these problems.

One major drawback of multidimensional and local recoding generalisation methods is that they produce tables with inconsistent attribute domains. For example, generalised values $(\alpha, \beta)$, and un-generalised values $\alpha$ and $\beta$ co-exist in Attribute 2 in Figure 1(c) and 1(d). This may cause difficulty when analysing the table in many real world applications. We will initiate discussions of inconsistency problem of local recoding generalisation, and study a approach to handle inconsistent domains of a generalised table.

This paper extends our work reported in [13]. In addition to defining a distance metric and splitting an equivalence class to a stub and a trunk for local recoding, we add comprehensive discussions on the inconsistency problem of local recoding and possible solutions. We also upgrade the experimental comparisons from comparing with a global recoding method based on one quality metric to comparing with both global and multidimensional recoding methods based on four quality metrics.

## 2 Related work

In general, there are three categories of privacy preserving methods in the data mining literature. The first category consists of perturbation methods, typified by [2, 3, 17]. These methods make use of randomised techniques to perturb data and statistical techniques to reconstruct distribution of data. The second category comprises of cryptographic methods, such as [14, 21, 23]. Cryptographic techniques have been used to encrypt data so that neither party can see other parties' data when they share data to work out common interesting solutions. The third category includes $k$-anonymity methods, such as [18, 20]. A $k$-anonymity method de-identifies a data set so that individuals in the data set cannot be identified. Our study belongs to this category.

$k$-anonymisation methods are generally divided into two groups: task-specific and non-specific methods. For task-specific $k$-anonymisation, the released tables are undergoing some specific data mining processes (e.g. building decision tree models). The purpose of anonymisation is to keep sufficient protection of sensitive information while maintaining the precision for data mining tasks, such as classification accuracy. There have been a number of proposals in this group [7, 8, 22]. In most cases, data owners do not know the ultimate use of the released tables. Therefore a general anonymisation goal should not be associated with a specific data mining task, but should minimise distortions in the released table. The methods in this category are called non-specific $k$-anonymisation methods (e.g. [1, 4, 9, 15, 18, 19]).

An alternative taxonomy of $k$-anonymisation methods includes three groups: global, multidimensional and local recoding methods. LeFevre et al. [10] divide multidimensional recoding methods into global and local methods. In this paper, multidimensional recoding means multidimensional global recoding. Local recoding includes multidimensional and single dimensional local recoding. Justifications for our classification are provided in Section 3.

Global recoding methods generalise a table at the domain level. Many works of $k$-anonymisation are based on the global recoding model, such as [4, 7, 8, 9, 18, 19, 22]. A typical global recoding generalisation method is Incognito [9]. Incognito produces minimal full-domain generalisations. Incognito is the first algorithm for the minimal full-domain generalization on large databases. A global recoding method may over-generalise a table. For example, to protect a male patient in a specific region, postcodes of thousands of records are generalised even though there are a lot of male patients in other regions which can have their original postcodes.

Both multidimensional and local recoding methods generalise a table at cell levels. They do not over generalise a table and hence may minimise the distortion of an anonymity view. LeFevre et al. first studied the multidimensional recoding problem [10], and proposed an efficient partition method, Multi, for multidimensional recoding anonymisation. Aggarwal et al. [1] and Meyerson et al. [15] analysed the computational complexity of local recoding methods on a simplified model: suppressing values only. Both conclude that optimal $k$-anonymisation, minimising the number of cells being suppressed, is NP-hard. Some new local recoding works are reported in [6, 24]. These works mainly deal with numerical and ordinal attributes. Although work in [24] touches on hierarchical attributes, its quality metric for hierarchical attributes is a direct extension of that for numerical attributes. The quality of generalising categorical values in [24] is determined by the number of distinct values in a generalised category and the total number of distinct values of the attribute, but not by hierarchical structures. Consider two attributes with the same number of distinct values. Information losses of two generalisations are the same if the generalised categories include the same number of distinct values, although their hierarchical structures are different. In contrast, the generalisation distance in this paper is determined by hierarchical structures.

Other typical approaches to achieve $k$-anonymity are through clustering [2, 5]. These methods normally handle numerical and ordinal attributes, and are not global recoding methods. They use different representations, such as mean values instead of intervals as in generalisation. For data sets with numerical attributes, there are rarely identical tuples in the quasi-identifier attribute set, defined in Section 3, (overlapping data points in a data space) since there are too many distinct values in each attribute. Therefore, there is not a point to distinguish local recoding and multidimensional recoding. In general, most $k$-anonymity methods can be interpreted as variant clustering approaches, either through division or agglomeration. Local and multidimensional recoding methods are differentiated by whether overlapping clusters are allowed.

## 3 Problem Definitions

The objective of $k$-anonymisation is to make every tuple in identity-related attributes of a published table identical to at least $(k-1)$ other tuples. Identity-related attributes are those which potentially identify individuals in a table. For example, the record describing a middle aged female in the suburb with the postcode of 4352 is unique in Table 1, and hence her problem of stress may be revealed if the table is published. To preserve her privacy, we may generalise Gender and Postcode attribute values such that each tuple in attribute set $\{$Gender, Age, Postcode$\}$ has at least two occurrences. A view after this generalisation is given in Table 1(b). We provide running examples based on Table 1.

Since various countries use different postcode schemes, in this paper, we adopt a simplified postcode scheme, where its hierarchy $\{$4201, 420*, 42**, 4***, *$\}$ corresponds to $\{$suburb, city, region, state, unknown$\}$, respectively. A tuple for an attribute set in a record is an ordered list of values corresponding to the attribute set in the record.

**Definition 1 (Quasi-identifier attribute set)** *A* quasi-identifier attribute set *(QID) is a set of attributes in a table that potentially identify individuals in the table.*

For example, attribute set $\{$Gender, Age, Postcode$\}$ in Table 1(a) is a quasi-identifier. Table 1(a) potentially reveals private information of patients (e.g. the problem of stress

| No. | Gender | Age | Postcode | Problem |
|---|---|---|---|---|
| 1 | male | middle | 4350 | stress |
| 2 | male | middle | 4350 | obesity |
| 3 | male | middle | 4350 | obesity |
| 4 | female | middle | 4352 | stress |
| 5 | female | old | 4353 | stress |
| 6 | female | old | 4353 | obesity |

(a)

| No. | Gender | Age | Postcode | Problem |
|---|---|---|---|---|
| 1 | * | middle | 435* | stress |
| 2 | * | middle | 435* | obesity |
| 3 | * | middle | 435* | obesity |
| 4 | * | middle | 435* | stress |
| 5 | * | old | 435* | stress |
| 6 | * | old | 435* | obesity |

(b)

| No. | Gender | Age | Postcode | Problem |
|---|---|---|---|---|
| 1 | male | middle | 4350 | stress |
| 2 | male | middle | 4350 | obesity |
| 3 | male | middle | 4350 | obesity |
| 4 | female | * | 435* | stress |
| 5 | female | * | 435* | stress |
| 6 | female | * | 435* | obesity |

(c)

| No. | Gender | Age | Postcode | Problem |
|---|---|---|---|---|
| 1 | male | middle | 4350 | stress |
| 2 | male | middle | 4350 | obesity |
| 3 | * | middle | 435* | obesity |
| 4 | * | middle | 435* | stress |
| 5 | female | old | 4353 | stress |
| 6 | female | old | 4353 | obesity |

(d)

Table 1: (a) A raw table. (b) A 2-anonymity view by global recoding. (c) A 2-anonymity by multidimensional recoding. (d) A 2-anonymity by local recoding.

of the middle-aged female). Normally, a quasi-identifier attribute set is specified 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, 2 and 3 in Table 1(a) 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 attribute set if the size of every equivalence class with respect to the attribute set is $k$ or more.*

$k$-anonymity requires that every tuple occurrence for a given quasi-identifier attribute set has a frequency of at least $k$. For example, Table 1(a) does not satisfy $2$-anonymity property since the tuple {female, middle, 4352} occurs once.

**Definition 4 ($k$-anonymisation)** $k$-anonymisation *is a process to modify a table to a* *view that satisfies the k-anonymity property with respect to the quasi-identifier.*

For example, Table 1(b) is a 2-anonymity view of Table 1(a) since the size of all equivalence classes with respect to the quasi-identifier $\{$Gender, Age, Postcode$\}$ is at least 2.

A table may have more than one $k$-anonymity view, but some are better than others. For example, we may have other 2-anonymity views of Table 1(a) as in Table 1(c) and Table 1(d). Table 1(b) loses more detail than Table 1(c) and Table 1(d).

Therefore, another objective for $k$-anonymisation is to minimise distortions. We will give a definition of distortion later in Section 4. Initially, we consider it as the number of cells being modified.

There are three ways to achieve $k$-anonymity, namely *global recoding*, *multidimensional recoding*, and *local recoding*. LeFevre divided multidimensional recoding as having two subtypes [10]: global and local methods. Though the multidimensional global recoding contains the word of global, its generalised tables are quite different from those from the global recoding generalisation but are closer to those from the local recoding generalisation. Both multidimensional and local recoding methods produce tables with mixed values from different domains in a field whereas all values are from the same domain in a field of a globally generalised table. To avoid confusion, we use the terminology 'multidimensional recoding' instead of 'multidimensional global recoding'. It is not significant to distinguish multidimensional and single dimensional local recoding since it does not lead to different approaches to generalise one value or more values in a tuple for local recoding. We call both local recoding.

Another name for global recoding is *domain generalisation*. The generalisation happens at the domain level. A specific domain is replaced by a more general domain. There are no mixed values from different domains in a table generalised by global recoding. When an attribute value is generalised, every occurrence of the value is replaced by the new generalised value. A global recoding method may *over generalise* a table. An example of global recoding is given in Table 1(b). Two attributes Gender and Postcode are generalised. All gender information has been lost. It is not necessary to generalise the Gender and the Postcode attribute as a whole. So, we say that the global recoding method over-generalises this table.

Multidimensional and local recoding methods generalise attribute values at cell level. They generalise cell values when necessary for $k$-anonymity. Values from different domains co-exist in a field of a generalised table. They do not over generalise a table, and hence they may minimise the distortion of an anonymous view. Tables generalised by multidimensional and local recoding methods are given in Table 1(c) and Table 1(d). Another interpretation of multidimensional and local recoding is that they map a set of values to another set of values. The difference between multidimensional recoding and local recoding generalisation is that the former does not allow an equivalence class to be mapped to two or more equivalence classes while the latter does. For example, three equivalence classes in Table 1(a) are generalised to two equivalence classes in Table 1(c). The two equivalence classes {female, middle, 4352} and {female, old, 4353} are generalised to one equivalence class {female, *, 435*}. No equivalence class is split, and this is a result of multidimensional recoding. Three equivalence classes in Table 1(a) are generalised to three equivalence classes in Table 1(d). The equivalence class {male, middle, 4350} is split into two identical equivalence classes. One contains the first two tuples, $t_1$ and $t_2$, and the other contains the third tuple, $t_3$. The equivalence class containing $t_3$ is generalised with the equivalence class containing $t_4$. The equivalence class containing $t_1$ and $t_2$ remains un-generalised. Therefore, Table 1(d) is a result of local recoding. A large equivalence class may be generalised into a number of equivalence classes in local recoding.

There are many possible ways for local recoding generalisation. Aggarwal et al. [1] and Meyerson et al. [15] analyse a simplified local recoding model where values either exist or are suppressed. When the optimisation goal is to minimise cells being suppressed, both papers conclude that optimal $k$-anonymisation by local recoding is NP-hard. Therefore, heuristic methods are typically employed in local recoding generalisation.

## 4 Measuring the Quality of $k$-anonymisation

In this section, we discuss metrics for measuring the quality of $k$-anonymisation generalisation.

There are a number of quality measurements presented in previous studies. Many metrics are utility based, for example, model accuracy [7, 11] and query quality [10, 24]. They are associated with some specific applications. Two generic metrics have been used in a number of recent works.

The Discernability metric was proposed by Bayardo et al. [4] and has been used in [10, 24]. It is defined in the following:

$$\mathrm{DM} = \sum_{\mathrm{EquivClasses\,E}} |\,\mathrm{E}\,|^2$$

where $|E|$ is the size of equivalence class $E$. The cost of anonymisation is determined by the size of equivalence classes. An optimisation objective is to minimise discernability cost.

Normalised average equivalence class size was proposed by LeFevre et al. [10], and has been used in [24]. It is defined as the following.

$$\mathrm{CAVG} = (\frac{\mathrm{total\_records}}{\mathrm{total\_equiv\_classes}})/(k)$$

The quality of $k$-anonymisation is measured by the average size of equivalence classes produced. An objective is to reduce the normalised average equivalence class size.

These measurements are mathematically sound, but are not intuitive to reflect changes being made to a table. In this paper, we use the most generic criterion, called distortion. It measures changes caused by generalisation.

A simple measurement of distortion is the *modification rate*. For a $k$-anonymity view $V$ of table $T$, the *modification rate* is the fraction of cells being modified within the quasi-identifier attribute set. For example, modification rate from Table 1(a) to Table 1(b) is 66.7% and modification rate from Table 1(a) to Table 1(c) is 33.3%.

This criterion does not consider attribute hierarchical structures. For example, the distortion caused by the generalisation of Postcode from suburb to city is significantly different from the distortion caused by the generalisation of Gender from male/female to *. The former still keeps some information of location, but the latter loses all information of sex. The modification rate is too simple to reflect such differences.

We first define a metric measuring the distance between different levels in an attribute hierarchy.

**Definition 5 (Weighted Hierarchical Distance (WHD))** *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*

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

Figure 2 show two examples of attribute hierarchies, and Figure 3 shows the numbering method of hierarchical levels and weights between hierarchical levels. Level 1 is always the most general level of a hierarchy and contains one value, unknown.

We have the following two simple but typical definitions for weight $w_{j,j-1}$ in generalisation.

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

In this scheme, WHD is ratio of the steps a cell being generalised to all possible generalisation steps (the height of a hierarchy). For example, let the Date-of-Birth hierarchy be {day/month/year, month/year, year, 10year-interval, child/youth/middle-age/old-age, *$\}$. WHD of the generalisation from day/month/year to year is $\mathrm{WHD}(6, 4) = (1 + 1)/5 = 0.4$. In a Gender hierarchy, $\{$male/female, *$\}$, WHD from male/female to * is $\mathrm{WHD}(2, 1) = 1/1 = 1$. This means that distortion caused by the generalisation of five cells from day/month/year to year is equivalent to distortion caused by the generalisation of two cells from male/female to *.

This scheme does not capture the fact that generalisations at different levels yield different distortions. A generalisation nearer to the root of the hierarchy distorts a value more than a generalisation further away from the root. For example, in the Date-of-Birth hierarchy distortion caused by the generalisation of a value from day/month/year to month/year is less than distortion caused by the generalisation from year to 10year-interval. This example motivates us to propose another scheme.

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

The intuition is that generalisation nearer the root results in larger distortion than generalisation further away from the root. In this scheme, weights nearer the root are larger than weights further away from the root. For example, in the Date-of-Birth attribute, let $\beta = 1$, WHD of the generalisation from day/month/year to year is $\mathrm{WHD}(6, 5) = (1/5)/(1/5 + 1/4 + 1/3 + 1/2 + 1) = 0.087$. In the Gender hierarchy $\{$male/female, *$\}$, WHD from male/female to * is $\mathrm{WHD}(2, 1) = 1/1 = 1$. Distortion caused by the generalisation of one cell from male/female to * in the Gender attribute is more than distortion caused by the generalisation of 11 (i.e. $1/0.087$) cells from day/month/year to month/year in the Date-of-Birth attribute. If a user wants to penalise more on the generalisation close to the root, $\beta$ can be set to a larger value (e.g. 2).

There are other possible schemes for various applications. An immediate enhancement is to assign weights by attribute. We adopt simple schemes for better illustration in this paper. 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 $t$ where $m$ is the number of attributes in the quasi-identifier. 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 weights of WHD be defined by the uniform weight scheme, the attribute Gender be in the hierarchy of $\{$male/female, *$\}$ and attribute Postcode be in the hierarchy of $\{$dddd, ddd*, dd**, d***, *$\}$. Let $t_4$ be tuple 4 in Table 1(a) and $t'_4$ be tuple 4 in Table 1(b). For attribute Gender, $\mathrm{WHD} = 1$. For attribute Age, $\mathrm{WHD} = 0$. For attribute Postcode, $\mathrm{WHD} = 1/4 = 0.25$. Therefore, $\mathrm{Distortion}(t_4, t'_4) = 1.25$.

**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 $D$.*

For example, from Table 1(a) to Table 1(b), $\mathrm{WHD}(t_1, t'_1) = \ldots = \mathrm{WHD}(t_6, t'_6)$ = 1.25. The distortion between the two tables is $\mathrm{Distortion}(D, D') = 1.25 \times 6 = 7.5$.

## 5 Generalisation Distances

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

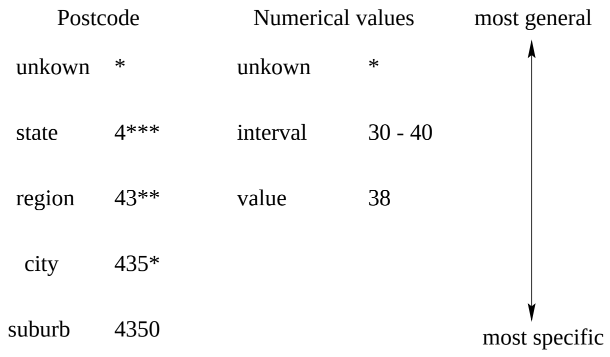


Figure 2: Examples of domain hierarchies.

## 5.1 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. $t_{1,2}$ is the closest common generalisation of $t_1$ and $t_2$ for all $i$. The value of the closest common generalisation $t_{1,2}$ is*

$$v^i_{1,2} = \begin{cases} v^i_1 & \text{if } v^i_1 = v^i_2 \\ \text{the value of the closest common ancestor} & \text{otherwise} \end{cases}$$

*where, $v^i_1$, $v^i_2$, and $v^i_{1,2}$ are the values of the $i$-th attribute in tuples $t_1$, $t_2$ and $t_{1,2}$.*

For example, Figure 3 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 $t_2 = \{female, young, 4352\}$. $t_{1,2} = \{*, young, 435*\}$.

Now, we define the distance between two tuples.

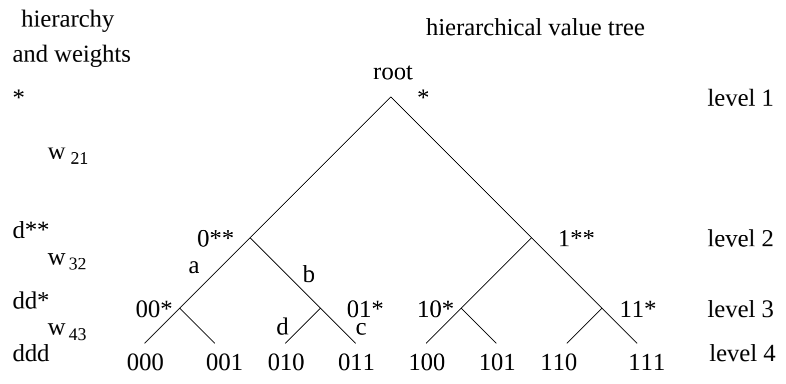


Figure 3: An example of weights and a simplified hierarchical value tree.

**Definition 9 (Distance between two tuples)** *Let $t_1$ and $t_2$ be two tuples and $t_{1,2}$ be their closest common generalisation. The distance between the two tuples is defined as*

$$\mathrm{Dist}(t_1, t_2) = \mathrm{Distortion}(t_1, t_{1,2}) + \mathrm{Distortion}(t_2, t_{1,2})$$

For example, let weights of WHD be defined by the uniform weight scheme, attribute Gender be in the hierarchy of {male/female, *} and attribute Postcode be in the hierarchy of {dddd, ddd*, dd**, d***, *}. $t_1 = \{male, young, 4351\}$ and $t_2 = \{female, young, 4352\}$. $t_{1,2} = \{*, young, 435*\}$. Dist$(t_1, t_2) = \mathrm{Distortion}(t_1, t_{1,2})$ $+ \mathrm{Distortion}(t_2, t_{1,2}) = 1.25 + 1.25 = 2.5$.

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 9. We prove property 3 here.

We first consider a single attribute. To make notations simple, we omit the superscript for the attribute. Let $v_1$ be the value of tuple $t_1$ for the attribute, $v_{1,3}$ be the value of the generalised tuple $t_{1,3}$ 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. This proof is true for all attributes. Therefore, the property 3 is proved. $\square$

An example of Property 3 can be found in the hierarchial value tree of Figure 3. 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 $t_2$. $t_{1,2}$ is the closest common generalisation of $t_1$ and $t_2$. The distance between two equivalence classes is defined as follows.*

$\mathrm{Dist}(C_1, C_2) = n_1 \times \mathrm{Distortion}(t_1, t_{1,2}) + n_2 \times \mathrm{Distortion}(t_2, t_{1,2})$

Note that $t_{1,2}$ is the tuple that $t_1$ and $t_2$ will be generalised to if the two equivalence classes $C_1$ and $C_2$ are generalised into one equivalence class. The distance is equivalent to the distortions of the generalisation and therefore the choice of generalisation should be those equivalence classes with the smallest distances.

We consider a property of merging equivalence classes.

**Lemma 2** *Associative property of generalisation*
*Let $C_1$, $C_2$, and $C_3$ be equivalence classes containing single tuples $t_1$, $t_2$ and $t_3$ respectively, $C_{1,2}$ be the equivalence class containing two generalised tuples $t_{1,2}$ of $t_1$ and $t_2$ and $C_{2,3}$ be the equivalence class containing two generalised tuples $t_{2,3}$ of $t_2$ and*

$t_3$. *We have the following equality,* $\mathrm{Dist}(C_1, C_2) + \mathrm{Dist}(C_{1,2}, C_3) = \mathrm{Dist}(C_2, C_3) + \mathrm{Dist}(C_1, C_{2,3})$.

**Proof** We start with a single attribute, and consider the hierarchical value tree of the attribute. To make the notations simple, we omit the superscript for the attribute. Let $v_1$ be the value of tuple $t_1$ for the attribute, $v_{1,3}$ be the value of the generalised tuple $t_{1,3}$ for the attribute, and so forth.

Within this hierarchical tree, let node $v_{1,2,3}$ represent the closest common ancestor of $v_1$, $v_2$ and $v_3$. Each side of the equation is the sum of WHD from $v_1$, $v_2$ and $v_3$ to $v_{1,2,3}$. We use $\mathrm{Dist}(C_1, C_2) + \mathrm{Dist}(C_{1,2}, C_3)$ as an example to show this. $t_{1,2}$ is a descendant of $t_{1,2,3}$ (or is the same as $t_{1,2,3}$, and in this case the proof is even simpler.). $\mathrm{Dist}(C_1, C_2)$ sums the WHDs from $v_1$ and $v_2$ to $v_{1,2}$. $\mathrm{Dist}(C_{1,2}, C_3)$ sums the WHDs from $v_3$ to $v_{1,2,3}$ and twice the WHDs from $v_{1,2}$ to $v_{1,2,3}$, where one is for $v_1$ and the other is for $v_2$. Therefore, $\mathrm{Dist}(C_1, C_2) + \mathrm{Dist}(C_{1,2}, C_3)$ sums the WHDs from $v_1$, $v_2$ and $v_3$ to $v_{1,2,3}$.

The overall distance is the sum of the distances of individual attributes. The above proof is true for all attributes. The lemma is proved. □

The lemma shows distortions do not relate to the order of generalisation, but only relate to the elements in the generalised group.

## 6 Racing attributes and inconsistency

In local recoding generalisation, a decision of generalisation is made locally to minimise distortions. However, when there are a number of choices that cause the same amount of distortions, they lead to different outcomes. Let us start with an example. In Table 2(a), attributes Gender and Marriage form the quasi-identifier. The Gender attribute is in the hierarchy of {male/female, *} and the Marriage attribute is in the hierarchy of {married/unmarried/divorced/widowed, *}. In Table 2(a), $\mathrm{Dist}(t_1, t_2) = \mathrm{Dist}(t_1, t_3)$. If we choose to generalise $t_1$ and $t_3$ first, the resultant 2-anonymity view is in Table 2(b). If we choose to generalise $t_1$ and $t_2$ first, the resultant 2-anonymity view is in Table 2(c). Both views have the same distortions over the original table. If users do not have preferences, both views in Table 2 are acceptable.

| No. | Gender | Marriage | Problem |
|---|---|---|---|
| 1 | male | married | stress |
| 2 | male | unmarried | obesity |
| 3 | female | married | stress |
| 4 | female | unmarried | obesity |

(a)

| No. | Gender | Marriage | Problem |
|---|---|---|---|
| 1 | * | married | stress |
| 2 | * | unmarried | obesity |
| 3 | * | married | stress |
| 4 | * | unmarried | obesity |

(b)

| No. | Gender | Marriage | Problem |
|---|---|---|---|
| 1 | male | * | stress |
| 2 | male | * | obesity |
| 3 | female | * | stress |
| 4 | female | * | obesity |

(c)

Table 2: An example of racing attributes. (a) A raw table. (b) A 2-anonymity view. (c) An alternative 2-anonymity view.

When we consider a more complicated example as in Table 3, Table 3(c) is better than Table3(b). Although their distortions are identical, we may not be able to use both attributes Gender and Marriage in Table 3(b) since no reliable statistical or data mining results can be derived from both attributes, whereas Gender attribute in Table 3(c) is complete.

We call this phenomenon *racing attributes*. More precisely, we have the following definition.

**Definition 11 (Racing attributes)** *If Dist($C_1, C_2$) = Dist($C_1, C_3$) =* $\min_{\forall i,j} \mathrm{Dist}(C_i, C_j)$, *we call attributes involved in the generalisation of* $t_1$ *and* $t_2$ *and the generalisation of* $t_1$ *and* $t_3$ racing attributes.

When the smallest distance is between two or more equivalence class pairs and we are going to choose one pair to generalise, the attributes involved in generalising the tuples of equivalence classes are called racing attributes.

| No. | Gender | Marriage | Problem |
|---|---|---|---|
| 1 | male | married | stress |
| 2 | male | unmarried | obesity |
| 3 | female | married | stress |
| 4 | female | unmarried | obesity |
| 5 | male | divorced | stress |
| 6 | male | widowed | obesity |
| 7 | female | divorced | stress |
| 8 | female | widowed | obesity |

(a)

| No. | Gender | Marriage | Problem |
|---|---|---|---|
| 1 | * | married | stress |
| 2 | * | unmarried | obesity |
| 3 | * | married | stress |
| 4 | * | unmarried | obesity |
| 5 | male | * | stress |
| 6 | male | * | obesity |
| 7 | female | * | stress |
| 8 | female | * | obesity |

(b)

| No. | Gender | Marriage | Problem |
|---|---|---|---|
| 1 | male | * | stress |
| 2 | male | * | obesity |
| 3 | female | * | stress |
| 4 | female | * | obesity |
| 5 | male | * | stress |
| 6 | male | * | obesity |
| 7 | female | * | stress |
| 8 | female | * | obesity |

(c)

Table 3: Another example for racing attribute. (a) A raw table. (b) A 2-anonymity view. (c) An alternative 2-anonymity view. View (c) is more consistent than view (b).

To facilitate the following discussions on racing attributes, we introduce a measurement.

**Definition 12 (Inconsistency)** *Let inconsistency of attribute $i$ be $\text{inconsist}_i = (1 - \max_j(p_{ij}))$ where $p_{ij}$ is the fraction of values in domain level $j$ of attribute $i$ over all values in attribute $i$. Let the inconsistency of a data set be $\text{inconsist}_D = \max_i(p_i)$ where $1 \leq i \leq m$ where $m$ is the number of attributes in the quasi-identifier.*

Low inconsistency means that attribute values are mostly from one domain. High inconsistency indicates that attribute values are mixed from more than one domain. For example, inconsistency of the Gender attribute in Table 3(b) is 50% because four unknown values (*) are from domain level 1 and four values of male and female are from domain level 2. Inconsistency of the attribute Marriage in Table 3(b) is 50% too. As a result, inconsistency of Table 3(b) is 50%. Inconsistency of Table 3(c) is 0%.

An anonymity table is normally used for data mining or statistical analysis. Most data mining and statistical tools assume that values are drawn from the same domain of an attribute. When values are drawn from more than one domain, values from a more general domain do not provide the same detailed information as values from a more specific domain. There are two ways to handle the situation without changing data mining or statistical software tools. When the number of values from a more general domain is not too many, consider them as missing values and disregard them in the analysis process. When values from a more general domain are too many to be ignored, generalise other values in more specific domains to the more general domain to make the attribute consistent. In the latter solution, low distortion is sacrificed for high consistency.

We discuss three approaches for handling racing attributes and controlling inconsistency.

The first approach is to randomly select racing attributes to generalise. Consider a large data set where a small number of values are generalised. We wish that these generalised values, which may be considered missing values in an analysis process, are scattered across all attributes. The randomness of a small number of generalised values does not cause a big impact on any attribute, and therefore does not affect analysis results significantly.

The second approach is to set priority attributes. More often than less, attributes have different importance in data for an application. For example, the attribute Age is usually more important than the attribute Postcode in a medical data set. We may sacrifice postcode information for the integrity of age information as much as possible. Attributes to be sacrificed are set with high priority. High priority attributes receive low weights in calculating distortions while low priority attributes receive high weights. As a result, more generalisations will occur in high priority attributes than low priority attributes. This could reduce the overall inconsistency. For example, when we set attribute Marriage in Table 3(a) higher priority than attribute Gender, Table 3(a) will be generalised as Table 3(c), which has an inconsistency of 0%.

The third approach is to incorporate global recoding generalisation into local recoding generalisation. The inconsistency from the global recoding generalisation is always zero. However, the global recoding methods may over-generalise a table and cause high distortions. The strength and weakness of the local recoding generalisation complement those of the global recoding generalisation. Ideally, we wish that the consistency of a table is high and that a small number of more generalised values are scattered among attributes.

To make the inconsistency controllable, we introduce another requirement, the maximum inconsistency. We require that the inconsistency of a generalised table is smaller than *max_inconsist*.

We present the following metric to be a criterion for deciding when to choose global recoding generalisation.

**Definition 13 (Generalisation portion)** *Let values of an attribute be drawn from a number of domains* $< D_b, D_{b-1}, \ldots >$, *where* $D_b$ *is the most specific domain. The generalisation portion is defined as* $\mathrm{genportion} = 1 - P_{D_b}$ *where* $P_{D_b}$ *is the fraction of values in domain* $D_b$ *over all values of the attribute.*

Values in an attribute are split into base (the most specific) and generalisation portions. Note that the base portion is not necessarily from the most specific domain of an attribute hierarchy but the most specific one from domains which the attribute currently draws values from. For example, let the attribute Date-of-Birth be in domain levels $\{$day/month/year, month/year, year, 10year-interval, *$\}$. Assume that fractions of values drawn from each domain level are listed in the following: 0% from level day/month/year, 20% from level month/year, 40% from level year, 20% from 10year-interval, and 20% from *. The base domain is at domain level month/year since there are not values drawn from domain level day/month/year. As a result, $\mathrm{genportion} = 1 - 20\% = 80\%$.

We have the following relationship between generalisation portion and inconsistency.

**Lemma 3** *For an attribute, when the generalisation portion is less than 50%,* $\mathrm{inconsist} = \mathrm{genportion}$.

**Proof** When generalisation portion is less than 50%, the fraction of values drawn from the base domain is greater than 50%. As a result, the base domain has the largest fraction of values among all domains. Therefore, $\mathrm{inconsist} = 1 - P_{D_b} = \mathrm{genportion}$. $\square$

The relationship between generalisation portion and inconsistency is not this simple when generalisation portion is greater than 50%. In the previous example, Date-of-Birth values are drawn from the following four domain levels: 20% from level month/year, 40% from level year, 20% from 10year-interval and 20% from *. $\mathrm{genportion} = 1 - 20\% = 80\%$ whereas $\mathrm{inconsist} = 1 - 40\ = 60\%$.

In most applications, the required maximum inconsistency is less than 50%. Therefore, the requirement for the inconsistency of a generalised table to be less than max_inconsist is equivalent to the requirement that the generalisation portion is less than max_inconsist for every attribute.

The reason for using generalisation portion instead of inconsistency is that generalisation portion gives a desirable direction for generalisation. See the following two examples. Attribute 1: 90% of values are generalised to a more general domain, and 10% values remain at the original domain. We have $\mathrm{inconsist} = 10\%$ and $\mathrm{genportion} = 90\%$. Attribute 2: 10% of values are generalised to a more general domain, and 90% of values remain in the original domain. We have $\mathrm{inconsist} = 10\%$ and $\mathrm{genportion} = 10\%$. In the former case, 10% of of detailed information does not improve the quality of the attribute significantly, but reduces its utility. So we generalise the 10% of values to the more general domain for a 100% consistency. In the latter case, it is worthwhile to sacrifice 10% values to keep 90% detailed information. Therefore, we do not generalise the remaining values.

We use the generalisation portion as a criterion for switching on the global recoding. If the generalisation portion is larger than max_inconsist, we have to generalise values in the base domain (the current most specific one) to a more general domain. In other words, we need to use a global generalisation method to generalise an attribute until the portion of values to be generalised further is less than max_inconsist. The following lemma gives an indicator for this.

**Lemma 4** *Let $D$ be a table to be generalised into a $k$-anonymity view. Consider an attribute $i$ in the quasi-identifier and* $\mathrm{inconsist}_i = 0$. *Let $f_j$ be the frequency of value $j$ in the attribute. The lower bound of the generalisation portion is* $(\sum_{f_j<k} f_j)/|D|$.

**Proof** When the frequency of a distinct value in an attribute is less than $k$, this value will be generalised to satisfy the $k$ anonymity requirement. All such values are to be generalised. The number of generalised values is hence at least $\sum_{f_j<k} f_j$ since some other values may be involved in the generalisation. Therefore, the lower bound of generalisation portion is $(\sum_{f_j<k} f_j)/|D|$. $\square$

As a result, we can generalise an attribute globally and recursively until the lower bound of the generalisation portion is less than max_inconsist. Then the data set is ready for local recoding generalisation.

The objective of keeping low inconsistency contradicts the objective of minimising distortions. The maximum inconsistency gives users a means to achieve balance between minimising distortions and keeping the consistency of a generalised table.

## 7 Two local recoding anonymisation algorithms

After the distortion has been mapped to a proper distance metric, it is a natural way to achieve $k$ anonymisation by a clustering approach. An agglomerative hierarchical clustering method [12] suits $k$-anonymisation by local recoding generalisation very well. An agglomerative hierarchical clustering method works in the following way. Initially, each object is assigned as a cluster. Then two clusters with the smallest distance are merged into one cluster. This procedure repeats until the number of clusters reaches the user's specified number. We modify the agglomerative hierarchial clustering algorithm for $k$-anonymisation by local recoding.

One issue needs to be resolved when using a clustering algorithm for local recoding generalisation. One equivalence class is initially assigned as a cluster. In multidimensional local recoding generalisation, each equivalence class as a whole is to merge with another equivalence class to form a new equivalence class. In local recoding generalisation, only a portion of tuples in an equivalence class merge with another equivalence class. In other words, overlapping clusters are allowed and data points in the identical position are mapped into different clusters.

The purpose of allowing overlapping clusters is to preserve partial detailed information of a large equivalence class. For example, a small equivalence class (e.g. containing one tuple) is generalised with a large equivalence class (e.g. containing a hundred tuples). Should we generalise the whole large equivalence class in order to absorb the small equivalence class? We should not. A better solution is to allocate a small number of tuples, $k$-1 tuples, from the large equivalence class to generalise with the small equivalence class. As a result, information in most tuples of the large equivalence class is preserved. Data points representing tuples in the large equivalence class belong to two clusters, the one for the large equivalence class and the one merging with data points of the small equivalence class.

We propose two concepts, stub and trunk, to facilitate local recoding $k$-anonymisation by clustering.

**Definition 14 (Stub and Trunk of equivalence class)** *Suppose a small equivalence class $C_1$ and a large equivalence class $C_2$ are to be generalised for $k$-anonymity. If $|C_1| < k$ and $|C_1| + |C_2| \geq 2k$, $C_2$ is split into two parts, a* stub *and a* trunk. *The stub contains $(k - |C_1|)$ tuples, and the trunk contains $(|C_1| + |C_2| - k)$ tuples. The stub is to be generalised with the small equivalence class $C_1$.*

After this split, both the new generalised equivalence class and the remaining trunk of $C_2$ satisfy the $k$-anonymity property. The detailed information in the trunk is preserved.

We modify the distance calculation between two equivalence classes $C_1$ and $C_2$, where $|C_1| < k$, in the following, to support stub and trunk splitting.

- if $(|C_1| + |C_2| < 2k)$, calculate the distance between $C_1$ and $C_2$.
- if $(|C_1| + |C_2| \geq 2k)$, calculate the distance between $C_1$ and the stub of $C_2$.

We present two algorithms. The first algorithm does not have the maximum inconsistency constraint, and the second algorithm does.

The pseudo-code of the first algorithm is presented in Algorithm 1. In the algorithm, we say that an equivalence class $C$ is generalised with another equivalence class $C'$. We mean that $C$ is generalised with the stub of equivalence class $C'$ when $(|C| + |C'| \geq 2k)$ and $C$ is generalised with $C'$ when $(|C| + |C'| < 2k)$.

**Algorithm 1** $K$-Anonymization by Clustering in Attribute hierarchies (KACA1)

```
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
   ($^*$ simplified statements. Read explanation for details.)
```

Line 1 forms equivalence classes. Sorting data will speed up the process. One tuple is also called an equivalence class. The generalisation process continues in lines 2-6 when there is one or are 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. Note that the distance of $C$ and another equivalence class $C'$ means the distance of $C$ and the stub of $C'$ when $(|C|+|C'| \geq 2k)$. Line 5 finds the equivalence class $C'$ with the smallest distance to $C$. When there are more than one such equivalence classes, we select one randomly. Line 6 generalises the equivalence classes $C$ and $C'$.

This implies that $C$ is generalised with the stub of $C'$ if $(|C| + |C'| \geq 2k)$. When $C$ is generalised with the stub of $C'$, the trunk of $C'$ remains as an equivalence in the next round. This means that a large equivalence class can be split into a number of generalised equivalence classes. The algorithm terminates when there is no equivalence class whose size is smaller than $k$ left.

The complexity of KACA1 is analysed in the following. Let $n$ be the number of tuples. All tuples are sorted and only $O(n)$ passes are needed to find all equivalence classes. The complexity of this step is $O(nlog\, n)$. Let $|E|$ be the number of all equivalence classes, and $|E_s|$ be the number of equivalence classes whose size is less than $k$. Each iteration chooses an arbitrary equivalence class, which takes $O(1)$ time, evaluates the pairwise distance, which takes $O(|E|)$ time, finds the equivalence class with the smallest distance, which takes $O(|E|)$ time, and finally generalises the equivalence class, which takes $O(1)$ time. As there are $O(|E_s|)$ iterations, the overall runtime is $O(nlog\, n + |E| * |E_s|)$.

We present another algorithm that extends KACA1 by using the constraint of maximum inconsistency. The pseudo-code is listed in Algorithm 2.

**Algorithm 2** $K$-Anonymization by Clustering in Attribute hierarchies with the maximum inconsistency constraint (KACA2)

```
1: for each attribute in the quasi-identifier do
2:     generalise the attribute by the global recoding till the lower bound of genportion
       $<$ max_inconsistency according to Lemma 4
3: end for
4: call KACA1
5: for each attribute $i$ in the quasi-identifier do
6:     if inconsist_$i$ $>$ max_inconsistency then
7:         generate the attribute till inconsist_$i$ $<$ max_inconsistency
8:     end if
9: end for
```

The maximum inconsistency constraint is used in KACA2 to balance consistency and distortion. KACA2 incorporates the global recoding generalisation to reduce inconsistency. In line 2, the employment of global recoding generalisation is determined by the lower bound of the generalisation portion from Lemma 4. KACA1 is called after the lower bound of the generalisation portion is less than max_inconsist for each attribute. After local recoding generalisation by calling KACA1, a final generalisation step is conducted when necessary to ensure the inconsistency of each attribute is less than the user specified threshold. In this step, low distortion is sacrificed for high consistency.

The complexity of KACA2 has a similar formulation as that of KACA1. Global recoding generalisation takes $O(n)$ for each generalisation. The number of global generalisations has a lower bound of 0 and an upper bound of $m * (h - 1)$ where the $m$ is the number of attributes in the quasi-identifier and $h$ is the maximum height of attribute hierarchies. In practice, some attributes do not need global generalisation to satisfy Lemma 4 and some attributes only need one or two global generalisations. We estimate the complexity of this step as $O(m * n)$. In sum, the time complexity of KACA2 is $O(nlog\, n + m*n + |E| * |E_s|)$. The additional computational cost for global generalization $O(m*n)$ can be well compensated for by the reduction in $O(|E| * |E_s|)$ since $|E_s|$, the number of equivalence classes whose size is less than $k$, is significantly reduced as a result of global generalisation.

## 8 Proof-of-concept experiments

Our proposed methods are compared with a typical global recoding method, Incognito [9], and a typical multidimensional recoding method, Multi [10]. Different methods are compared against four quality measures, distortion, discernability metric, normalised average equivalence class size, and inconsistency. Multi assumes fully ordered attributes and does not use attribute hierarchical taxonomies, and hence we do not have distortion and inconsistency results for it.

The adult data set from the UCIrvine Machine Learning Repository [16] has become a benchmark data set for comparing $k$-anonymity methods. The data set has been used in most recent $k$-anonymity studies [6, 7, 9, 10, 11, 24]. We eliminated the

|  | 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 | Occupation | 14 | Taxonomy Tree | 2 |
| 6 | Race | 5 | Taxonomy Tree | 2 |
| 7 | Sex | 2 | Suppression | 1 |
| 8 | Native Country | 41 | Taxonomy Tree | 3 |

Table 4: Description of Adult Data Set

records with unknown values. The resulting data set contains 45,222 tuples. Eight attributes were used as the quasi-identifier, as shown as in Table 4.

Experimental results are shown in Figure 4 and Figure 5. In Figure 4, the first six attributes are selected as the quasi-identifier. In Figure 5, $k$ is fixed to 10. Since there is random selection in our algorithms, we report the distortion, discernability, normalised average equivalence class size, and inconsistency of our methods based on the average of ten trials. Our methods have been evaluated in both uniform and height weight schemes. Conclusions from both schemes are very similar and here we only show results from the height weight scheme. For the height weight scheme, $\beta = 1$. For the KACA2 algorithm, the maximum inconsistency is set as 10%.

Based on three quality measures, namely distortion, discernability, and normalised average equivalence class size, KACA1 performs consistently better than other methods. This shows that local recoding based on the proposed distance metric achieves good quality $k$-anonymity tables based on the three measures. However, its inconsistency is the highest. In some cases, the inconsistency of tables produced by KACA1 can be 70%. In such cases, values are drawn from every domain level in the Age attribute. This may cause difficulty in data mining applications.

Based on the inconsistency measure, Incognito performs best since its inconsistency is always zero. However, Incognito suppresses more than 50% of values (being generalised to the top) in some cases. Such generalised tables also cause difficulty in data mining applications. KACA2 balances distortion and consistency. Its incon-

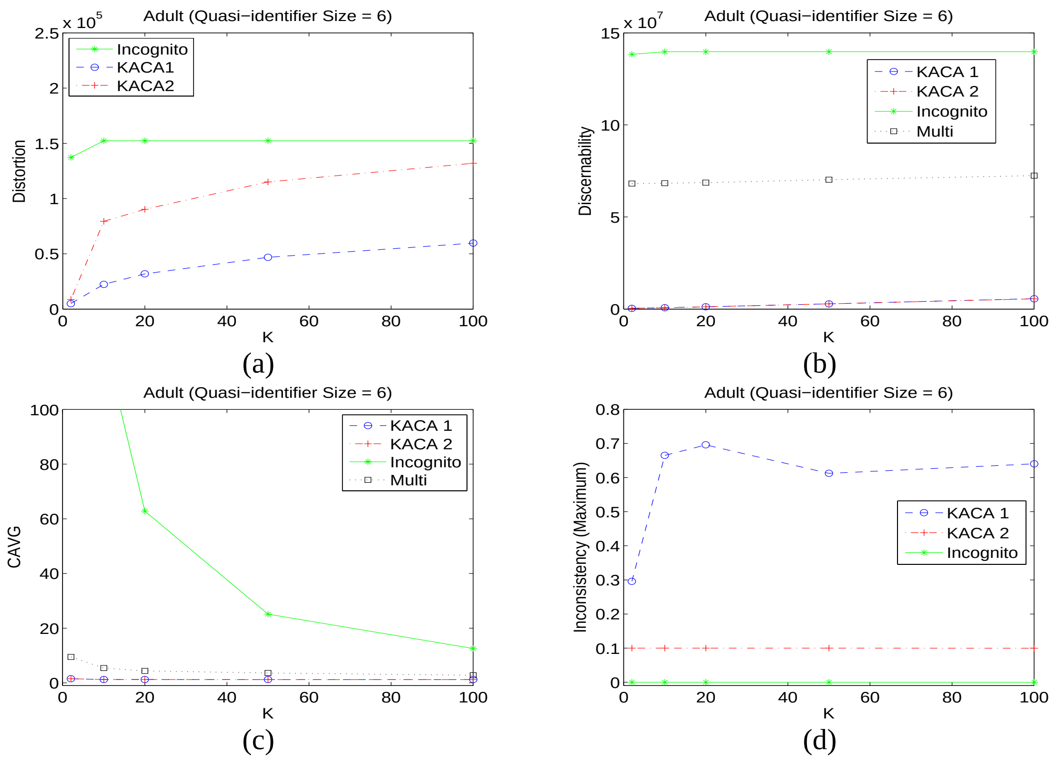


Figure 4: Performance of different methods with variant $k$ (a) Distortion (b) Discernablity (c) Normalised average equivalence class size (d) Inconsistency.

sistency is capped by 10%, and its distortion is in between the distortions of local and global recoding methods. We note that with the increase of $k$, inconsistency of KACA2 is closer to that of Incognito. This is because that larger $k$ requires larger equivalence classes. Based on Lemma 4, more attributes need global recoding when $k$ is larger. KACA2 balances distortion and inconsistency of local recoding and global recoding.

Based on discernability and normalised average equivalence class size measures, both KACA1 and KACA2 are better than Multi. Note that normalised average equivalence class sizes for the Multi in Figure 4b look flat. This is caused by the scale of Figure 4b. In comparison to big differences of normalised average equivalence size among different methods, differences of a method in variant $k$ are negligible. When we drew Multi results in a separate figure, it is consistent with Figure 10 in paper [10]. Fluctuations in Figure 5b are caused by different attributes in the quasi-identifier. A

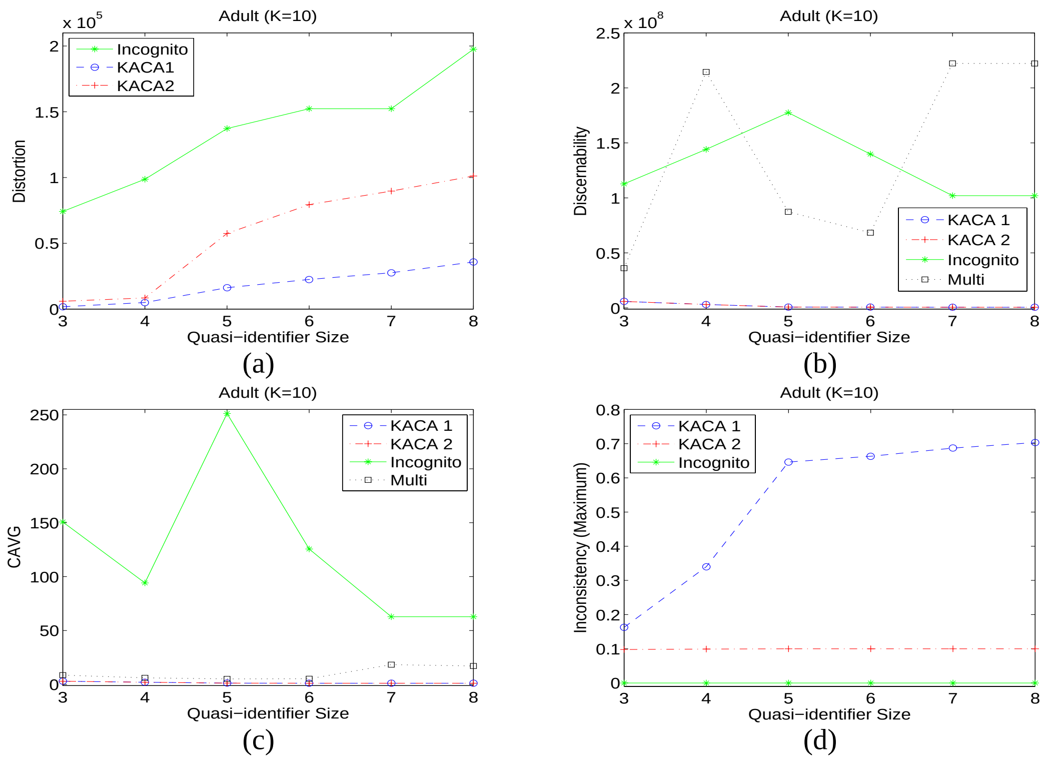


Figure 5: Performance of different methods with variant quasi-identifier size (a) Distortion (b) Discernablity (c) Normalised average equivalence class size (d) Inconsistency.

heuristic is used in the Multi algorithm [10] for choosing an attribute to partition the data space in the top down greedy algorithm. A new attribute leads to a new partition. Different partitions initiated from different attributes bear little similarities.

Both KACA1 and KACA2 are not as efficient as Incognito and Multi on the Adult data set. One computational intensive part of both algorithms is to compute the distances between equivalence classes to find the closest equivalence class pair. This time complexity is quadratic to the number of equivalence classes. The employment of an advanced indexing technique to keep track of closest equivalence classes for the efficient search of the closest equivalence class pair will improve the search efficiency significantly. This means that the current implementations of KACA1 and KACA2 are to be optimised for better efficiency.

## 9 Conclusions

In this paper, we study two major issues in local recoding $k$-anonymisation: measuring the distance of generalisation in data with attribute hierarchical taxonomies and handling the inconsistency of domains in the fields of a $k$-anonymity table. We define generalisation distances to characterise distortions of generalisations and discuss properties of the distance. We conclude that the generalisation distance satisfies properties of metric distances. We discuss how to handle a major problem in local recoding generalisation, inconsistent domains in a field of a generalised table, and propose a method to approach the problem. We show by experiments that the proposed local recoding method based on the distance metric achieves better quality $k$-anonymity tables by three quality measures than a typical global recoding method and a typical multi-dimensional recoding method, and that our inconsistency handling method balances distortion and consistency of a $k$-anonymity table well.

## Acknowledgements

Authors thank anonymous reviewers for their constructive suggestions. This research was supported by ARC discovery grant DP0774450 to Li, was supported in part by the RGC Earmarked Research Grant of HKSAR CUHK 4120/05E and 4118/06E to Fu, and a NSERC Discovery grant to Jian.

## References

[1] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Anonymizing tables. In *ICDT05: Proceedings of the 10th International Conference on Database Theory*, pages 246–258, Edinburgh, Scotland, 2005.

[2] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *PODS '01: Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 247–255, Santa Barbara, California, United States, 2001. ACM Press.

[3] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 19th ACM SIGMOD Conference on Management of Data*, pages 439–450, Dallas, Texas, May 2000. ACM Press.

[4] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE '05: Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*, pages 217–228, Tokyo, Japan, 2005. IEEE Computer Society.

[5] J. Domingo-Ferrer and V. Torra. Ordinal, continuous and heterogeneous k-anonymity through microaggregation. *Data Mining and Knowledge Discovery*, 11(2):195–212, 2005.

[6] Y. Du, T. Xia, Y. Tao, D. Zhang, and F. Zhu. On multidimensional k-anonymity with local recoding generalization. In *Proceedings of the 23rd International Conference on Data Engineering(ICDE2007)*, pages 1422–1424, Istanbul, Turkey, 2007.

[7] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *ICDE '05: Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*, pages 205–216, Tokyo, Japan, 2005. IEEE Computer Society.

[8] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *KDD '02: Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 279–288, Edmonton, Alberta, Canada, 2002. ACM Press.

[9] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In *SIGMOD '05: Proceedings of the 24th ACM SIGMOD international conference on Management of data*, pages 49–60, Baltimore, Maryland, 2005. ACM Press.

[10] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *ICDE '06: Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*, page 25, Washington, DC, USA, 2006. IEEE Computer Society.

[11] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Workload-aware anonymization. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 277–286, Philadelphia, PA, USA, 2006. ACM Press.

[12] K. Leonard and R. Peter. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience Publication, 1990.

[13] J. Li, R. C.-W. Wong, and A. W.-C. F. andf Jian Pei. Achieving *k*-anonymity by clustering in attribute hierarchical structures. In *Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery*, pages 405–416, Krakow, Poland, 2006.

[14] Y. Lindell and B. Pinkas. Privacy preserving data mining. *Journal of Cryptology*, 15(3):177–206, 2002.

[15] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *PODS '04: Proceedings of the 23rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 223–228, Paris, France, 2004. ACM Press.

[16] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI repository of machine learning databases, `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 1998.

[17] 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, Hong Kong, China, 2002. VLDB Endowment.

[18] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[19] 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.

[20] L. Sweeney. k-anonymity: a model for protecting privacy. *International journal on uncertainty, Fuzziness and knowldege based systems*, 10(5):557 – 570, 2002.

[21] J. Vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *KDD '03: Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 206–215, Washington, D.C., 2003. ACM Press.

[22] K. Wang, P. S. Yu, and S. Chakraborty. Bottom-up generalization: A data mining solution to privacy protection. In *ICDM '04: Proceedings of the 4th IEEE International Conference on Data Mining (ICDM'04)*, pages 249–256, Washington, DC, USA, 2004. IEEE Computer Society.

[23] R. Wright and Z. Yang. Privacy-preserving bayesian network structure computation on distributed heterogeneous data. In *KDD '04: Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 713–718, Seattle, WA, USA, 2004. ACM Press.

[24] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. W.-C. Fu. Utility-based anonymization using local recoding. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 785–790, Philadelphia, PA, USA, 2006. ACM Press.