DATA UTILITY AND PRIVACY PROTECTION IN DATA PUBLISHING

GRIGORIOS LOUKIDES

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN
COMPUTER SCIENCE

SCHOOL OF COMPUTER SCIENCE
CARDIFF UNIVERSITY
NOVEMBER 2008

DECLARATION

This work has not previously been accepted in substance for any degree and is not concurrently submitted in candidature for any degree.

Signed .... (candidate) Date 12/11/2008

STATEMENT 1

This thesis is being submitted in partial fulfillment of the requirements for the degree of PhD.

Signed .... (candidate) Date 12/11/2008

STATEMENT 2

This thesis is the result of my own independent work/investigation, except where otherwise stated. Other sources are acknowledged by explicit references.

Signed .... (candidate) Date 12/11/2008

STATEMENT 3

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed .... (candidate) Date 12/11/2008

Abstract

Data about individuals is being increasingly collected and disseminated for purposes such as business analysis and medical research. This has raised some privacy concerns. In response, a number of techniques have been proposed which attempt to transform data prior to its release so that sensitive information about the individuals contained within it is protected. k-Anonymisation is one such technique that has attracted much recent attention from the database research community.

k-Anonymisation works by transforming data in such a way that each record is made identical to at least $k - 1$ other records with respect to those attributes that are likely to be used to identify individuals. This helps prevent sensitive information associated with individuals from being disclosed, as each individual is represented by at least k records in the dataset. Ideally, a k-anonymised dataset should maximise both data utility and privacy protection, i.e. it should allow intended data analytic tasks to be carried out without loss of accuracy while preventing sensitive information disclosure, but these two notions are conflicting and only a trade-off between them can be achieved in practice. The existing works, however, focus on how either utility or protection requirement may be satisfied, which often result in anonymised data with an unnecessarily and/or unacceptably low level of utility or protection.

In this thesis, we study how to construct k-anonymous data that satisfies both data utility and privacy protection requirements. We propose new criteria to capture utility and protection requirements, and new algorithms that allow k-anonymisations with required utility/protection trade-off or guarantees to be generated. Our extensive experiments using both benchmarking and synthetic datasets show that our methods are efficient, can produce k-anonymised data with desired properties, and outperform the state of the art methods in retaining data utility and providing privacy protection.

Acknowledgements

I owe deep debt of gratitude to my supervisor, Dr. Jianhua Shao, for his guidance, enthusiasm and valuable criticism of my work. I am also thankful to Prof. Alex Gray, Prof. David Walker and Dr. Mikhaila Burgess for their constructive comments on my work, as well as to Prof. David Bell and Dr. Richard White for providing valuable feedback that has served to improve this thesis.

I would like to record my thanks to Tiancheng Li for providing the source code of the Incognito with *t*-closeness approach that I used in the experiments of this thesis, as well as to the anonymous reviewers of the conferences and journals for their constructive comments on my submitted papers.

Finally, on a personal note, I would like to thank my family and friends for their continuous support, and my girlfriend Jennifer for her love, understanding and encouragement.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Data about individuals is being increasingly collected, analysed and disseminated. Examples include the Electronic Patient Record [61] (a large database containing British patients' demographics, test results and billing information), the UK National DNA Database [9] (containing the genetic profile of more than 4 million citisens involved in criminal actions) and DoubleClick's database (containing more than 100 millions of customer profiles) [2]. The collected data can contribute significantly to studies such as medical research [30] and business intelligence [21].

Due to its sensitive nature, however, such data may compromise individuals' privacy, for example, as a result of accidental data loss (e.g. the recent HM Revenue and Customs case where over 25 million individuals' data was lost [5]), security infrastructure breaches [55, 107, 104, 67] (e.g. the UKvisa case where applicants' personal data was made visible on the web [7]) or adversarial data analysis and mining [15, 22, 106] (e.g. the case reported in [117] where a supermarket planned to use mined information about a customer's frequent alcohol purchases against him during trial). Such incidents have raised many privacy concerns. According to a recent study [120], 86% of internet users stated that their permission should be gained before their data was collected, while 94% of individuals would want privacy violators to be punished. Furthermore, there is evidence that individuals have attempted to protect privacy themselves by disabling cookies in their web-browsers, using fake personal data, or avoiding to disclose consumer preferences, detailed demographic or contact information [62, 70, 12].

1.1 Privacy and Its Protection

While privacy may be understood intuitively, it is a rather difficult term to define and analyse. Aristotle was the first to distinguish between "the public sphere of political activity and the private sphere associated with family and domestic life" [8], while Romans further developed these ideas and protected privacy legally [108]. Modern philosophers also agree that privacy is a meaningful and valuable concept. For example, Westin views privacy as "the claim of individuals, groups, or institutions to determine for themselves when, how and to what extent information is communicated to others" [119], while Gavison argues that the concept of privacy is related to limited accessibility to individuals, noting that privacy can be achieved by *solitude* (the lack of physical access to an individual), *secrecy* (the lack of information about an individual), and *anonymity* (the lack of attention to an individual) [52].

It is interesting to observe how many modern privacy-preserving technologies have been developed based on these views. WORM-based solutions[1] [134, 118], for example, ensure that private data cannot be altered, while encryption [60, 20] attempts to restrict access to it. Thus, both of these approaches may be seen as providing solitude. A popular system based on the WORM paradigm is Sun's StorageTek VolSafe [14], while PGP is a famous encryption-based program [50]. On the other hand, secrecy can be achieved based on cryptography [112, 129, 28, 93, 37] or access control models

[1] WORM (Write-Once Read-Many) is a paradigm according to which data can only be written once. WORM may be used to provide immutable versioning and secure logging.

[55, 107, 104, 102], which guarantee that unauthorised parties cannot interfere communication or access data. For example, cryptography is extensively used in systems ranging from satellite TV decoders to public key infrastructure systems (PKI) [72]. Finally, anonymous communication systems [101, 38, 122], such as the Anonymizer [1, 74] and data masking techniques [42, 22, 106, 110], such as data suppression [46], attempt to protect privacy through anonymity.

While preserving all these notions of privacy is important and necessary, this thesis focuses on a specific type of threat to anonymity called *individual identification*, which occurs when an attacker is able to infer the identity of an individual along with his or her sensitive information from data.

1.2 Preventing Individual Identification

Preventing individual identification has been a long-standing problem, and is required by some relevant legislation, such as the European Union Directive 95/46/EC [11] or the United States Healthcare Information Portability and Accountability Act (HIPAA) [4]. These laws enunciate the need for an "appropriate level of security" so that the published data does not provide "a reasonable basis to identify an individual". However, neither of these rules specify what form of identification should be prevented, how protection from identification should be achieved and what level of protection should be considered to be adequate. Thus, there is a large scope for considering how specific technologies may be developed for individual identification protection in practice.

There are two common scenarios under which individual identification may be examined [48, 110, 20]. The first involves an untrusted data collector/user, who attempts to identify individuals by combining collected data with other data [128, 15], bypassing security mechanisms [38, 87] or performing data analysis or mining [20]. A real-world case of individual identification under this setting is reported in [3], where Electronic Privacy Information Center, a U.S. public interest research center, alleged that a company called DoubleClick is "unlawfully tracking the online activities of Internet users (through the placement of cookies) and combining surfing records with detailed personal profiles contained in a national marketing database". Here, the goal of protecting privacy is to prevent the data collector from identifying individuals. The second scenario assumes a trusted data collector/publisher, who wishes to release data in a privacy preserving way, either for profit (e.g. selling customer data) or for benefiting the society at large (e.g. releasing demographic or medical data for research). In this case, the goal is to publish the data, but make it difficult for data users to identify individuals from the data. This thesis considers the second scenario.

Publishing data that prevents individual identification is not a straightforward task in the second scenario, since naive methods such as "data de-identification", i.e. removing information such as names or phone-numbers that may explicitly identify individuals, are not sufficient in practice. To demonstrate this, Sweeney [110] linked a voters list with some publicly available de-identified medical data containing "innocuous" information about patients, such as their date of birth, zip code and sex, and inferred medical information about William Weld, a former governor of the state of Massachusetts. It was also suggested that more than 87% of U.S. citisens can be identified as a result of data linkage [110]. To illustrate how this may happen, consider the data about individuals contained in Table 1.1, for example. Assume also that this data is to be released, but no one should be able to infer the salary of any individual. De-identification will simply remove individuals' names, releasing the data as shown in Table 1.3. However, if we assume that these individuals' names can be found on a possible external dataset (e.g. a voters list), such as the one depicted in Table 1.2, then by performing a simple join between Tables 1.2 and 1.3, one can reveal the salary of an individual pretty precisely.

Name	***Postcode***	***Salary (K)***
John	*NW10*	*10*
Mary	*NW10*	*10*
Alex	*NW11*	*10*
George	*NW12*	*10*
Mike	*NW28*	*37*
Anne	*NW30*	*40*
Tony	*NW30*	*39*
Helen	*NW30*	*38*

Table 1.1: Original data

Name	***Postcode***
John	*NW10*
Mary	*NW10*
Alex	*NW11*
George	*NW12*
Mike	*NW28*
Anne	*NW30*
Tony	*NW30*
Helen	*NW30*

Table 1.2: External dataset related to Table 1.1

Postcode	***Salary (K)***
NW10	*10*
NW10	*10*
NW11	*10*
NW12	*10*
NW28	*37*
NW30	*40*
NW30	*39*
NW30	*38*

Table 1.3: The result of de-identifying Table 1.1

1.3 k-Anonymisation and Research Challenges

Recognising the limitation of data de-identification, Sweeney proposed a technique called k-anonymisation [110]. To illustrate this technique, we assume that individuals' data (referred to as *microdata* [106]) is stored as a relational table T such that each tuple corresponds to one individual, and is comprised of three different types of attribute:

- *Identifiers (IDs)*: These attributes can be used to uniquely identify individuals. Examples include Social Security Numbers, UK National Insurance Numbers (NINs), and phone numbers.
- *Quasi-identifiers (QIDs)*: These attributes may be used in combination to identify a small group of individuals [32]. Examples include age and gender.
- *Sensitive attributes (SAs)*: These attributes contain sensitive values about individuals, for example individuals' income levels or medical conditions.

Informally, k-anonymisation is a process that attempts to convert T into T^* (possibly through some data transformation), such that each tuple in T^* is identical to at least $k-1$ other tuples with respect to QIDs. For example, Table 1.4 is a 4-anonymisation of Table 1.1, assuming that *Name* is an ID, *Postcode* a QID and *Salary* an SA. Here, a tuple having a value $NW[28-30]$ in *Postcode* may be associated with any of the 4 individuals having a *Postcode* between $NW28$ and $NW30$ in Table 1.1. As such, even if an external dataset exists, one will not know for certain which SA value in a k-anonymous dataset is associated with which specific individual, since this tuple corresponds to at least k individuals in the original dataset.

While forming groups of at least k tuples and transforming them to satisfy k-anonymity is not difficult, finding a "good" k-anonymisation is challenging [106]. First, there is a need to understand and define what makes a good k-anonymisation. Generally speaking, a good k-anonymisation should preserve data utility. Since transforming data to satisfy k-anonymity can excessively distort QID values, the resultant data may become practically useless. For instance, Table 1.4 may be considered as preserving data utility reasonably well, since postcode values are not "too" distorted, i.e. there are only a "few" possible postcode values covered by each anonymised value. Unfortunately, the notion of data utility is difficult to define [54], and finding a

Postcode	***Salary (K)***
NW[10-12]	*10*
NW[10-12]	*10*
NW[10-12]	*10*
NW[10-12]	*10*
NW[28-30]	*37*
NW[28-30]	*40*
NW[28-30]	*39*
NW[28-30]	*38*

Table 1.4: A 4-anonymisation of Table 1.1

Postcode	***Salary (K)***
NW[10-30]	*10*
NW[10-30]	*10*
NW[10-30]	*37*
NW[10-30]	*40*
NW[10-30]	*10*
NW[10-30]	*10*
NW[10-30]	*39*
NW[10-30]	*38*

Table 1.5: Another 4-anonymisation of Table 1.1

k-anonymisation with optimal data utility is both conceptually and computationally hard [91, 19, 127, 35].

Second, k-anonymised data should also make linking sensitive information to individuals difficult. Unfortunately, as first pointed out by Machanavajjhala et al. [85], k-anonymity alone may fail to prevent an attacker from inferring an individual's actual SA value. This may occur when an attacker can identify a k-anonymous group that contains this individual's tuple and all SA values contained in this group are identical. For example, knowing that *Mary* lives in an area with a postcode $NW10$, one can infer that the tuple representing Mary is contained in the first anonymous group of Table 1.4, and that Mary's salary is $10K$, since all the tuples contained in this group have the same salary value. Also, some sensitive information may be inferred through "ranges" [80, 71]. For instance, assume that an attacker knows that Tony lives in an area with a postcode $NW30$. Using Table 1.4, this attacker can infer that the tuple representing Tony is in the second anonymised group and that Tony's salary is in a small range $[37K - 40K]$. Thus, robust protection measures must be considered when k-anonymising data.

Third, generating an optimal k-anonymisation with respect to both data utility and privacy protection is computationally impossible, since they are conflicting requirements [85, 124]. Thus, a possible strategy for a data publisher (anonymiser) is to release k-anonymisations with a "good" trade-off between data utility and privacy protection. An example of such an anonymisation for Table 1.1 is shown in Table 1.5. Here, data utility may be preserved fairly well, since one may still use Table 1.5 to learn how many individuals live in an area with a postcode between $NW10$ and $NW30$, for example. Furthermore, one is not likely to infer an individual's exact salary or a small range in which this individual's salary lies using Table 1.5, since each tuple in this table is associated with at least 3 different salary values that form a relatively wide range. Allowing k-anonymisations to be generated with a desired utility/protection trade-off, or with a controlled level of utility and protection, is extremely important for applications, since data publishers and users often have diverse utility and protection requirements [51, 4, 99]. For example, when a pharmaceutical company uses data published by a healthcare trust, it may require the data to have a minimum level of utility with respect to a certain measure, e.g. classification accuracy, while the trust may require the anonymised data to have a minimum level of protection in order to ensure that it will not be possible for the company to infer sensitive information about patients included in the data. Developing techniques that allow such flexibility in k-anonymising data is a significant challenge.

Finally, datasets of different characteristics must also be considered. However, existing k-anonymisation methods do not work well with high-dimensional [16] or sparse [86] datasets, since it is difficult to find good groupings without incurring excessive information loss. Preventing individual identification in presence of data updates is also difficult, as an attacker may be able to identify individuals by observing k-anonymisations of incrementally updated tables [36, 125].

In this thesis, we address primarily the third challenge. Although it may seem surprising, despite the need for generating anonymised data that has a "good" level of data utility and protection in practice, existing k-anonymisation approaches do not consider how this may be achieved. They optimise either data utility or privacy protection [76, 77, 85, 80], and as a result the anonymisations they generate may not achieve a good balance between the two, or guarantee that a minimum level of data utility or privacy protection is achieved. Our aim is to develop k-anonymisation methods that can balance or guarantee these two properties.

1.4 Research Contributions

This thesis makes the following contributions:

- We propose new criteria to quantify data utility and protection that are founded on well-established notions of utility and protection. Our utility criteria are based on how original values may be reconstructed from generalised data [127, 64], and capture the accuracy of a number of tasks that are commonly performed on k-anonymised data, while our protection criteria consider the risk of private information disclosure through linking individuals to specific values or ranges of values. These criteria are applicable to both numerical and categorical attributes, and capture utility and protection uniformly. This allows them to be used for different types of dataset, and be employed as optimisation heuristics in our approach to generating k-anonymisations with required utility/privacy trade-off or guarantees.

- We develop an efficient algorithmic framework for k-anonymising data. Our method works by performing top-down data partitioning, followed by a greedy clustering guided by heuristics that allow both data utility and protection requirements to be incorporated into the anonymisation process. Important characteristics of our method include its effectiveness, its low time complexity, its ability to deal with data skewness, and its flexibility.

- We introduce two new k-anonymisation approaches. The Trade-Off Constrained (TOC) approach can generate k-anonymised data with an optimal trade-off between data utility and privacy protection, whereas the Utility and Protection Constrained (UPC) approach can guarantee that anonymised data will satisfy the required level of data utility and privacy protection. Both our approaches use optimisation heuristics based on our utility and protection criteria, and employ our algorithmic framework to generate k-anonymised data.

Extensive experiments using both benchmarking and synthetic data show that our approaches can generate k-anonymisations with good utility/protection trade-off or guarantees. More specifically, the data utility and privacy protection of the constructed anonymisations are comparable to those generated by the best-so-far methods, and the performance scales very well to large datasets.

1.5 Thesis Organisation

Chapter 2 reviews the related k-anonymisation literature focusing on data utility and privacy protection measures and algorithms. Furthermore, we briefly survey relevant techniques that attempt to provide protection for individual identification.

Chapter 3 discusses our new approach to k-anonymisation. We introduce our criteria for capturing data utility and privacy protection uniformly, and present our method that is capable of achieving k-anonymisations with a utility/protection trade-off.

In Chapter 4, we perform an extensive experimental study of the method described in Chapter 3 using both benchmarking and synthetic datasets. We evaluate the effectiveness of our method in terms of producing k-anonymisations with the required utility/protection trade-off, and compare the level of data utility and privacy protection offered by k-anonymisations generated by our method to some benchmark k-anonymisation methods [77, 35] using a number of alternative quality indicators [25, 77]. We also study the efficiency and scalability of our method.

In Chapter 5, we suggest further enhancement to the basic method presented in Chapter 3. In particular, we formally define the problem of range disclosure, introduce our Range Diversity criterion, and present our UPC approach to constructing anonymisations with a guaranteed amount of data utility and protection from range disclosure.

In Chapter 6, we empirically evaluate the enhancements proposed in Chapter 5 by comparing the enhanced method to three well-known mechanisms that are designed to optimise protection [85, 124, 80], our TOC method and benchmark algorithms [77, 35], using both benchmarking and synthetic datasets. We examine the quality of generated anonymisations using a number of utility and protection criteria, as well as the runtime performance.

Finally, we conclude the thesis and discuss directions for future work in Chapter 7.

Chapter 2

Background

In this chapter, we present an overview of the k-anonymisation literature and discuss relevant privacy preserving techniques that can be used to guard against individual identification.

2.1 k-Anonymisation: Concepts and Techniques

Informally, k-anonymisation is a process to find groups having a minimum of k tuples, and then to transform the QID values in each group so that they become indistinguishable. Formally, k-anonymisation is explained in Definition 2.1.1.

Definition 2.1.1 (k-Anonymisation). *k-Anonymisation is the process in which a table $T(a_1, \ldots, a_d)$, where $a_i, i = 1, \ldots, m$ are quasi-identifiers (QIDs) and $a_i, i = (m+1), \ldots, d$ are sensitive attributes (SAs), is partitioned into groups $\{g_1, \ldots, g_h\}$ s.t. $|g_j| \geq k, 1 \leq j \leq h$, where $|g_j|$ denotes the size of g_j, and tuples in each g_j are made identical w.r.t. QIDs.*

While the concept of k-anonymity is simple, what constitutes a "good" k-anonymisation and how such an anonymisation may be derived turn out to be not so easy to answer. In the following sections, we will review the existing works on k-anonymisation, analysing how they have attempted to address these two questions. We classify existing works on k-anonymisation using the scheme illustrated in Figure 2.1.

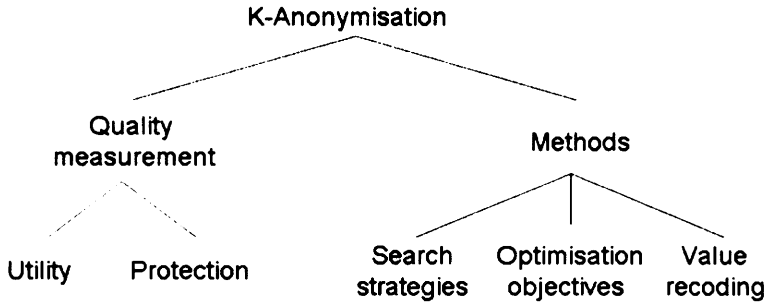


Figure 2.1: The problem space of k-anonymisation

Quality measurement is concerned with the metrics that can capture "how good" a k-anonymisation is in terms of data utility and privacy protection. We discuss data utility and protection measures in Sections 2.1.1 and 2.1.2 respectively. k-Anonymisation methods attempt to produce high quality anonymisations by performing data grouping and value recoding (to form groups of at least k tuples and to make QID values in each group identical). We review methods for k-anonymising data in Section 2.1.3. examining specifically strategies for forming groups. objectives to achieve during group formation. and ways in which QID values are recoded to satisfy k-anonymity.

2.1.1 Utility Criteria

One of the key requirements for k-anonymised data is that it should remain useful when used in applications [106]. For example, Table 2.2 is a valid 4-anonymisation of Table 2.1, which is derived by replacing all values in *Postcode* by a most generalised symbol "*". Clearly, the amount of information loss or data distortion is

Id	*Postcode*	*Salary (K)*
t_1	NW10	*10*
t_2	NW15	*10*
t_3	NW12	*10*
t_4	NW13	*10*
t_5	NW20	*20*
t_6	NW30	*40*
t_7	NW30	*40*
t_8	NW25	*30*

Table 2.1: Original data

Id	*Postcode*	*Salary (K)*
t_1	*	*10*
t_2	*	*10*
t_3	*	*10*
t_4	*	*10*
t_5	*	*20*
t_6	*	*40*
t_7	*	*40*
t_8	*	*30*

Table 2.2: A 4-anonymisation of Table 2.1

Id	*Postcode*	*Salary (K)*
t_1	NW[10-15]	*10*
t_2	NW[10-15]	*10*
t_3	NW[10-15]	*10*
t_4	NW[10-15]	*10*
t_5	NW[20-30]	*20*
t_6	NW[20-30]	*40*
t_7	NW[20-30]	*40*
t_8	NW[20-30]	*30*

Table 2.3: Another 4-anonymisation of Table 2.1

severe in this case, making the anonymised data practically useless. In contrast, the 4-anonymisation shown in Table 2.3 for the same data is intuitively better from the utility point of view. To capture the amount of information distortion in k-anonymised data, a number of criteria have been proposed.

Criteria based on information loss

One way to capture utility is by measuring the level of information loss incurred in k-anonymisation. Proposed measures can be classified into three groups according to how they capture information loss.

First, a larger group may incur more information loss, as more tuples are made indistinguishable after anonymisation. Based on this observation, Bayardo et al. [25] proposed the Discernability Metric (DM), and LeFevre et al. [77] suggested the Normalised Average Equivalence Class Size Metric (C_{AVG}), respectively.

Definition 2.1.2 (Discernability Metric (DM)). *Assume that a table T comprised of m QIDs $\{a_1, \ldots, a_m\}$ is partitioned into groups $\{g_1, g_2, \ldots, g_h\}$, such that $|g_j| \geq k, 1 \leq j \leq r$, and $|g_j| < k, r < j \leq h$. Tuples in each $g_j, 1 \leq j \leq r$, will have the same values in each QID after anonymisation, and tuples in each $g_j, r < j \leq h$, will not appear in the anonymised result. The DM of T under this partitioning is defined as*

$$DM = \sum_{j=1}^{r}(|g_j|^2) + \sum_{j=r+1}^{h}(|T| \times |g_j|)$$

where $|g_j|$ and $|T|$ denote the size of g_j and T respectively.

According to Definition 2.1.2, each tuple is penalised by the size of the group to which it belongs, while every tuple that belongs to a group containing less than k tuples is assigned a penalty of $|T|$.

Definition 2.1.3 (Normalised Average Equivalence Class Size Metric (C_{AVG})). *Assume that a table T comprised of m QIDs $\{a_1, \ldots, a_m\}$ is partitioned into groups*

$\{g_1, g_2, \ldots, g_h\}$, *such that* $|g_j| \geq k, 1 \leq j \leq h$, *and tuples in each* g_i *will have the same values in each QID after anonymisation. The Normalised Average Equivalence Class Size Metric* C_{AVG} *of* T *under this partitioning is defined as*

$$C_{AVG} = \frac{|T|}{h \times k}$$

where $|T|$ *denotes the size of table* T.

Both DM and C_{AVG} do not consider how data is transformed when measuring information loss. So, according to these measures, Tables 2.2 and 2.3 are considered to be equally useful. Intuitively, however, Table 2.3 retains significantly more information than Table 2.2, since it contains more detailed postcode values. To capture this, a number of measures have been proposed to quantify information loss by taking into account the way original QID values are transformed to anonymise data. As will be elaborated later in the section, original QID values are often replaced by more general ones to achieve k-anonymity. For example, the value $NW10$ in t_1 in Table 2.1 is replaced by $NW[10-15]$ when this table is anonymised to Table 2.3. Furthermore, data transformations are often conducted with respect to Domain Generalisation Hierarchies (DGHs) [106, 110, 19, 127]. For example, a possible DGH for *Postcode* is given in Figure 2.2.

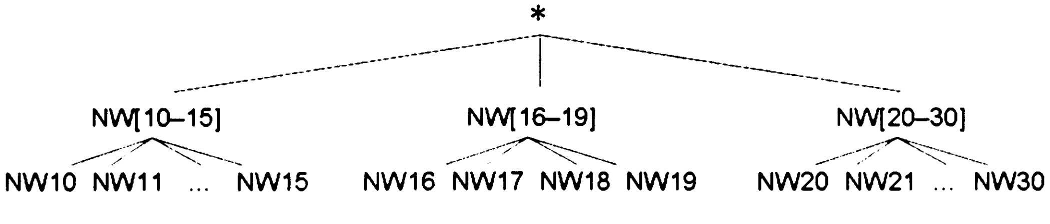


Figure 2.2: A DGH for *Postcode* in Table 2.3

Aggarwal et al. [19] measured information loss by considering DGHs. The closer a generalised value in the hierarchy to the leaves of a DGH is, the less information loss is considered to have been incurred. For example, the values $NW[10-15]$ and $NW[20-30]$ have a lower height than the value $*$ does in the DGH given in Figure 2.2, so Table 2.3 is considered to be more useful than Table 2.2 according to this principle. Based on this idea, a measure called Generalisation Cost (GENC) was proposed in [19]. A reformulation of this measure is illustrated in Definition 2.1.4.

Definition 2.1.4 (Generalisation Cost (GENC)). *Assume that a table T comprised of m QIDs $\{a_1, \ldots, a_m\}$ is partitioned into groups $\{g_1, g_2, \ldots, g_h\}$, such that $|g_j| \geq k, 1 \leq j \leq h$, and tuples in each g_j will have the same values in each QID after anonymisation. Let $\pi_{a_i}(g_j)$ denote the projection of g_j on a categorical QID attribute a_i, $i = 1, ..., m$, $u(\pi_{a_i}(g_j))$ be the generalised value of $\pi_{a_i}(g_j)$, and $f : u(\pi_{a_i}(g_j)) \rightarrow \mathbb{N}$ be a function that returns the level to which $u(\pi_{a_i}(g_j))$ lies at H_{a_i}, the DGH of a_i. The Generalisation Cost (GENC) is defined as*

$$GENC = \sum_{j=1}^{h} \left(|g_j| \sum_{i=1}^{m} \frac{f(u(\pi_{a_i}(g_j)))}{|H_{a_i}|} \right)$$

where $|H_{a_i}|$ is the height of H_{a_i}.

Although the GENC measure considers how values are transformed when capturing information loss, it requires DGHs to work. Recent research [96, 25] has shown that the use of DGHs may cause excessive value distortion for numerical QIDs, so this measure is only appropriate for handling categorical attributes in practice.

To overcome this limitation, a number of works [127, 64, 96, 124] proposed to capture information loss based on the probability of reconstructing an original value from anonymised data. For instance, given the first group $\{t_1, ..., t_4\}$ in Table 2.3, we can infer (based on the principle of indifference [65]) that the original postcode value for any individual represented in this group is a value in $\{NW10, NW11, ..., NW15\}$ with an equal probability of $\frac{1}{6}$. Generally, the probability of reconstructing a value in a QID a_i using a k-anonymous group g_j is $\frac{1}{r(\pi_{a_i}(g_j))+1}$, where $\pi_{a_i}(g_j)$ denotes the projection of tuples in g_j on a_i, and $r(\pi_{a_i}(g_j))$ gives the length of the range of $\pi_{a_i}(g_j)$ when a_i is numerical or the number of distinct values in $\pi_{a_i}(g_j)$ when a_i is categorical. Intuitively, larger ranges incur more information loss, since we are less certain about original values. Note that this is fundamentally different from measuring information loss using the height of DGHs. That is because QID values having the same height in a DGH do not necessarily have the same ranges. For instance, $NW[10-15]$ is narrower than $NW[20-30]$ in the DGH shown in Figure 2.2, implying that the first group in Table 2.3 incurs less information loss than the second group $\{t_5, ..., t_8\}$. Based on this principle, a measure called Normalised Certainty Penalty (NCP) was proposed in [127]. A reformulation of the NCP measure is illustrated in Definition 2.1.5.

Definition 2.1.5 (Normalised Certainty Penalty (NCP)). *Assume that a table T comprised of m QIDs $\{a_1, \ldots, a_m\}$ is partitioned into groups $\{g_1, g_2, \ldots, g_h\}$, such that $|g_j| \geq k, 1 \leq j \leq h$, and tuples in each g_j will have the same values in each QID after anonymisation. Let $r : V \longrightarrow \mathbb{N}$ be a function that, given a set of values V, returns the length of the range of values (number of distinct values) in V. The Normalised Certainty Penalty measure (NCP) is defined as*

$$NCP = \sum_{j=1}^{h} \left(|g_j| \sum_{i=1}^{m} \frac{r(\pi_{a_i}(g_j))}{r(\pi_{a_i}(T))} \right)$$

where $\pi_{a_i}(g_j)$ and $\pi_{a_i}(T)$ denote the projection of g_j and T on a QID attribute a_i respectively.

Several utility metrics that are conceptually similar to NCP, such as the Loss Metric [64], the Ambiguity Metric (AM) [96] and the Information Loss Metric [124] were also proposed. These measures use slightly different definitions for the range-mapping function r, and they are all applicable to both numerical and categorical attributes.

Criteria based on the accuracy of tasks

Another way to capture utility is based on measuring the accuracy of a specific task performed on anonymised data. Iyengar [64] observed that anonymisation can make it difficult to build an accurate classification model. To see this, assume that an anonymiser releases Table 2.4. A classifier is to be built using *Postcode* as a predictor attribute and *Mortgage* as the class attribute, where Y indicates that an individual has a mortgage and N does not. Building an accurate classification model using Table 2.4 is possible, as this table contains unperturbed information about classes. The result is illustrated in Table 2.5. Now assume that the anonymiser releases instead a 2-anonymised version of Table 2.4 as shown in Table 2.6, where *Age* and *Postcode* are QIDs, and *Salary* is an SA. In this case, it can be difficult for an accurate classifier to be built, as we can no longer be certain whether a class label of a tuple is Y or N due to anonymisation. For instance, the class label of a tuple in the first group of Table 2.6 can be either Y or N, depending on its original postcode value shown in Table 2.4.

Clearly, anonymised data helps building a classification model when it contains groups with a single class label. Thus, to capture data utility, Iyengar introduced the *Classification Metric* (CM) [64], which is expressed as the number of tuples whose class labels are different from that of the majority of tuples in their group, normalised by the table size.

Id	*Age*	*Postcode*	*Salary (K)*	*Mortgage*
t_1	15	NW10	10	Y
t_2	30	NW15	10	N
t_3	30	NW12	10	Y
t_4	15	NW13	10	N
t_5	40	NW20	20	Y
t_6	80	NW30	40	N
t_7	80	NW30	40	N
t_8	40	NW25	30	Y

Table 2.4: Table used for classification

Function	*Mortgage*
(10 $\leq$ postcode $<$ 13)$\vee$(20 $\leq$ postcode $\leq$ 25)	Y
(13 $\leq$ postcode $<$ 20)$\vee$(26 $\leq$ postcode $\leq$ 30)	N

Table 2.5: A classification derived from Table 2.4

Id	*Age*	*Postcode*	*Salary (K)*	*Mortgage*
t_1	15	NW[10-13]	10	Y
t_4	15	NW[10-13]	10	N
t_2	30	NW[12-15]	10	N
t_3	30	NW[12-15]	10	Y
t_5	[40-80]	NW[20-30]	20	Y
t_6	[40-80]	NW[20-30]	40	N
t_7	[40-80]	NW[20-30]	40	N
t_8	[40-80]	NW[20-30]	30	Y

Table 2.6: A 2-anonymisation of Table 2.4

Definition 2.1.6 (Classification Metric (CM)). *Assume that a table T comprised of m QIDs $\{a_1, \ldots, a_m\}$ is partitioned into groups $\{g_1, g_2, \ldots, g_h\}$, such that $|g_j| \geq k, 1 \leq j \leq h$, and tuples in each g_j will have the same values in each QID after anonymisation, and let $y : g_j \longrightarrow \mathbb{N}$ be a function that given a group g_j, computes the number of tuples that have a different class label from the majority of tuples in g_j. Classification Metric (CM) can be defined as*

$$CM = \frac{\sum_{i=1}^{h} y(g_j)}{|T|}$$

where $|T|$ denotes the size of T.

A large CM score indicates that the table may not support classification induction well, as most groups will have more than one class label. For example, Table 2.6 may not be considered as useful for classification, since the CM score for this table is $\frac{1+1+2}{8} = \frac{4}{8}$. On the other hand, Table 2.7 is considered to be more useful for classification, as its CM score is $\frac{0+0+0+0}{8} = 0$. While CM can capture the accuracy of classification, it may not indicate the level of information loss incurred when anonymising data, as it does not take into account the way QIDs are transformed. For example, Table 2.6 retained more information than Table 2.7 (the AM scores for Tables 2.6 and 2.7 are 0.164 and 0.252 respectively), despite having a higher (worse) CM score.

Id	Age	*Postcode*	*Salary (K)*	*Mortgage*
t_1	[15-30]	NW[10-12]	10	Y
t_3	[15-30]	NW[10-12]	10	Y
t_2	[15-30]	NW[13-15]	10	N
t_4	[15-30]	NW[13-15]	10	N
t_5	[40-80]	NW[20-25]	20	Y
t_8	[40-80]	NW[20-25]	30	Y
t_6	[40-80]	NW[30]	40	N
t_7	[40-80]	NW[30]	40	N

Table 2.7: A 2-anonymisation of Table 2.4 that helps classification

LeFevre et al. [77] considered measuring the utility of anonymised data when used for aggregate query answering. Consider for example, the following query on Table 2.3:

```
Q:  select COUNT(*)
    from table
    where Postcode = NW10
```

Clearly, only the tuples in the first group can be used to answer this query. However, since these tuples have a value of $NW[10-15]$, we cannot be sure about the exact number of tuples having a value $NW10$, so Q can only be answered through estimation [124]. On the other hand, using Table 2.1 we can produce an answer to Q precisely. By measuring the normalised difference between the answers to a query using the anonymised and original data, we may quantify data utility. Intuitively, a smaller difference implies that anonymised data is more useful.

Discussion

We classify the metrics discussed above as shown in Figure 2.3 and compare them using three criteria that are particularly important in applications [127, 96]: whether these metrics can handle both numerical and categorical QIDs, capture the accuracy of tasks performed using anonymised data, and be incorporated into anonymisation methods. We summarise the characteristics of existing measures with respect to our criteria in Table 2.8.

Ideally, a utility metric should be applicable to both numerical and categorical QIDs, since many datasets that contain mixed attributes need to be anonymised in practice [35, 124]. As can be seen from Table 2.8, the measures that are based on group size, reconstruction probability and aggregate query answering offer this flexibility. Furthermore, it is important for a measure to capture data utility in terms of performing tasks using anonymised data. Since anonymisers often do not know how anonymised data will be used, e.g. in cases where multiple data users have diverse

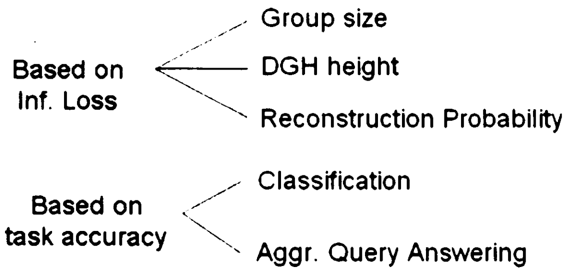


Figure 2.3: A classification of utility metrics

Reference	Utility criterion	Type of QID	Task	Optimisation
[25, 77]	Group size	any	any	no
[19]	DGH height	categorical	any	yes
[127, 64] [96, 124]	Reconstruction probability	any	any	yes
[64]	Classification	categorical	classification	yes
[77]	Aggregate Query Answering	any	query answering	yes

Table 2.8: Summary of existing utility metrics

data analysis requirements, the applicability of the task-based measures [64, 77], which assume one type of task, is limited. On the other hand, the measures that are based on reconstruction probability can capture the accuracy of many data mining tasks, such as classification, parameter estimation, hypothesis testing and regression [69]. Finally, besides being used for evaluating the quality of anonymised data, utility measures should also be easy to be incorporated into k-anonymisation algorithms as heuristics, so that k-anonymisations with intended utility quality may be produced. With the exception of those based on group size, all utility measures have been used as optimisation criteria in k-anonymisation methods [78, 127, 35, 96, 64, 124].

From the above analysis, we conclude that measures based on reconstruction probability are most useful in practice, while group size measures should only be used to roughly indicate data utility. We also see the need for further research in developing task-based measures for a wide range of tasks beyond classification and aggregate query answering.

2.1.2 Protection Criteria

As discussed in Introduction, individuals can still be identified when de-identified data is linked to publicly available data (illustrated by link (1) in Figure 2.4). k-Anonymisation attempts to weaken link (1) by grouping tuples together and making QID values in each group identical. Teng et al. [111] proposed several measures to quantify the strength of this link based on how "easily" an attacker can infer the original QID value of an individual. However, making the inference of a QID value hard does not completely eliminate the possibility of identifying sensitive information about individuals. For instance, when there is a strong correlation between anonymised QIDs and SAs (link (2) in Figure 2.4), an attacker may infer the sensitive values of individuals using additional background information about the QID values [85].

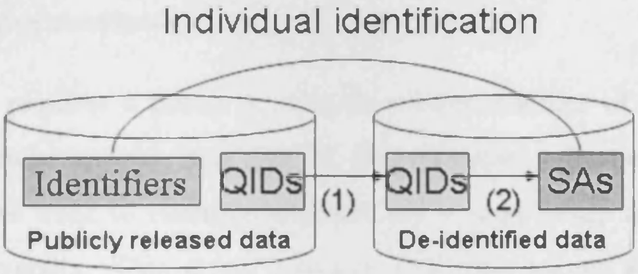


Figure 2.4: Individual identification through data linkage

We refer to this problem as *value disclosure*. Consider Table 2.3 again, for example. Knowing that an individual lives in an area with $Postcode = NW10$, one can infer that this individual's salary is $10K$, as all tuples with a postcode value between

$NW10$ and $NW15$ have a salary value of $10K$. On the other hand, even when one knows that an individual lives in an area with *Postcode* = $NW30$, he/she cannot be certain about this individual's salary, as any of the three distinct salary values, which appears in the second anonymised group of Table 2.3, is possible.

A principle against value disclosure called l-diversity was proposed in [85], requiring each anonymised group to contain at least l "well represented" values in an SA attribute. There are several instantiations of l-diversity [85], the most flexible of which is given in Definition 2.1.7.

Definition 2.1.7 (Recursive (c, l)-diversity). *Assume that a table T comprised of m QIDs $\{a_1, \ldots, a_m\}$ is partitioned into groups $\{g_1, g_2, \ldots, g_h\}$, such that $|g_j| \geq k, 1 \leq j \leq h$, and tuples in g_j will have the same values in each QID after anonymisation, and sa is an SA. T is (c, l)-diverse when*

$$r_1 < c \times (r_l + r_{l+1} + \ldots + r_n)$$

holds for every g_j, where $r_i, i = 1, ..., n$ is the number of times the i-th frequent SA value appears in g_j, n is the total number of distinct SA values in g_j, and c, l are anonymiser-specified constants.

This measure requires a group to contain a large number of distinct SA values, none of which should appear "too" often. For example, assume that both c and l are set to 2 and we want to examine whether the second group of Table 2.3 satisfies recursive $(2, 2)$-diversity. This group contains 3 distinct values whose frequencies in descending order are $r_1 = 2, r_2 = 1$ and $r_3 = 1$ (r_1 corresponds to the most frequent value $40K$, r_2 to the second most frequent value $30K$ and r_3 to the third most frequent value $20K$). Since l and c are equal to 2, we check whether $r_1 < c \times (r_2 + r_3)$ or equivalently whether $2 < 2 \times (1 + 1)$. As the inequality holds, the second group of Table 2.3 satisfies recursive $(2, 2)$-diversity. The idea behind Definition 2.1.7 is simple. To infer the actual SA value of an individual, an attacker has to associate an individual with a value in the group, and assuming that any tuple in a group can represent an individual equally likely [85, 124], he/she is more likely to succeed when a specific SA value appears many times in the group.

Other criteria that attempt to provide protection for value disclosure by enforcing different frequency-based constraints on SA values include (a, k)-anonymity [121] and p-sensitive-k-anonymity [113]. Both criteria require each group of SA values to have less than a specified number of distinct values. Chen et al. [39] recently proposed a value disclosure measure that takes into account some interesting types of background knowledge that an attacker may have when measuring protection. This includes knowledge about SA values that an individual I does not have, knowledge about SA values belonging to another individual, and knowledge that a group of individuals in which I is included has a certain SA value.

Li et al. [80] argued that to determine whether a k-anonymised group poses a risk of disclosing sensitive information, the distribution of data in the whole dataset should be taken into account. Suppose, for example, that we have a dataset in which 60% of tuples have the value *Flu* in an SA *Disease*, and we form a k-anonymous group G_1 which also has 60% of its disease values as *Flu*. Then, although an attacker can infer that an individual in G_1 suffers *Flu* with relatively high confidence (i.e. 60%), Li et al. [80] consider that G_1 is well protected, since this fact can be inferred from the dataset itself. On the other hand, a k-anonymous group G_2 having 80% of its disease values as *Flu* is considered to be at risk of disclosing sensitive information, as it reveals some additional information that is not available from the dataset. Here, the privacy leak is caused by some semantic inference on data distribution. This is in contrast to value disclosure measures, which consider the frequency of values occurring in a group only. We refer to this type of disclosure as *semantic disclosure*. To capture these semantic inferences, Li et al. [80] proposed to measure the level of semantic similarity between the SA values in a group. This is done using a measure called t-closeness that is based on the distance between the probability distribution of the SA values in an anonymised group and that of SA values in the whole dataset. The smaller the distance the higher the level of protection achieved. For instance, G_1 is considered to be better protected than G_2 according to this measure.

Some privacy criteria attempted to prevent sensitive information to be given away in the form of ranges. Consider for example a k-anonymous group comprised of three tuples containing the values $10K$, $11K$ and $12K$ in a numerical SA *Salary*. As discussed in Introduction, knowing that an individual's tuple is contained in this group, an attacker can learn that this individual's salary is in the range of $[10K-12K]$. Clearly, when this range is small, the disclosure may be considered as sensitive. We call this problem *range disclosure*. A number of works proposed to capture the level of protection against range disclosure offered by anonymised data, using the maximum range [71] or the variance [78] of SA values in a group. Intuitively, groups with small SA ranges (e.g. containing salary values in $[10K - 12K]$) may allow sensitive information to be inferred.

Xiao and Tao [124] suggested that range disclosure may be prevented by generalising SA values. Although they do not require k-anonymity to be satisfied, as we will explain in Section 2.2.1, their protection criterion is applicable to k-anonymised data. For example, consider a group containing the values $20K$ and $29K$ in a categorical SA *Salary*. This makes it possible for an individual represented in this group to be associated with a small range $[20K - 29K]$. To protect this disclosure, the method proposed in [124] will generalise the salary values in this group with respect to the hierarchy shown in Figure 2.5, i.e. it will replace these values with a general value $[20K - 40K]$. Protection for range disclosure is measured by calculating the probability of associating an individual with a specific range using anonymised data. For example, the level of protection offered by anonymising the SA values in the group as described above is $\frac{1}{2}$, since the probability of a tuple having a generalised value $[20K - 40K]$ to actually have a value of $20K$ or $29K$ is $\frac{1}{2}$ according to the hierarchy illustrated in Figure 2.5 [1].

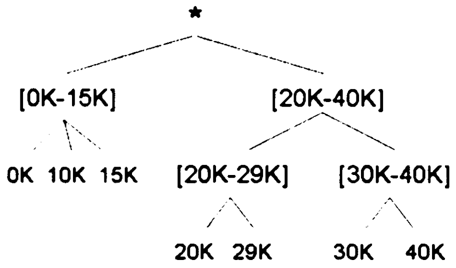


Figure 2.5: A hierarchy for *Salary* in Table 2.3

Discussion

We classify the protection criteria discussed above as illustrated in Figure 2.6 and examine two important properties that these criteria should satisfy in order to be useful in applications: whether they can handle both numerical and categorical QIDs and whether they can capture protection for different types of disclosure. We summarise the characteristic of existing measures w.r.t. these criteria in Table 2.9.

As can be seen, with the exception of t-closeness [80], all protection measures are applicable to either numerical or categorical SAs, and thus cannot handle datasets having mixed SA attributes.

Ideally, a measure should be able to offer protection from all types of attack, since it is not practical to anticipate what kind of knowledge an attacker may have. However, with the exception of the measure proposed in [124], all measures can only provide protection for one specific type of disclosure. In what follows, we explain why

[1] This probability is computed as the fraction between the number of values that belong to the subtree rooted at $[20K - 29K]$ and the number of values that belong to the subtree rooted at $[20K - 40K]$ in the hierarchy illustrated in Figure 2.5

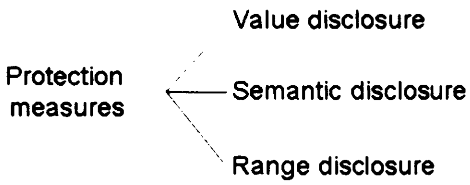


Figure 2.6: A classification of protection metrics

Reference	Type of disclosure	Type of SAs
[85, 121, 113, 39, 124]	Value disclosure	categorical
[80]	Semantic disclosure	any
[71, 78]	Range disclosure	numerical
[124]	Range disclosure	categorical

Table 2.9: Summary of existing protection metrics

existing measures fail to guard against different types of disclosure from the one they assume.

We first consider existing measures that are designed to guard against value disclosure. As discussed above, these measures may not provide protection against semantic disclosure, as they do not take semantic information associated with SA values into account. Furthermore, value disclosure-based measures may not provide protection for range disclosure, as SA values in a group may form a small range after anonymisation even though they are all distinct.

We also observe that preventing semantic disclosure using t-closeness [80] may not provide protection against value disclosure. For instance, consider a dataset comprised of one million tuples having two values $\{Flu, HIV\}$ in an SA *Disease*, and that we have 30% of individuals suffering HIV and 70% suffering Flu. Assume also that the goal is to prevent an attacker from inferring if an individual suffers HIV. As explained in Section 2.1.2, a group G_3 containing 30 individuals suffering HIV and 70 individuals suffering Flu offers optimal protection from semantic disclosure, as the data distribution of SA values in this group is the same as the one in the whole dataset. Now consider another group G_4 containing 10 individuals suffering HIV and 90 individuals suffering Flu. The data distribution of the SA values in G_4 is different from the one in the whole dataset, and thus G_4 is considered to offer worse protection for semantic disclosure than G_3. However, G_4 provides better protection for value disclosure than G_3, as the probability of inferring that an individual suffers HIV using G_4 is lower than that when G_3 is used. Similarly, it can also be shown that the measure proposed in [80] does not prevent range disclosure, as "mirroring" the distribution of SA values in a dataset that allows range disclosure will result in creating badly protected groups. While Li et al. [80] claimed that such inferences could be allowed as they can be made from using the entire dataset anyway, there are applications in the medical or financial domain where protection against these inferences should be guaranteed in any case [124].

Furthermore, we note that providing protection for range disclosure using the measures proposed in [71, 78] may not eliminate the value disclosure problem. This is because, maximising the range [71] or variance [78] of SA values in a group, may not lower the probability of inferring an SA value, as the frequency of this value occurring in the group is not limited. For example, consider a 3-anonymous group that contains 1 value of $0K$ and 2 values of $40K$ in an SA *Salary*. These values lie in a large range $[0K - 40K]$ and have a large variance of 355.55. However, they still allow an attacker to infer that an individual has a salary value $40K$ with a probability of $\frac{2}{3}$. In contrast, the method introduced in [124] can provide protection for both range and value disclosure, since an attacker may not observe the frequency of the actual SA value of an individual if SA values in a group are generalised. For instance, the generalised value $[20K - 40K]$ does not allow one to infer the number of individuals that have an actual salary value $20K$ or $40K$ in the group. However, by generalising SA values, the method proposed in [124] may increase the amount of information loss incurred when anonymising data. In addition, all the existing range disclosure-based measures cannot provide protection from semantic disclosure. This is because, the distribution of original SA values in the entire dataset is not taken into account when measuring protection, and thus semantic inferences using the distribution of SA values in anonymised groups may not be prevented.

So, with the exception of t-closeness, the existing protection measures are still rather limited in terms of their applicability in practice, since they cannot handle both types of SA attribute. Furthermore, offering protection for more than one type of disclosure while allowing original SA values to be released is clearly useful, as using the approach proposed in [82] may incur excessive information loss as a result of generalising SA values.

2.1.3 k-Anonymisation Algorithms

To derive a k-anonymisation, an algorithm should form groups of at least k tuples in a way that optimises one or more of the criteria discussed in Sections 2.1.1 and 2.1.2, and recode QID values to make tuples in these groups identical. In the following, we review existing k-anonymisation techniques in terms of search strategies, optimisation objectives and value recoding models.

Search strategies

Finding an optimal k-anonymisation with respect to the criteria discussed in Section 2.1.1 is an NP-hard problem [91, 19, 127, 35]. Thus, many methods employ heuristic search strategies to form groups. Samarati [106] proposed a binary search on the height of DGHs, LeFevre et al. [76] suggested a search similar in principle to the Apriori [21] used in association rule mining, and Iyengar [64] used a genetic algorithm. Partitioning has also been used to form groups in k-anonymisation. LeFevre et al. [77, 78] examined several partitioning strategies including techniques originally proposed for kd-tree construction [47], while Iwuchukwu et al. [63] developed a set of heuristics inspired from R-tree construction [57]. One of the most popular data grouping strategies is clustering [43, 17, 18, 127, 96, 79]. The main objective of clustering-based k-anonymisation methods is not to derive "natural" groups or structures in data, as traditional clustering algorithms do [58, 132], but to form groups that optimise a particular objective criterion (e.g. the NCP measure illustrated in Definition 2.1.5 [127]). In order to do so, they perform greedy search constructing groups in a bottom-up [43, 17, 96, 79] or a top-down fashion [127]. Figure 2.7 summarises existing search strategies.

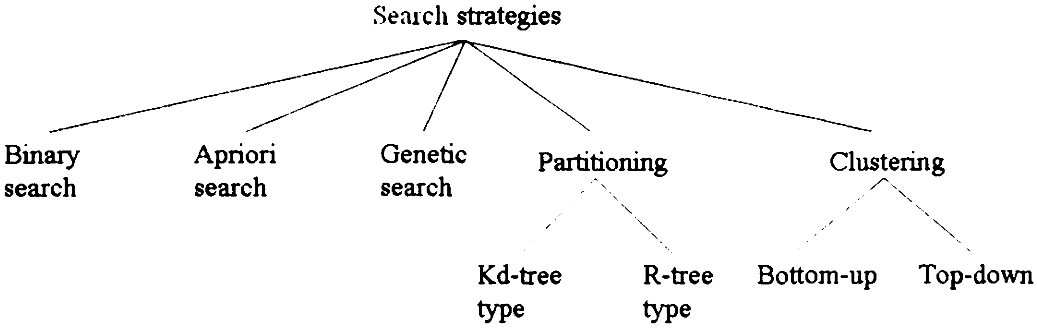


Figure 2.7: A classification of search strategies

Reference	Search strategy	Quality	Efficiency
[106]	Binary Search	very low	exponential
[76]	Apriori search	very low	exponential
[64]	Genetic search	very low	exponential
[77, 78]	Kd-tree type partitioning	low	log-linear
[63]	R-tree type partitioning	low	log-linear
[43, 17, 96, 79]	Bottom-up clustering	high	quadratic
[127]	Top-down clustering	high	quadratic

Table 2.10: Summary of existing search strategies

Using a certain search strategy to anonymise data may affect both the quality of the resultant anonymisation as well as the efficiency of producing it, as illustrated in Table 2.10. In particular, experimental studies [127, 77, 17] showed that binary, apriori and genetic-based methods are worse in terms of both effectiveness and efficiency compared to partitioning and clustering-based ones. This is because these methods explore a much smaller search space than partitioning and clustering-based methods, while their worst-case time complexity is exponential w.r.t. the cardinality of the dataset. Thus, more recent research has focused on developing methods that use partitioning or clustering-based search [127, 96, 77]. Furthermore, it was also shown in [127, 96] that partitioning-based methods tend to produce anonymisations of lower quality compared to those constructed by clustering-based methods, as they attempt to form groups by splitting data based on one attribute at a time, rather than considering all attributes together as clustering-based methods do. As a result, these methods are sensitive to the choice of splitting attribute and tend to perform poorly, particularly when the dataset is skewed [35, 96]. However, partitioning is faster than clustering by orders of magnitude, requiring $O(n \times log(n))$ time to derive an anonymisation, as opposed to $O(n^2)$ for clustering, where n is the cardinality of the dataset. Clearly, it is desirable to have methods that allow high-quality anonymisations to be derived efficiently.

Optimisation Objectives

Existing grouping strategies fall into three groups regarding their optimisation objectives. The goal of some of the earlier k-anonymisation methods is to achieve a maximum level of data utility subject to a minimum group size requirement expressed as k [76, 25, 77, 78, 127, 35]. We refer to these methods as *group-size constrained.* Gedik and Liu [53] noted that the level of information loss incurred during anonymisation should be bounded, so as to ensure that data remains useful for applications such as location privacy preserving services. We refer to this approach as *utility constrained.* However, as first pointed out by Machanavajjhala et al. [85], both group-size and utility constrained methods may result in an unacceptably low level of privacy protection. As a result, some *protection constrained* methods [85, 124, 80, 71] were proposed, introducing additional protection constraints that released data must satisfy. These constraints can be a minimum score in l-diversity or other protection criteria discussed earlier in Section 2.1.2.

Reference	Optimisation objective	Utility	Protection
[76, 25, 77] [78, 127, 35]	group-size constrained	optimal	
[53]	utility constrained	guarantee	
[85, 124, 80, 71]	protection constrained	optimal	guarantee

Table 2.11: Summary of existing grouping strategies w.r.t. their objectives

We now summarise the approaches discussed above as shown in Table 2.11 and compare them using two criteria that are important in applications [51, 4, 99]: whether they can optimise or guarantee data utility and whether they can optimise or guarantee privacy protection. As can be seen, the protection constrained approach is the only one that attempts to satisfy both data utility and privacy protection requirements, by achieving optimal utility for a certain protection requirement. However, it does not consider how utility and protection can be balanced, as it treats protection as a constraint anonymised data needs to satisfy and not as an optimisation objective. Furthermore, it can also be observed that none of the existing approaches can provide both data utility and protection guarantees.

Value recoding

There are three major techniques to recode QID values in data groups: bucketisation [123], microaggregation [43, 44] and generalisation [2] [110, 106]. Bucketisation works by releasing two separate tables T_q and T_s which contain the values of T in QIDs and SAs respectively, and a group membership attribute *Group-id* which specifies the associations between tuples in T_q and T_s. For example, Tables 2.12 and 2.13 are a pair of tables that together form a bucketisation of Table 2.1. Bucketisation allows data linkage between T_q and an external dataset, and achieves k-anonymity linkage between Group IDs.

Id	*Postcode*	*Group-id*
t_1	NW10	*1*
t_2	NW15	*1*
t_3	NW12	*1*
t_4	NW13	*1*
t_5	NW20	*2*
t_6	NW30	*2*
t_7	NW30	*2*
t_8	NW25	*2*

Table 2.12: Table containing QIDs (T_q)

Id	*Group-id*	*Salary (K)*
t_1	1	*10*
t_2	1	*10*
t_3	1	*10*
t_4	1	*10*
t_5	2	*20*
t_6	2	*40*
t_7	2	*40*
t_8	2	*30*

Table 2.13: Table containing SAs (T_s)

In contrast to bucketisation, microaggregation and generalisation explicitly distort QID values to make data linkage difficult. Microaggregation involves replacing a group of values in a QID using the group centroid [43] or median value [44] for numerical and categorical QIDs respectively. Generalisation suggests replacing QID values by more general but semantically consistent ones. Two generalisation models, called *global* and *local* recoding, have been proposed in the literature. In what follows, we

[2] We view suppression, i.e. tuple removal, as a special case of generalisation following [19].

briefly discuss these models and refer the reader to [76, 81] for more details and formal definitions.

Global recoding involves mapping the domain of QIDs into generalised values [76, 25], often consistently with DGHs. One such method uses *full domain* generalisation [106], where the entire domain of each QID is mapped to a more general domain, in a consistent way, i.e. all original QID values are mapped to generalised ones that lie at the same level of the given DGH for this QID. An example of this recoding model is shown in Figure 2.8. Given the hierarchy for *Postcode* shown in Figure 2.2, the values $NW10$ and $NW12$ are mapped to $NW[10-15]$, $NW16$ to $NW[16-19]$, and $NW20$ and $NW21$ to $NW[20-30]$. Observe that all of the generalised values lie at the same level of the DGH shown in Figure 2.2. Iyengar [64] considered a more flexible model called *full subtree* generalisation, according to which the path from each original value to the root in the DGH contains no more than one generalised value. For instance, when the full subtree model is employed, the values $NW10$ and $NW12$ are mapped to $NW[10-15]$, but $NW16$ is mapped to $NW[16]$, as shown in Figure 2.8. This is because $NW16$ is the only value in the path from $NW16$ to $*$ according to the hierarchy illustrated in Figure 2.2.

These models have been shown to enhance the efficiency of k-anonymisation methods by pruning the search space of generalisations, and to ensure semantical consistency of anonymised data. But they may incur excessive information loss [96, 25]. Thus, recent research has dropped the restrictions imposed by DGHs. Bayardo et al. [25] proposed an *ordered set-partitioning* model which uses a total order instead of a DGH. According to this model, generalised values in a QID need to define a partition according to the total order associated with this QID. For instance, given a totally ordered set $\{NW10, NW12, NW16, NW20, NW21\}$, the values in *Postcode* shown in Figure 2.8 can be generalised using a partition

$\{\{NW10, NW12\}, \{NW16\}, \{NW20, NW21\}\}$. Li et al. [81] considered an *unordered set-partitioning scheme* in which generalised values in this QID need to define a partition in the domain of this QID. In this model, a DGH or total order for a QID is not required. For example, the postcode values shown in Figure 2.8 can be generalised as $\{\{NW10, NW16\}, \{NW12\}, \{NW20, NW21\}\}$. Values in a QID are recoded by sets when ordered or unordered set-partitioning models are used.

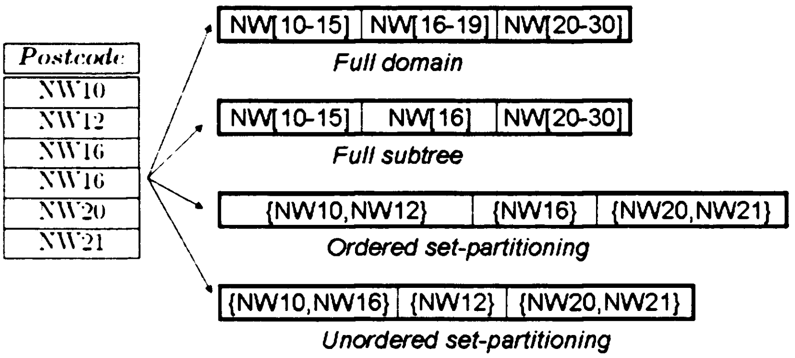
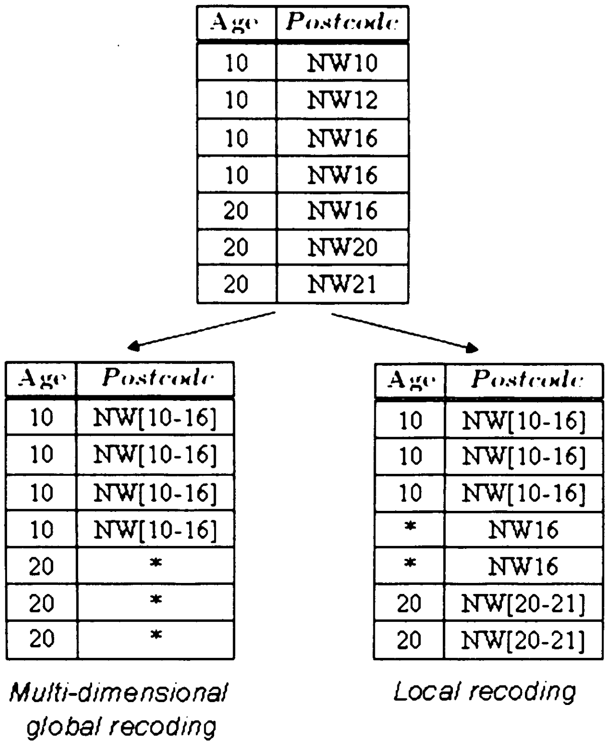


Figure 2.8: Global recoding under different single-dimensional models

All of the recoding models discussed so far are *single-dimensional*, i.e. they recode the domain of each QID separately. A *multi-dimensional* generalisation model that recodes the multi-attribute domain of QID values was proposed by LeFevre et al. [77] to further enhance utility. The main difference from the single-dimensional models is that two tuples having the same value in a QID attribute may be generalised differently. For instance, the postcode value $NW16$ in Figure 2.9 is generalised both as $NW[10-16]$ and $*$, when multi-dimensional global recoding is used.

Some works use *local recoding*, mapping QID values of individual tuples into generalised ones on a group by group basis [127]. Similarly to the multi-dimensional global recoding model, tuples with the same value in a QID attribute can be generalised differently in different groups when local recoding is used. However, local recoding differs from the multi-dimensional global recoding model in that it allows



Age	Postcode
10	NW10
10	NW12
10	NW16
10	NW16
20	NW16
20	NW20
20	NW21

Age	Postcode
10	NW[10-16]
10	NW[10-16]
10	NW[10-16]
10	NW[10-16]
20	*
20	*
20	*

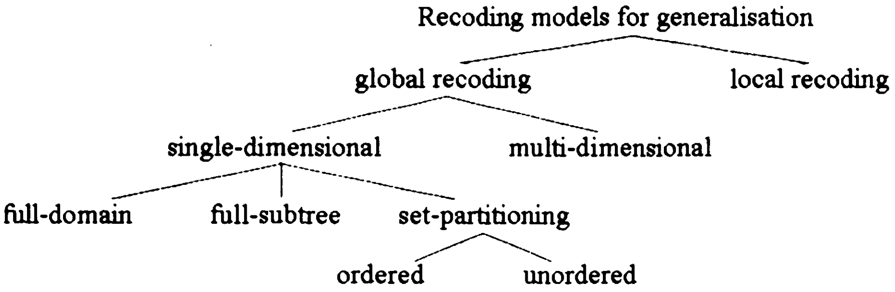
Multi-dimensional global recoding

Age	Postcode
10	NW[10-16]
10	NW[10-16]
10	NW[10-16]
*	NW16
*	NW16
20	NW[20-21]
20	NW[20-21]

Local recoding

Figure 2.9: Multi-dimensional global and local recoding models

tuples with the same values in all QIDs to be placed into different groups. For example, as illustrated in Figure 2.9, the two tuples having values 10 and $NW16$ in *Age* and *Postcode* respectively belong to different groups after anonymisation. Local recoding is a more general model than global recoding and was shown to achieve better data utility among all generalisation models [127, 35, 96]. Figure 2.10 summarises the discussed recoding models for generalisation.



Reference	Recoding model
[106]	Full-domain
[64]	Full-subtree
[25]	Ordered set-partitioning
[81]	Unordered set-partitioning
[77]	Multi-dimensional
[127]	Local recoding

Figure 2.10: A classification of generalisation recoding models

2.2 Related Privacy-Preserving Techniques

As we pointed out in Introduction, apart from k-anonymisation, other privacy-preserving techniques exist. These are designed either to prevent individual identification by ensuring that data published by a trusted data collector/publisher is sufficiently protected, or to guard against an untrusted data collector/user attempting to breach privacy by bypassing security mechanisms and mining data in a certain way. We refer to the first case as "privacy-preserving data publication" and the second as "privacy-preserving data usage", and we briefly review them in Sections 2.2.1 and 2.2.2 respectively.

2.2.1 Privacy-Preserving Data Publication

Generally speaking, protecting privacy in the data publication scenario is typically achieved by masking data before its release. Techniques that do so by seeking to release specific data about each individual [110, 124, 123, 95], or by attempting to preserve aggregate information (e.g. statistical information [100] or data distribution [22]) in the released data have been proposed.

Techniques of the first category are closely related to k-anonymisation since they share the same goal: hiding individuals' identity while enabling examination of data representing an individual. The most commonly used of these technologies is de-identification [73]. That is, replacing identifying attributes (e.g. names or phone numbers) by a generated key (e.g. a unique positive integer). De-identification is simple to perform and compliant with many privacy rules, such as the U.S. Food and Drug Administration [10] or HIPAA guidelines [13]. However, as mentioned in Introduction, this technique is unable to protect privacy when public data can be linked to de-identified one.

k-Anonymisation exerts the same amount of privacy for the data about each individual contained in the dataset. However, individuals may want to specify the level of data protection that they will receive from anonymised data based on their personal preferences. In response, Xiao and Tao proposed Personalised anonymity [124]. This privacy model differs from k-anonymisation in that the group size is not lower-bounded by k, i.e. anonymised groups can contain any number of tuples. Thus, this approach may not provide as strong protection from data linkage as k-anonymisation, since a released tuple may correspond to a single individual in the original dataset. Furthermore, Personalised anonymity applies generalisation to SA values in order to increase protection, and thus it may incur excessive information loss compared to approaches that allow original SA values to be released [85, 71, 80].

Techniques that attempt to publish protected data that preserves aggregate information are also important. However, they should not be considered as alternatives to k-anonymisation for two reasons. First, they may perturb data in a way that an individual's record cannot be examined separately, e.g. by releasing values that are not semantically consistent with this individual's original values. Second, despite hiding individuals' personal information, these techniques do not attempt to prevent data linkage.

Techniques falling into this category include data suppression, synthetic data generation, data swapping and randomisation. Data suppression [46, 23] works by removing unique or rare attribute values that are likely to breach privacy before data release, while synthetic data generation does not release the original dataset, but a synthetic one that preserves order statistics (e.g mean and variance) instead [105, 89]. However, much of the useful information included in the original data may be lost when either of these techniques is used. Data swapping [42, 100, 97] addresses this limitation by swapping the values so that much useful information, such as count and order statistics, are preserved. Randomisation involves adding noise to data in a way that "accurate models without access to precise information in individual data record" can be constructed [22]. Agrawal et al. [22] were the first to show how data distribution may be estimated fairly accurately from randomised data. A number of works considered applying randomisation to allow data mining tasks, such as classification [131], association rule mining [103] and top-k query answering [126], to be performed in a privacy-preserving way.

2.2.2 Privacy-preserving Data Usage

We now briefly consider techniques that attempt to prevent an untrusted data collector/user to breach data privacy. Detailed discussion on these techniques, which are orthogonal to k-anonymisation, can be found in [15, 116].

One way to guarantee data privacy in this setting is to prohibit illegitimate access to data. This is often achieved by using encryption [20, 60] or access control methods [55, 107, 104, 102]. Encryption has been used in numerous applications, such as to protect outsourced data from an untrusted service provider [60] (e.g. a company that modifies data stored by clients in its database or sells it to third parties) or to enable privacy preserving data integration [20]. Access control has been used in applications ranging from traditional database systems [55, 107] to multi-level secure ones, where users are grouped according to their security clearance level and can only manage data appropriate to their level [27].

Alternatively, some works attempt to control how data is used by employing query auditing [15, 68] or secure function evaluation [129]. Generally speaking, query auditing attempts to check whether a collection of queries which may compromise privacy have been performed, while the goal of secure function evaluation is to enable a number of parties to learn the result of a function while keeping their input private [129]. Techniques based on secure function evaluation were recently used to solve a number of data mining problems in a privacy-preserving way [114, 115, 66, 128, 34].

We classify the existing privacy-preserving technologies in Figure 2.11 and summarise their characteristics in Table 2.14.

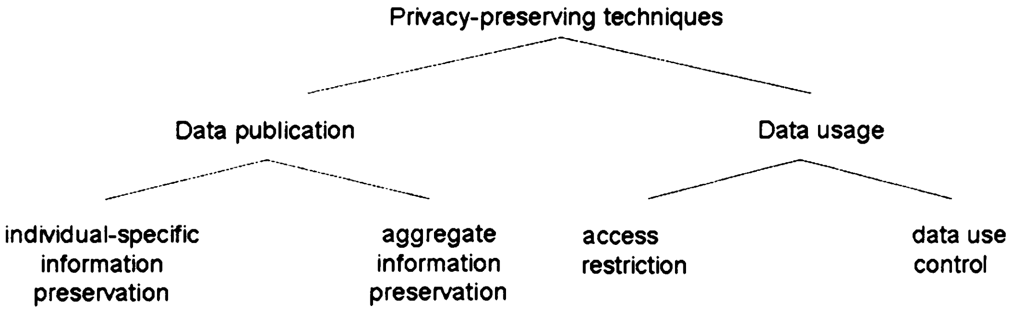

Privacy-preserving techniques
Data publication
Data usage
individual-specific information preservation
aggregate information preservation
access restriction
data use control

Figure 2.11: A classification of privacy-preserving techniques

Reference	Privacy Goal	Technique
[73]	indiv. inform. preserv.	de-identification
[124]	indiv. inform. preserv.	personalised anonymity
[46, 23]	aggreg. inform. preserv.	data suppression
[105, 89]	aggreg. inform. preserv.	synthetic data generation
[42, 100, 97]	aggreg. inform. preserv.	data swapping
[22, 131, 103, 126]	aggreg. inform. preserv.	randomisation
[20, 60]	access restriction	encryption
[55, 107, 104, 102, 27]	access restriction	access control
[15, 68]	data use control	query auditing
[129, 114, 115, 66, 128, 34]	data use control	secure function evaluation

Table 2.14: Summary of privacy-preserving techniques

2.3 Summary

In this chapter, we surveyed the existing k-anonymisation techniques that have been developed for privacy-preserving data publishing. We discussed existing utility and protection metrics, and considered three main components of a k-anonymisation algorithm: search strategy, optimisation objective and value recoding strategy. We also reviewed other privacy-preserving technologies that can help provide protection for data publication and data usage. We have seen that some data publication techniques, such as de-identification, allow publishing specific data about each individual, while others, such as randomisation, attempt to preserve aggregate information about individuals. Different from k-anonymisation, these techniques cannot provide strong protection for data linkage. Finally, we described a number of technologies that attempt to provide protection for data that may be used by an untrusted user.

Despite the progress that has been made in k-anonymisation in terms of quality measures and optimisation methods, there are still issues outstanding. In terms of utility measurement, with the exception of those based on reconstruction probability, most utility measures cannot capture the accuracy of a wide range of tasks that need to be performed using anonymised data. In terms of protection measurement, existing criteria focus largely on value disclosure, and those providing protection for range disclosure are either applicable to numerical attributes only, or can incur excessive information loss. Moreover, current k-anonymisation methods use partitioning or clustering to form groups, and tend to focus on either execution efficiency or quality of produced anonymisations, rather than aiming to achieve both. Finally, existing approaches tend to optimising utility or guaranteeing protection, and have not considered how anonymised data with a "good" level of both data utility and protection may be produced. The ability to meet these requirements in k-anonymisation is desirable for many applications.

Chapter 3

Trade-Off Constrained k-Anonymisation

The requirement for a good k-anonymisation is two-fold. On the one hand, the sensitive information associated with individuals contained in the dataset needs to be protected, and on the other hand released data should be useful in analysis. Ideally, anonymisations should maximise both protection and utility, but this is computationally impossible [85]. Many existing works follow a "protection constrained" approach, i.e. they attempt to satisfy utility and protection requirements by providing a sufficient level of protection while minimising information loss [85, 80, 71].

While minimising information loss is generally beneficial to data users, we claim that it may not represent the best interest of anonymisers or the individuals contained in the data. Suppose, for example, that there are 3 k-anonymisations (a, b and c) possible for a dataset, as shown in Figure 3.1. If the only requirement for the anonymisation is that it satisfies a minimum level of protection, then a is preferred as it offers more utility. This is what the protection constrained methods are designed to deliver. However, in some applications both data users and anonymisers may demand that data utility and privacy protection be kept as high as possible. For example, in location privacy preserving systems, location information of mobile users (e.g. mobile phone holders) is anonymised before being released to location-based service providers. Naturally, the data anonymiser in this case would like to link the current location of a mobile user to as a large geographical region as possible in order to provide adequate protection for him (i.e. having anonymisation b in Figure 3.1). Equally, the service providers in this case would prefer to have the geographical region associated with the mobile user as small as possible, so that they can provide more targeted service for him (i.e. having anonymisation a in Figure 3.1). In such scenarios, we argue that it is important to consider how to achieve a balance between data utility and privacy protection. That is, we should aim to derive c instead of a or b in Figure , which offers a good utility/protection trade-off.

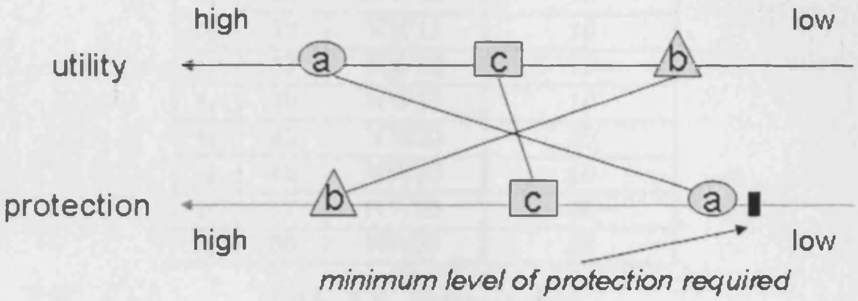


Figure 3.1: k-Anonymisation approaches

A Trade-Off Constrained k-Anonymisation Method

In response, we propose a method that allows this flexibility. Different from the protection constrained approach, our approach can represent an anonymiser's best interest in achieving an "optimal" trade-off between utility and protection. Consider Table 3.1 for example. We assume that *Id* is a tuple identifier which we use for illustration purposes, *Age* and *Postcode* are QIDs and *Salary* is an SA. Tables 3.2

and 3.3 illustrate two different 4-anonymisations of Table 3.1. Intuitively, Tables 3.2 and 3.3 can be considered as having exercised a different trade-off between utility and protection. For example, approximating the postcode value for the individual represented by t_1 using Table 3.2 is easier than doing so based on Table 3.3, as the number of possible postcode values this individual may have is smaller using Table 3.2. However, Table 3.2 can be considered as less protected than Table 3.3, as for instance, more accurate sensitive salary information can be inferred for the individual represented by t_1 when Table 3.2 is used. Knowing that this individual is aged 30, Table 3.2 would allow one to infer that this individual's salary is $10K$, while one can only infer a much larger range $[10K - 40K]$ when Table 3.3 is used.

Id	*Age*	*Postcode*	*Salary (K)*
t_1	30	NW10	*10*
t_2	32	NW15	*10*
t_3	37	NW12	*10*
t_4	40	NW13	*10*
t_5	45	NW20	*20*
t_6	46	NW30	*40*
t_7	57	NW30	*40*
t_8	60	NW25	*30*

Table 3.1: Original data

Id	*Age*	*Postcode*	*Salary (K)*
t_1	[30-40]	NW[10-15]	*10*
t_2	[30-40]	NW[10-15]	*10*
t_3	[30-40]	NW[10-15]	*10*
t_4	[30-40]	NW[10-15]	*10*
t_5	[45-60]	NW[20-30]	*20*
t_6	[45-60]	NW[20-30]	*40*
t_7	[45-60]	NW[20-30]	*40*
t_8	[45-60]	NW[20-30]	*30*

Table 3.2: A 4-anonymisation of Table 3.1

Id	*Age*	*Postcode*	*Salary (K)*
t_1	[30-46]	NW[10-30]	*10*
t_2	[30-46]	NW[10-30]	*10*
t_5	[30-46]	NW[10-30]	*20*
t_6	[30-46]	NW[10-30]	*40*
t_3	[37-60]	NW[12-30]	*10*
t_4	[37-60]	NW[12-30]	*10*
t_7	[37-60]	NW[12-30]	*40*
t_8	[37-60]	NW[12-30]	*30*

Table 3.3: Another 4-anonymisation of Table 3.1

The above example suggests one way of realising the utility/privacy trade-off: by balancing information loss in QIDs for data utility and range formation in SAs for privacy protection. Our approach to achieving an anonymisation with a utility/protection trade-off is based on this observation. Since data contains QID and SA values that can be grouped in many different ways depending on the quality requirements [44]. we first need to consider how utility and protection can be captured using quality measures. We then need to consider how anonymised data with "good" quality in terms of both utility and protection can be generated by incorporating these measures in an optimisation algorithm. Furthermore, it is essential that we can handle QIDs and SAs on equal terms, since this makes it possible to achieve a utility/protection trade-off. For instance, trading-off classification accuracy of anonymised data using the CM measure (see Definition 2.1.6) with l-diversity (see Definition 2.1.7) is theoretically possible, but is difficult in practice since these measures treat QIDs and SAs very differently.

In this thesis, we develop distance-based criteria that can capture the utility/privacy trade-off and treat QIDs and SAs uniformly. We show that our notion of distance on QIDs attempts to capture information loss based on the accuracy of reconstructing original values from generalised data, and it is applicable to both numerical and categorical QIDs. We explore the relation between adversarial background knowledge and the protection offered by k-anonymisation, and propose a measure to capture protection based on the maximum range of SA values in a group. Intuitively, when this range in a group is small, it is more likely for an attacker to establish the sensitive information associated with the individuals represented in this group.

We also propose an efficient algorithmic framework for k-anonymising data. Our method has a number of useful features:

- It produces anonymisations with a desired utility/protection trade-off. This is achieved by controlling the weighted sum of the amount of generalisation of QIDs and the amount of protection of SAs, as they are captured using our quality criteria.
- It uses a local recoding strategy to generalise QID values [127, 35]. This model considers a much larger number of different anonymisations [44] than the full-domain global recoding model employed by most protection constrained methods [85, 80], and thus can achieve better data utility [77, 121].
- It achieves both quality and efficiency in the k-anonymisation process by combining clustering with partitioning-based grouping strategies.
- It is practical to use, since it can be configured using fast and accurate sample-based heuristics and employs a flexible strategy to handle data skewness.

The rest of the chapter is organised as follows. In Section 3.1 we discuss how data utility and privacy protection are captured. We present our basic algorithmic framework and sample-based heuristics for it in Sections 3.2 and 3.3 respectively. Section 3.4 concludes the chapter.

3.1 Optimality Criteria

This section presents optimality criteria for utility and protection. For each criterion, its relation to existing criteria is also discussed.

3.1.1 Utility Criterion

Generalised data contains less information than original data, hence may affect the accuracy of some tasks that are performed on it. As discussed in Chapter 2, data utility is captured by either assessing how well anonymised data supports an intended task (e.g. classification accuracy [49]) or by directly measuring information loss incurred by generalisation [106, 25, 127, 96].

We propose measuring utility by quantifying information loss based on the accuracy of reconstructing original values from generalised data. To illustrate this idea, observe that tuple t_1 in Table 3.1 can be mapped to a point $(\pi_{Age}(t_1), \pi_{Salary}(t_1))$ in the 2D space defined by *Age* and *Salary*, where $\pi_{Age}(t_1)$ and $\pi_{Salary}(t_1)$ denotes the projection of t_1 on *Age* and *Salary* respectively.

Id	*Age*	*Salary (K)*
t_1	[30-40]	*10*
t_2	[30-40]	*10*
t_3	[30-40]	*10*
t_4	[30-40]	*10*
t_5	[45-60]	*20*
t_6	[45-60]	*40*
t_7	[45-60]	*40*
t_8	[45-60]	*30*

Table 3.4: A 4-anonymisation of Table 3.1

Now assume that Table 3.1 has been anonymised to Table 3.4. Due to generalisation, we can no longer be sure of t_1's original value, but we can model t_1 using a probability density function $f(x)$, where x is a random variable in the 2D space defined by *Age* and *Salary*. Without additional information and following the principle of indifference [65], it is assumed that the real age values in Table 3.4 are uniformly distributed and attributes are independent. Thus, $f(x)$ can be computed as

$$f(x) = \begin{cases} \frac{1}{11} & , \pi_{Age}(x) \in [30 - 40] \text{ and } \pi_{Salary}(x) = 10 \\ 0 & , \text{otherwise} \end{cases}$$

since the value of t_1 in *Age* lies in any point in $[30 - 40]$ with equal probability. Intuitively, the wider the interval in which $\pi_{Age}(x)$ lies, the less accurately we can reconstruct the value of t_1 w.r.t. *Age*.

Generalising the previous example, assume a table $T(a_1, ..., a_d)$ comprised of d attributes, m of which are QIDs and the remaining are SAs, and a k-anonymisation T^* of T. Furthermore, consider a tuple t belonging to a k-anonymous group τ in T^*. For each QID $a_i, i = 1, ..., m$, tuple t has a generalised value $\pi_{a_i}(\tau)$ which is an interval enclosing the original value of t in a_i, and for each SA $a_i, i = (m + 1), ..., d$, t has its original value $\pi_{a_i}(t)$. Given a random variable x in the d-dimensional space R defined by $a_1, ..., a_d$, t can be modelled using a probability distribution function $g(x) : R \rightarrow [0, 1]^d$ such that

$$g(x) = \begin{cases} p & , \pi_{a_i}(x) \in \pi_{a_i}(\tau), \forall i = 1, ..., m \text{ and } \pi_{a_i}(x) = \pi_{a_i}(t), \forall i = (m + 1), ..., d \\ 0 & , \text{otherwise} \end{cases}$$

where p denotes the probability x takes the original value of t in each attribute or equivalently the probability of reconstructing the original values of t in all attributes using τ. To compute p, we observe that τ can be represented as a d-dimensional hyperrectangle $H(\tau)$, having a side length $|\pi_{a_i}(\tau)|$ that is equal to the number of possible values a tuple in τ may have w.r.t. a_i, for $i = 1, ..., m$, and to 1, for $i = (m+1), ..., d$. Let also $V(H(\tau))$ be the volume of $H(\tau)$, which is given by $V(H(\tau)) = \prod_{i=1}^{m} |\pi_{a_i}(\tau)|$. Using this notation, p can be computed as

$$p = \frac{1}{\prod_{i=1}^{m} |\pi_{a_i}(\tau)|}$$

Thus, when the number of QID values that generalised tuples may have is large, reconstructing the values of these tuples become less accurate, and hence data utility becomes low.

Q-diversity

To quantify information loss, we first introduce Q-diversity, a measure that captures the distance between a set of QID values.

Definition 3.1.1 (Q-diversity). *Assume that a is a QID, the domain of a is D_a, and $V_a \subseteq D_a$ is a subset of values obtained from a. The* Q-diversity *of V_a, denoted by $qd(V_a)$, is defined as*

$$qd(V_a) = \begin{cases} \frac{max(V_a)-min(V_a)}{max(D_a)-min(D_a)} & \textit{numerical values} \\ \frac{s(V_a)-1}{|D_a|-1} & \textit{categorical values} \end{cases}$$

where $max(V_a)$, $min(V_a)$, $max(D_a)$ and $min(D_a)$ denote maximum and minimum values in V_a and D_a respectively. If a DGH h exists for a, then $s(Va)$ is the number of leaves of the subtree of h rooted at the closest common ancestor of the values in V_a. Otherwise, $s(Va)$ is the number of distinct values in V_a. $|D_a|$ is the size of domain D_a.

For a numerical QID, distances are naturally captured using the Euclidean distance. For a categorical attribute, however, there is no ordering among its values and distance between them is often defined in terms of their semantic relationships specified by a user-defined DGH. Given a DGH, we define distance between a group of categorical values as the ratio of the number of leaves of the subtree rooted at the closest common ancestor of these values to the domain size, subtracting 1 from each term to achieve a range of $[0, 1]$. If there is no hierarchy for a categorical attribute, we assume a flat DGH comprised of two levels: the ground level containing original values and the top level containing a generalised value.

Based on Q-diversity, we define *Utility Measure* (UM) to be the average Q-diversity across groups over all m QIDs, assuming attribute independence.

Definition 3.1.2 (Utility Measure (UM)). *Assume that a table T comprised of m QIDs $\{a_1, \ldots, a_m\}$ is partitioned into groups $\{g_1, g_2, \ldots, g_h\}$, such that $|g_j| \geq k, 1 \leq j \leq h$, and tuples of g_j will have the same values in each QID after anonymisation. The Utility Measure is defined as*

$$utility = avg(\frac{1}{m} \times \sum_{i=1}^{m} qd(\pi_{a_i}(g_1)), \ldots, \frac{1}{m} \times \sum_{i=1}^{m} qd(\pi_{a_i}(g_h)))$$

where $qd(\pi_{a_i}(g_j))$ denotes the Q-diversity of group g_j w.r.t. a_i, $1 \leq j \leq h$ and $1 \leq i \leq m$.

UM measures the perimeter of the hyper-rectangle of a generalised group (normalised by the number of dimensions). Since a hyper-rectangle with a small perimeter has also a small volume, a small UM score implies that the values of generalised data can be reconstructed accurately, and thus the level of data utility provided by this data is high. We note that we do not measure the volume of the hyper-rectangle of a group, since this may not achieve a uniform handling of all QIDs. This is because the volume of the hyper-rectangle of a group is biased towards QIDs with small domains [26]. Example 3.1.1 illustrates how UM can be computed.

Example 3.1.1. *Consider Table 3.2.* Age *is a numerical QID and* Postcode *a categorical QID whose values have been generalised according to the DGH shown in Figure 3.2. We first show how the Q-diversity score for the first group in Table 3.2 w.r.t.* Age *is computed. The ranges in* Age *for this group and the entire Table 3.2 are* $[30-40]$ *and* $[30-60]$ *respectively, thus the Q-diversity score for this group w.r.t* Age *is* $\frac{40-30}{60-30} = \frac{10}{30}$. *We can also compute the Q-diversity score w.r.t.* Postcode *for the same group in Table 3.2. According to the DGH in Figure 3.2, the closest common ancestor of values in this group is* $NW[10-15]$*, and the node* $NW[10-15]$ *in Figure 3.2 has 6 children (i.e.* $NW10, NW11, ..., NW15$*), while the entire DGH has* 21 *leaves in total. Thus, the Q-diversity score w.r.t.* Postcode *for the first group in Table 3.2 is* $\frac{6-1}{21-1} = \frac{5}{20}$. *Similarly, we can compute the Q-diversity scores w.r.t.* Age *and* Postcode *for the second group in Table 3.2, which are* $\frac{15}{30}$ *and* $\frac{10}{20}$ *respectively. Thus, the UM score for Table 3.2 is* $avg(\frac{1}{2} \times (\frac{10}{30} + \frac{5}{20}), \frac{1}{2} \times (\frac{15}{30} + \frac{10}{20})) = \frac{95}{240}$.

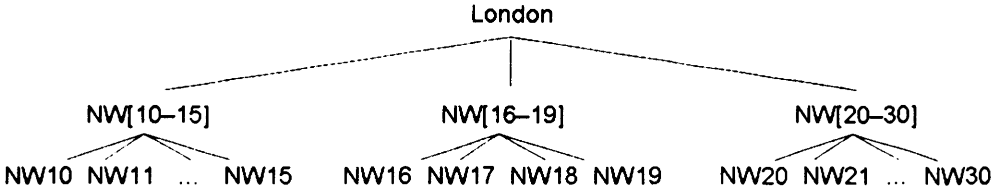


Figure 3.2: A DGH for *Postcode* in Table 3.2

Relation of Utility Measure to existing utility measures

As mentioned in Section 2.1.1, the Discernability Metric [25] and the Normalised Average Equivalence Class Size Metric [77] penalise a tuple based on the size of the group it belongs. However, the group size alone may not accurately capture information loss, since groups of the same size may be generalised differently. In contrast, UM takes into account how values are generalised and thus captures the distribution of QID values more precisely.

Example 3.1.2. *Consider Tables 3.2 and 3.3. Since both of these tables have two groups comprised of four tuples each, they have the same score in the Discernability Metric and the Normalised Average Equivalence Class Size Metric. In contrast, Table 3.2 has a smaller UM score compared to that of Table 3.3, since Table 3.2 has smaller ranges in* Age *and* Postcode. *We argue that UM correctly identifies Table 3.2 as incurring less information loss than Table 3.3, as it is intuitively plausible.*

We also note that UM is fundamentally different from utility measures that are based on the height of DGHs [106, 19]. Recall from Section 2.1.1 that, given a group of tuples, these measures quantify utility based on the level to which a generalised value lies at a DGH. UM, on the contrary, is directly related to the probability of reconstructing a tuple from generalised data. The main difference between the measures of [106. 19] and UM is highlighted in Example 3.1.3.

Example 3.1.3. *Consider two tables that have a single QID* Postcode *and are comprised of two tuples each. The postcode values in the first table are* $\{NW10, NW13\}$ *and those in the second are* $\{NW20, NW23\}$. *Furthermore, assume that these tables are* 2*-anonymised so that their tuples have generalised values* $NW[10-15]$ *and* $NW[20-30]$. *Since these generalised values lie in the second level of the DGH shown in Figure 3.2 the measures proposed in [106, 19] consider both tables as equally useful. However, the probability of reconstructing a tuple (see the equation on p.* 52*) is* $\frac{1}{15-10+1} \approx 0.167$ *and* $\frac{1}{30-20+1} \approx 0.091$ *for the first and second table respectively. Since the first table allows reconstructing the postcode values more precisely than the second, the first table is considered to be more useful. This is the case with the UM measure, with the UM scores being* $\frac{6-1}{21-1} \approx 0.25$ *and* $\frac{11-1}{21-1} \approx 0.5$ *for the first and second table* *respectively.*

Furthermore, unlike the utility measures proposed in [106, 19], *UM* allows all types of QID, numerical or categorical and with or without DGHs, to be handled uniformly.

Finally, we should note that information loss is quantified on a per group basis. Thus, UM is conceptually different from the Normalised Certainty Penalty [127] and other metrics based on the probability of reconstructing original QID values [64, 96, 124]. These measures all work on a per tuple basis. Thus, our approach allows to compare anonymisations of datasets with different characteristics in terms of their utility, as UM is independent of the dimensionality and cardinality of the table.

3.1.2 Protection Measure

This section presents a measure that captures the effect of data ranges on the protection of individuals' sensitive information. Before presenting our measure, we briefly consider a motivating example.

As discussed in Chapter 2, k-anonymity attempts to break harmful data linkage, but may not prevent sensitive information from being disclosed when the data in an SA is not diverse enough. To illustrate this, we applied an existing k-anonymisation method [77] to the Adults dataset [59] (a benchmark dataset for k-anonymisation algorithms), which contains census data of approximately 30000 US citizens. We used different values of k and measured the range in *Occupation*, which was treated as a numerical SA in this experiment. As can be seen from Figure 3.3, when k was set to 2, 57% of the SA groups had a zero range. This means that sensitive information about individuals can be accurately inferred if we know that someone's profile matches the QID values in one of these groups. Note that using a larger k may not eliminate the problem. For example, 21% of the SA groups still had a zero range when k was set to 5, as illustrated in Figure 3.4.

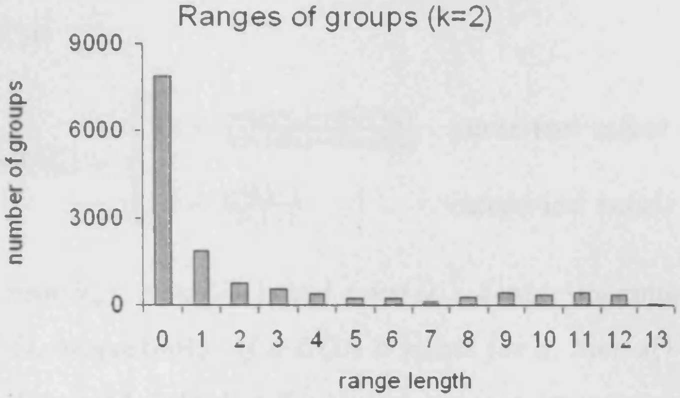


Figure 3.3: # SA groups vs. range length ($k = 2$)

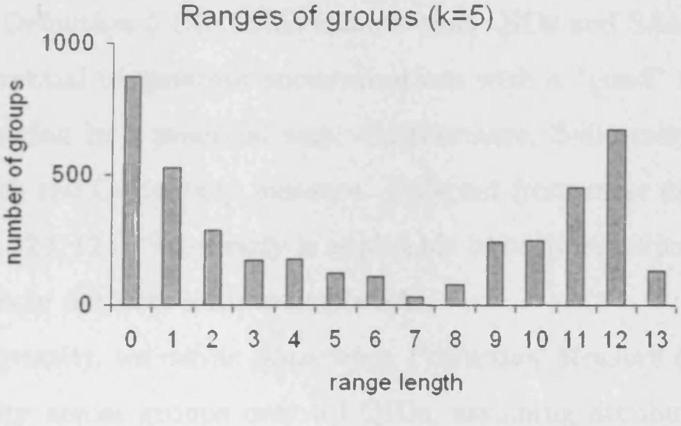


Figure 3.4: # SA groups vs. range length ($k = 5$)

As a narrower range w.r.t. SA values in a group may allow one to infer sensitive information more accurately, we can directly capture protection offered by a group by measuring the length of range w.r.t. the SA.

Definition 3.1.3 (S-diversity). *Assume that a is an SA, the domain of a is D_a, and $V_a \subseteq D_a$ is a subset of values obtained from a. The* S-diversity *of V_a, denoted by $sd(V_a)$, is defined as*

$$sd(V_a) = \begin{cases} 1 - \frac{max(V_a)-min(V_a)}{max(D_a)-min(D_a)} & \textit{numerical values} \\ 1 - \frac{s(V_a)-1}{|D_a|-1} & \textit{categorical values} \end{cases}$$

where $max(V_a)$, $min(V_a)$, $max(D_a)$ and $min(D_a)$ denote maximum and minimum values in V_a and D_a respectively. If a DGH h exists for a, then $s(Va)$ is the number of leaves of the subtree of h rooted at the closest common ancestor of the values in V_a. Otherwise, $s(Va)$ is the number of distinct values in V_a. $|D_a|$ is the size of domain D_a.

We note that S-diversity uses a similar notion of distance to the one used in Q-diversity (see Definition 3.1.1). This ensures that QIDs and SAs are handled uniformly, and is essential to generate anonymisations with a "good" trade-off between utility and protection in a practical way. Furthermore, S-diversity shares much of the flexibility with the Q-diversity measure. Different from most existing protection measures [85, 71, 124, 121], S-diversity is applicable to both numerical and categorical SAs, and can handle datasets with multiple SAs.

Based on S-diversity, we define *Max-range Protection Measure* (MPM) to be the average S-diversity across groups over all QIDs, assuming attribute independence. Intuitively, MPM reflects the average amount of protection that each group of tuples offers, and a small score is preferred.

Definition 3.1.4 (Max-range Protection Measure (MPM)). *Assume that a table T comprised of m SAs $\{a_1, \ldots, a_m\}$ is clustered into groups $\{g_1, g_2, \ldots, g_h\}$, such that $|g_j| \geq k, 1 \leq j \leq h$, and tuples of g_j will have the same values in each QID after anonymisation. The Max-range Protection Measure of T under this clustering is* *defined as*

$$mpm = avg(\frac{1}{m} \times \sum_{i=1}^{m} sd(\pi_{a_i}(g_1)), \ldots, \frac{1}{m} \times \sum_{i=1}^{m} sd(\pi_{a_i}(g_h)))$$

where $sd(\pi_{a_i}(g_j))$ denotes the S-diversity of group g_j w.r.t. a_i, $1 \leq i \leq m$.

Example 3.1.4 illustrates how MPM can be computed.

Example 3.1.4. *Consider Table 3.3.* Salary *is a numerical SA. We first show how the S-diversity score for the first group in Table 3.3 w.r.t.* Salary *can be computed. The range in* Salary *for this group is* $[10 - 40]$*, which is the range for the entire Table 3.3 w.r.t* Salary*. Thus, the S-diversity score for this group w.r.t* Salary *is* $1 - \frac{40-10}{40-10} = 0$*. Similarly, we can compute the S-diversity score w.r.t.* Salary *for the second group in Table 3.3. The range in* Salary *for this group is* $[30 - 10]$*, hence its S-diversity score is* $1 - \frac{30-10}{40-10} = \frac{1}{3}$*. This suggests that the first group offers better protection than the second one does. Finally, the MPM score for Table 3.3 is* $mpm = avg(0, \frac{1}{3}) = \frac{1}{6}$*.*

Relation of MPM to existing protection measures

A protection measure based on the idea of maximising ranges was also independently proposed by Koudas et al. [71]. As discussed in Section 2.1.2, it works by maximising the range in a group w.r.t. a numerical SA, and therefore it is similar to our MPM measure. LeFevre et al. [78] proposed to "disperse" the SA values in a group, using the variance of an numerical SA to capture protection. None of these protection measures can be applied to categorical SAs. This limits their applicability, since many datasets that need to be k-anonymised in practice contain mixed sensitive attributes. In contrast, our measure uses S-diversity and can handle both numerical and categorical attributes in a uniform way.

Several protection measures have also been proposed by considering the distribution of SA values rather than the range. These measures follow three general approaches. The *distribution controlling* approach followed by l-diversity [85] and other measures discussed in Section 2.1.2 [121, 113, 39], the *distribution mirroring* approach followed by t-closeness [80], and the *distribution hiding* approach followed by the measure proposed in [124]. Different from these measures, MPM is based on the length of ranges to capture protection, and offers a number of benefits compared to the distribution-based measures. Unlike the measures which are based on distribution control, MPM allows the handling of SAs with ordered domains (i.e. numerical SAs or categorical SAs with hierarchies). This makes our measure applicable to a wider class of datasets. Compared to t-closeness, MPM attempts to prevent inferences irrespectively of the distribution of SA values in the entire dataset. As mentioned in Section 2.1.2, this is more appropriate for some applications. Finally, MPM differs from the measure proposed in [124] in that it allows original SA values to be released. This is particularly important, as it can reduce the amount of information loss incurred to anonymise data.

3.2 Algorithmic Framework

In this section, we present our algorithmic framework which can generate anonymisations with a desired utility/protection trade-off. Our method uses both clustering and partitioning heuristics. That is, we first partition data into suitable sub-spaces, and then cluster data in each sub-space separately. Section 3.2.1 explains how greedy clustering can be employed to achieve high-quality anonymisations, and in Section 3.2.2 we propose the use of a multi-dimensional partitioning strategy to increase the efficiency of our clustering method without significantly degrading the quality of anonymisation.

3.2.1 Threshold-Based Greedy Clustering

Protection constrained k-anonymisation algorithms optimise data utility while satisfying a certain protection requirement [85, 80], but do not always produce data with a good trade-off between utility and protection [85, 106]. They often produce anonymisations with a high level of utility but no more than the minimal required level of protection. To derive k-anonymisations with "optimal" trade-off, we propose the following heuristic.

Definition 3.2.1 (Max-range Weighted Tuple Diversity). *Let $\tau \subseteq T$ be a set of tuples over a set of attributes $A = \{a_1, a_2, \ldots, a_d\}$. Without loss of generality, we assume that the first m attributes are QIDs and the rest are SAs. The* Max-range Weighted Tuple Diversity *of τ w.r.t. A, denoted by $mr\_wtd(\tau, A)$, is defined as:*

$$mr\_wtd(\tau, A) = w_u \sum_{i=1}^{m} \frac{qd(\pi_{a_i}(\tau))}{m} + w_p \sum_{i=m+1}^{d} \frac{sd(\pi_{a_i}(\tau))}{d-m}$$

where $\pi_{a_i}(\tau)$ denotes the projection of τ on attribute a_i, $qd(\pi_{a_i}(\tau))$ and $sd(\pi_{a_i}(\tau))$ denote the Q-diversity and S-diversity scores for $\pi_{a_i}(\tau)$ respectively, $w_u, w_p \in [0, 1]$ are the weights, and we require $w_u + w_p = 1$.

The idea of combining two search objectives into one using their weighted sum was originally proposed in multi-objective optimisation literature [33, 98]. However, to the best of our knowledge, such heuristics have not been used in the context of k-anonymisation nor incorporated into clustering algorithms. Our formulation has two interesting properties. First, it allows a trade-off between utility and protection to be made by placing different weights on Q-diversity and S-diversity measures. These weights can be specified by anonymisers, and changes in them can result in different anonymisations. Second, attributes of any type are treated uniformly, therefore datasets with mixed attributes can be handled.

Using this heuristic, the problem of finding an anonymisation with an optimal utility/protection trade-off can be formulated as in Definition 3.2.2.

Definition 3.2.2 (Optimal Clustering). *Let T be a table consisting of a set of attributes $A = \{a_1, a_2, \ldots, a_d\}$. Given a set of user specified weights w_u, w_p as defined in Definitions 3.2.1, an* optimal clustering *of T is a partition $P = \{g_1, \ldots, g_h\}$ of T such that $|g_j| \geq k, 1 \leq j \leq h$, $\bigcap_{j=1}^{h} g_j = \emptyset$, $\bigcup_{j=1}^{h} g_j = T$, and its* Average Max-range Weighted Tuple Diversity *score*

$$avg\_mr\_wtd(T, A) = \frac{\sum\limits_{j=1}^{h} mr\_wtd(g_j, A)}{h}$$

is minimal.

Unfortunately, solving the optimal clustering problem is NP-hard, as illustrated in Theorem 3.2.1.

Theorem 3.2.1. *The optimal clustering problem given in Definition 3.2.2 is NP-hard.*

Proof. Aggarwal et al. proved that optimally k-anonymising T using suppression is NP-hard [19]. We observe that this is a special case of our problem, where each QID is categorical, having a DGH with two levels (i.e. the root level corresponds to a suppression symbol and leaves are original QID values), and $w_p = 0$. Furthermore, verification of a partition P satisfying Definition 3.2.2 can be done in polynomial time. □

Thus, we introduce a heuristic *threshold-based greedy clustering* algorithm (shown in Algorithm 1) which works as follows. It randomly chooses a tuple t_i from a table T as a *seed* and makes it a cluster c (steps 2-3). Then, it iteratively extends c

with a tuple t_j that is closest to t_i w.r.t. the Max-range Weighted Tuple Diversity (*mr_wtd*). This is repeated until the *mr_wtd* of the cluster formed by adding t_j to c exceeds a threshold δ which controls the maximally-allowed *mr_wtd* score of a cluster (steps 4-8). Since extending c may affect both data utility and protection of the group, δ is used to ensure that the quality of anonymised data in c will be acceptable w.r.t. some specified threshold. When δ is exceeded, the size of cluster c is checked (step 9). The values of tuples contained in c are recoded if c contains at least k tuples, otherwise they are suppressed. We note that suppression may not always be a feasible choice in applications. For example, suppressing data in medical applications is usually unacceptable, since it may result in removing important information about some individuals. Acknowledging this issue, we will give an alternative strategy to suppression later in Section 3.3.

Algorithm 1 Greedy Clustering with Max-Range Weighted Tuple Diversity

```
1. while $T \neq \emptyset$ do
2.   $c \leftarrow t_i \in T$;
3.   $T \leftarrow T - \{t_i\}$;
4.   while true do
5.     find $t_j \in T$ s.t. $mr\_wtd(\{t_i, t_j\}, A)$ is minimum;
6.     if $(mr\_wtd(c \cup \{t_j\}, A) > \delta)$
         exit;
7.     $c \leftarrow c \cup \{t_j\}$;
8.     $T \leftarrow T - \{t_j\}$;
9.   if $(|c| \geq k)$
       recode($c$);
10.  else
       suppress $c$;
```

The proposed algorithm differs from existing clustering-based methods [35, 127, 96] in two main ways. First, it forms clusters based on both data utility and protection. Thus, it does not simply try to optimise utility, but attempts to achieve a "good" utility/protection trade-off. Second, it uses a quality-based instead of a size-based stopping criterion. That is, it restricts the maximally-allowed information diversity in a cluster instead of the maximum cluster size. This is because extending the cluster can help protection, since the maximum range of SA values in the cluster cannot decrease as a result of this. Thus, extending a cluster may help the Average Max-range Weighted Tuple Diversity.

Therefore, we use a threshold δ to control the level of utility/privacy trade-off in a cluster. δ can be selected by anonymisers according to a specified policy or in an automatic way using a heuristic to be discussed in Section 3.3.

On performance, Algorithm 1 has a quadratic time complexity to the cardinality of the dataset, since all candidate tuples are checked for insertion into a cluster every time a cluster is extended. However, sorting pairwise distances between the cluster seed and candidate tuples can speed up clustering. This is because a cluster can be formed by retrieving the $|c| - 1$ most similar tuples to a cluster seed in $O(|c|)$ time ($|c|$ is the cluster size), and sorting requires log-linear time to n, the cardinality of the dataset. Thus, assuming that each cluster is of size $|c|$, the complexity of Algorithm 1 becomes

$$O(|c| + (n-1) \times log(n-1) + |c| + (n-|c|-1) \times log(n-|c|-1) + ...)$$

$$\approx O(\frac{n}{|c|} \times |c| + \frac{n^2}{|c|} \times log(n)) \approx O(\frac{n^2}{|c|} \times log(n))$$

which is smaller than $O(n^2)$, as typically $|c| > log(n)$.

3.2.2 Median-Based Pre-partitioning

Attempts to improving clustering performance in general have been reported in the literature. The main idea behind these methods is to reduce the amount of computation required by pairwise distance comparisons by restricting search spaces using sampling [56] (i.e. they perform clustering on a data sample and then adjust clusters on the entire dataset), top-down bisection [127] (i.e. they use heuristics similar to bisecting k-means [109]) or a cheap pre-clustering step [90] (i.e. they generate clusters using a cheap to compute measure and subsequently refine them based on the original objective measure). All these methods are not efficient when a large number of small clusters are to be created, which is the case in k-anonymisation, and their similarity measures do not capture utility and protection, thereby affecting the quality of k-anonymisations that clustering can achieve. Thus, these solutions are not directly useful for k-anonymisation.

We propose a pre-partitioning step that is geared toward improving the performance of our threshold-based clustering without significantly affecting the quality of the anonymisations it produces. It follows a kd-tree type of partitioning [47, 77], which partitions data recursively into subspaces along the median of the QID with the largest normalised domain. Algorithm 2 shows our method. Given a set of tuples τ (initially the entire table T) and a size constraint s, the algorithm recursively derives a partition of τ as follows. First, it finds the QID attribute of τ that has the largest domain size and computes its median (step 2). Then, the data is partitioned around the median [1] (steps 3-5) until τ cannot be further divided without violating the size constraint (steps 6-7).

[1] When multiple tuples have the median value, we put half of them in each of the resultant subspaces.

Algorithm 2 Median-Based Pre-Partitioning

```
Pre-partition ($\tau, s$)
    find attribute $a_j \in QID$ s.t. values in $a_j$ have the largest range;
    $splitVal \leftarrow find\_median(\pi_{a_j}(\tau))$;
    $p \leftarrow \{t \in \tau : \pi_{a_j}(t) \leq splitVal\}$;
    $p' \leftarrow \{t \in \tau : \pi_{a_j}(t) > splitVal\}$;
    if ($|p| \geq s$ and $|p'| \geq s$)
        Pre-partition($p, s$) $\bigcup$ Pre-partition($p', s$);
    else return $\tau$;
```

Example 3.2.1 illustrates how Algorithm 2 works.

Example 3.2.1. *Figure 3.5 depicts microdata of 24 individuals.* Age *and* Height *are the QIDs, and Algorithm 2 is applied with s set to* 6. *First, the ranges in* Age *and* Height *are compared for the whole dataset. As can be seen, the range [20-70] in* Age *is larger than the range [150-190] in* Height, *thus data is split on the median along* Age. *This corresponds to cut C_1 in Figure 3.5. Then, Algorithm 2 is applied to each of the two resultant subspaces. Tuples of the first and the second subspace contain* Age *values in [20,45] and [46,70] respectively. Consider the first subspace. Since the range [150-190] in* Height *is larger than the range [20-45] in* Age, *data is split on the median along* Height. *This corresponds to cut C_2 in Figure 3.5, and results in subspaces S_1 and S_2. Finally, Algorithm 2 is applied on the second subspace. The range [150-190] in* Height *is larger than the range [46-70] in* Age. *Thus, cut C_3 along* Height *is performed, creating subspaces S_3 and S_4. Observe that the size of each of the resultant subspaces S_1, S_2, S_3 and S_4 is equal to the size constraint s. Consequently, no further split is possible without violating the size constraint and Algorithm 2 terminates, yielding four subspaces.*

A good partitioning strategy for improving the performance of clustering in k-anonymisation should satisfy three criteria. First, it should allow the follow-up clustering to be performed significantly more efficiently. This implies that each subspace

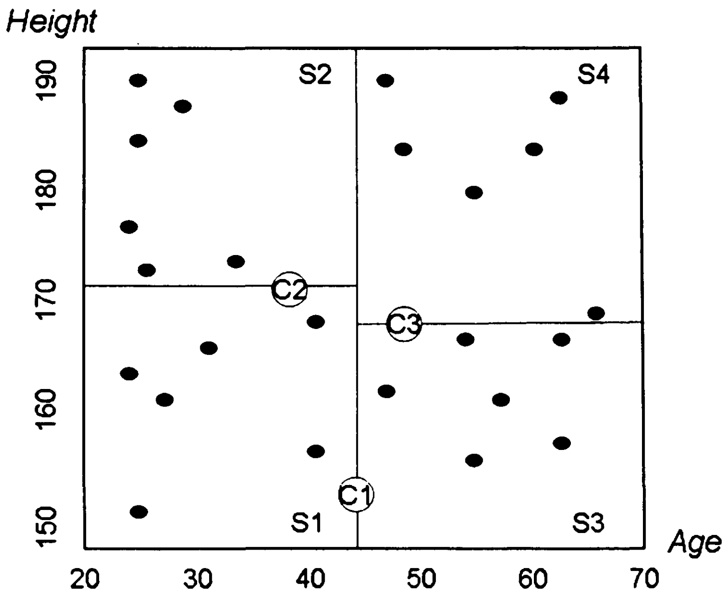


Figure 3.5: Illustration of Algorithm 2 with $s = 6$

should contain a small number of tuples. Second, it itself must be efficient, so that the cost of partitioning does not offset the savings gained by clustering in smaller subspaces. Third, the quality of clustering should not be affected too much. We now discuss our partitioning strategy in terms of these criteria.

Theorem 3.2.2 below shows that a significant speed up can be achieved if the subspaces created by pre-partitioning are relatively small, particularly when they are equal-sized. We note that our median-based partitioning strategy creates nearly equal-sized subspaces, as a result of splitting data around the median, thus is good for complexity reduction. Furthermore, it can be done efficiently [77], requiring only $O(n \times log(n))$ time to execute.

Theorem 3.2.2. *Pre-partitioning of a dataset T reduces the complexity of threshold-based clustering performed by Algorithm 1 to $O(s \times n)$, where n is the size of the* *dataset and s is the size constraint.*

Proof. Let $P = \{p_1, ..., p_h\}$ be a partition of T created by applying Algorithm 2. The worst-case time complexity of Algorithm 1 when applied to each p_i separately is $O(\sum_{i=1}^{h} |p_i|^2)$, which is minimised, using the standard Lagrange multiplier method [31], to $O(h \times |p_i|^2)$, when all p_i are equal-sized for a fixed h. Since we have $\sum_{i=1}^{h} |p_i| = h \times |p_i| = n$, the complexity of threshold-based clustering with pre-partitioning becomes $O(|p_i| \times n) \approx O(s \times n)$. □

We now consider the quality issues. First, we study the effect of pre-partitioning on data utility. Obviously, by partitioning data into subspaces, we restrict the number of tuples to be scanned while forming a cluster. Assume that S is a subspace and a cluster c is to be formed using $t \in S$ as a seed. We observe that the quality of c should not be affected if the nearest neighbours of t are also in S. This problem has been studied extensively in the context of multi-dimensional indexes [29]. To illustrate this, we give the following analysis.

Suppose that a dataset is cut along a QID q and two subspaces S_l, S_r are created as a result. It is possible that S_l and S_r will contain the same value in q (as we split at the median). In such cases, we say that the two subspaces overlap, and when this happens, a cluster may be better formed by considering tuples of both subspaces. Furthermore, as splits are based on only one QID at a time, subspaces may overlap in the remaining QIDs. Consider applying pre-partitioning to 2-anonymise the data shown in Table 3.5 with $s = 2$, for example. Since *Age* has a much larger domain than those of *Height* and *Postcode*, all splits are done around *Age* when Algorithm 2 is used prior to clustering. As pre-partitioning resulted in creating subspaces comprised of k tuples, applying clustering does not change the way data is grouped. Thus, data is anonymised as shown in Table 3.6. This anonymisation clearly incurs more information loss if we compare it to that shown in Table 3.7 which is derived by using clustering alone.

Age	Height	Postcode	Salary (K)
10	170	NW30	20
15	175	NW32	20
15	170	NW30	65
20	175	NW32	65
20	170	NW30	20
25	175	NW32	20
25	170	NW30	65
30	175	NW32	65

Table 3.5: Original data

Age	Height	Postcode	Salary (K)
[10-15]	[170-175]	NW[30-32]	20
[10-15]	[170-175]	NW[30-32]	20
[15-20]	[170-175]	NW[30-32]	65
[15-20]	[170-175]	NW[30-32]	65
[20-25]	[170-175]	NW[30-32]	20
[20-25]	[170-175]	NW[30-32]	20
[25-30]	[170-175]	NW[30-32]	65
[25-30]	[170-175]	NW[30-32]	65

Table 3.6: Partitioning Table 3.5

Age	Height	Postcode	Salary (K).
[10-15]	170	NW30	20
[10-15]	170	NW30	65
[15-20]	175	NW32	20
[15-20]	175	NW32	65
[20-25]	170	NW30	20
[20-25]	170	NW30	65
[25-30]	175	NW32	20
[25-30]	175	NW32	65

Table 3.7: Clustering Table 3.5

This is because, as illustrated in Figure 3.6, good groups depicted as small rectangles in the 3D space of $\{Age, Height, Postcode\}$, span across subspaces and thus are "broken" by pre-partitioning. Consequently, this results in less utility compared to what can be achieved when clustering without pre-partitioning is applied.

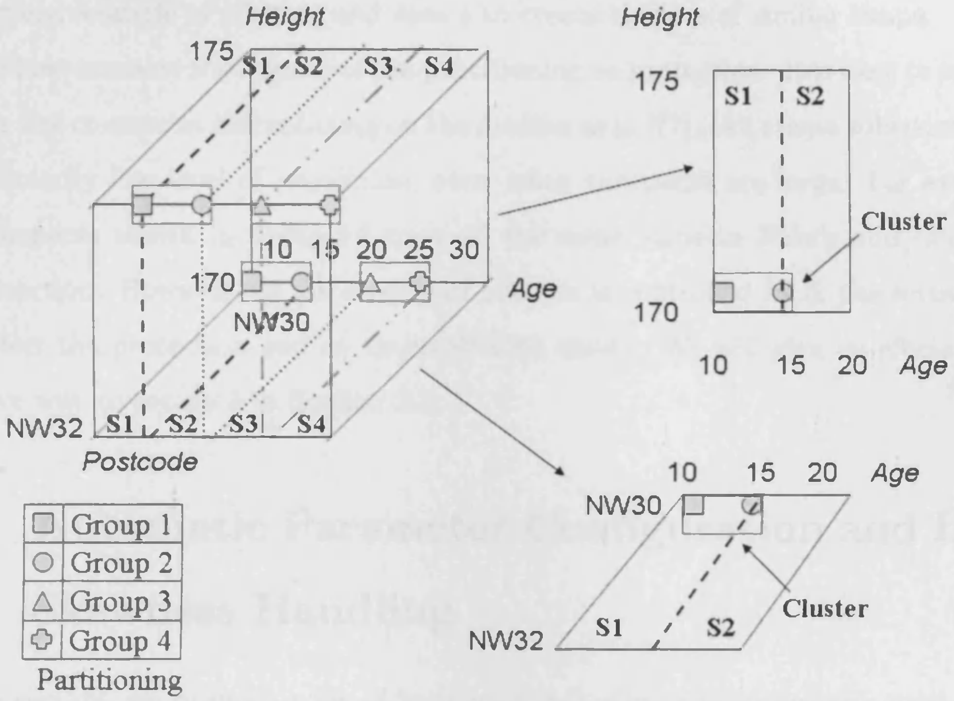


Figure 3.6: Clusters span across subspaces

So, if a clustering algorithm does not look beyond a single subspace when deriving clusters, a large number of overlapping subspaces may result in poor clusters. We observe however that the effect of overlap is significantly reduced if we use a sufficiently larger s than k. Our partitioning step ensures that this is the case by using a suitable subspace size threshold s. This threshold can be set by anonymisers or automatically as we will show in Section 3.3. Unclustering tuples of rejected clusters also helps, as they can be grouped together with their closest neighbours that may lie in a different subspace. Furthermore, we note that the size and shape of clusters can also affect the quality of anonymisations, as small (i.e. being represented as a hyper-rectangle whose perimeter is smaller than that of the hyper-rectangle of their corresponding subspace) and similar-shaped clusters are less likely to span across several subspaces created by median-based partitioning [94]. Algorithm 1 specifically minimises the perimeter of the hyper-rectangle of clusters and uses δ to create clusters of similar shape.

We now examine the impact of pre-partitioning on protection. It is easy to see that using a size constraint and splitting on the median as in [77], can create subspaces with an arbitrarily low level of protection, even when subspaces are large. For example, the subspaces shown in Table 3.6 have all the same value in *Salary* and thus offer no protection. However, as the quality of clusters is controlled by δ, the setting of δ can affect the protection and its trade-off with utility. We will give an efficient and effective way to set up δ in Section 3.3.

3.3 Automatic Parameter Configuration and Data Skewness Handling

In this section, we propose a set of heuristics that allow the parameters used in the method presented in Section 3.2 to be automatically configured. Furthermore, we present an alternative strategy to suppression that can be employed by this method. We begin by discussing why this may be useful in practice:

- We note that the efficiency and effectiveness of our method heavily depend upon the setting of thresholds. A "too small" size threshold for partitioning can mitigate the quality achievable by the subsequent clustering. For instance, partitioning the data in Figure 3.5 using $s = 3$ will result in four additional cuts $C4, C5, C6$ and $C7$ shown as dashed lines in Figure 3.7. This will exclude some close tuples (e.g. t) from being considered for clustering and thus may affect quality. Also, a "too large" quality threshold for cluster (denoted as δ' in Figure 3.7) may group distant tuples together, resulting in too much information loss when generalising the group and making the anonymised data practically useless. Unfortunately, relying on the data anonymiser to specify these thresholds may not always result in optimal settings.

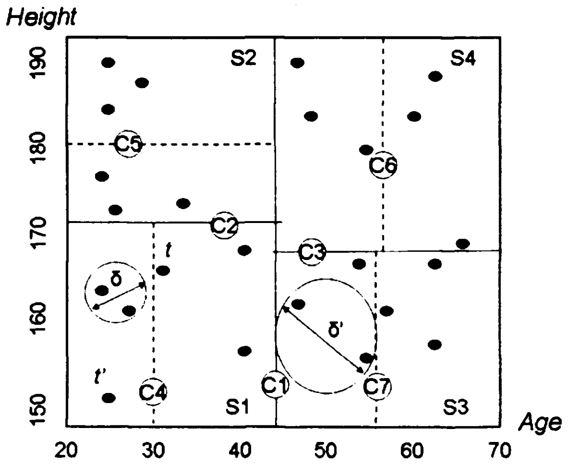


Figure 3.7: Effect of thresholds s and δ

- Suppressing tuples in clusters with less than k tuples can substantially degrade the overall quality of a k-anonymisation, particularly when the thresholds are not optimally set. For instance, tuple t' in Figure 3.7 has to be suppressed when $s = 3$ and the given δ is used. If many tuples are suppressed as a result of this policy, the anonymised data can be significantly distorted from the original one.

We now outline our methods that address the issues discussed above.

- In Section 3.3.1, we propose two simple and efficient heuristics to derive "good" δ and s thresholds for the method presented in Section 3.2 automatically. Our main observation is that the average cluster diameter of all clusters produced by our method is similar to that when clustering is applied to a small random sample of the dataset. Based on this observation, we apply clustering to a random sample a number of times, control and vary the δ and s values in each run, and choose the values that result in the best grouping w.r.t. utility and protection.

- Section 3.3.2 discusses a more effective strategy to handle clusters with tuples lying far from most other tuples in their subspace w.r.t. QIDs or close to most other tuples in their subspace w.r.t. SAs. We un-cluster the tuples in these clusters and put them in their closest cluster in any subspace. This simple heuristic improves quality, particularly when data is skewed, and is very efficient.

3.3.1 Sample-Based Heuristics

In this section, we present sample-based methods for determining the quality and size thresholds used in the algorithmic framework presented in Section 3.2.

Setting the quality threshold δ

We begin by discussing how δ used in the clustering part of our method (see Algorithm 1) can be set. The optimal setting of the threshold δ depends on data distribution. Intuitively, a "good" δ should mean that data is grouped in a way that the resultant clusters help anonymisation quality. There are two "obvious" ways to select a δ value. First, a specific data distribution can be considered, and δ is set for that distribution. For example, if we assume that data in clusters is uniformly distributed, then δ (i.e. the cluster diameter) can be set as a function of the distance between data items in the cluster and the cluster size [133]. However, there are two problems with this strategy: i) assuming a certain data distribution for a large number of very small clusters, which is the case in k-anonymisation, implies that the entire dataset, not just the clustered data, must follow a certain distribution; this is quite unlikely in practice; and ii) assuming a certain size for all clusters is unjustified as the size of a cluster is not the only factor that affects the quality of a k-anonymisation. The second method is to let the anonymiser perform clustering with different δ values and select one that results in the best grouping. This would require a substantial amount of experiment time, since our method (clustering with pre-partitioning) has a complexity of $O(n \times s)$, where n is the size of dataset and s the subspace threshold, and there is a very large number of δ values to be considered.

We propose a simple and efficient heuristic to approximate an optimal value for δ. The idea is to take a uniform random sample of the dataset with replacement, apply clustering without pre-partitioning to the sample several times, each time with a different δ value in $[0, 1]$, and choose the δ that results in the best grouping w.r.t. the sum of UM and MPM scores. The method is given in Algorithm 3.

Algorithm 3 Sampling-based selection heuristic for δ.

```
$S \leftarrow sample(T)$;
$i \leftarrow 1$;
while $i < max\_iterations$ do
    split $[0, 1]$ into $2^i$ equal-length intervals;
    choose the mean of each interval as $\delta_r$;
    for each $\delta_r$ do
        apply clustering without pre-partitioning to $S$;
choose $\delta_r$ s.t. the sum of $UM$ and $MPM$ scores for $S$ is minimum;
```

Two conditions determine the success of this heuristic. First, the sum of utility and protection (i.e. the quality of anonymisation) achieved by clustering for the sample with δ_r should be similar to that achieved when the same algorithm is applied to the whole dataset. Second, only a small data sample needs to be used and a small number of different values of δ_r need to be considered. Otherwise, the heuristic will not be efficient.

We observed through our experiments that both conditions hold. Since each tuple is equally likely to be included in the sample due to uniform random sampling, most clusters are well represented in the sample. This is illustrated in Figure 3.8, which shows the result of applying this heuristic to a 2.5% random sample of the Adults dataset [59], with $k = 5$, $w_u = w_p = 0.5$ and a maximum of 3 iterations. As can be seen, δ can be fairly accurately estimated, since scores for utility and protection are similar when our algorithm is applied to the sample and to the entire dataset.

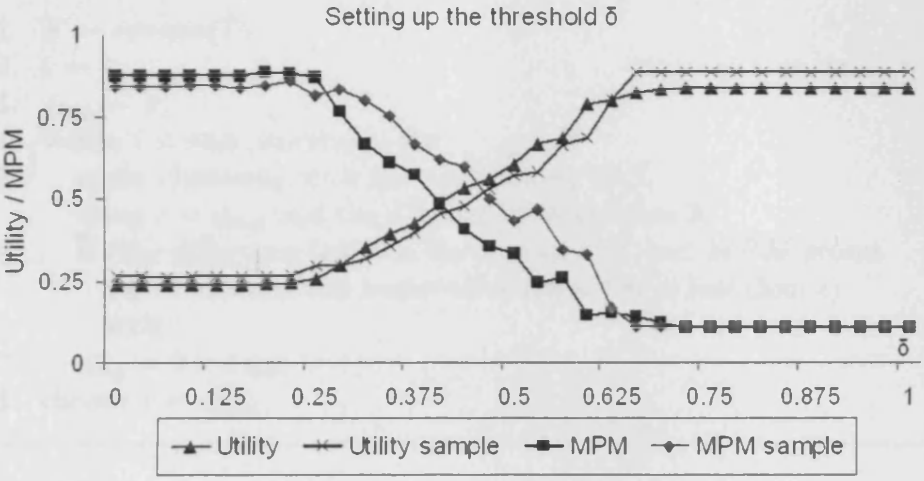


Figure 3.8: δ selection heuristic for Algorithm 1

Since this heuristic requires applying Algorithm 1 i times on a sample of data, its time complexity is $O(i \times \frac{n_s^2}{|c|} \times log(n_s))$, where n_s is the sample size.

Setting the size threshold s

A good value for s should achieve a good speed up factor for our method without affecting the quality of clustering too much. To find such an s, we propose the following heuristic. We apply clustering with pre-partitioning to the same sample used in Algorithm 3, using the optimal δ value found by this algorithm and setting s to k. This is repeated i times, and each time we double s. This process implies that clustering is applied to each of the last i levels of the tree created by pre-partitioning. Intuitively, using a larger s improves the quality of clustering due to the larger space explored, but speed deteriorates. So our heuristic stops when the quality improvement between consecutive levels does not exceed a threshold ϵ. Algorithm 4 illustrates our heuristic.

Algorithm 4 Sampling-based selection heuristic for s.

```
1. $S \leftarrow sample(T)$;
2. $i \leftarrow 1$;
3. $s_{tmp} \leftarrow k$;
4. while $i < max\_iterations$ do
5.    apply clustering with pre-partitioning to $S$
      using $s = s_{tmp}$ and the $\delta$ found by Algorithm 3;
6.    if (the difference between the sum of $UM$ and $MPM$ scores
        for $S$ between two consecutive iterations is less than $\epsilon$)
        exit;
7.    $s_{tmp} = 2 \times s_{tmp}$;
8. choose $s = s_{tmp}$;
```

Since our method requires $O(n_s \times s)$ time in each step, the complexity of this heuristic is $O(n_s \times s + n_s \times 2 \times s + \ldots + n_s \times 2^{i-1} \times s) = O(n_s \times s \times (2^i - 1))$. However, we observed that quality converges typically after $3 - 4$ iterations in our experiments and thus the efficiency of our method is not substantially affected. Figure 3.9 shows the result of applying this process to the Adults dataset with the same settings as the one for deriving δ in Figure 3.8. As can be seen, after executing our heuristic 3 times (i.e. when s was $4 \times k = 20$), we found that the utility and protection are relatively stable (i.e. the difference in quality scores between two consecutive levels does not exceed ϵ). Thus, setting s to $4 \times k = 20$ can be a good choice.

Clearly, selecting a good value for s will result in grouping most tuples into the

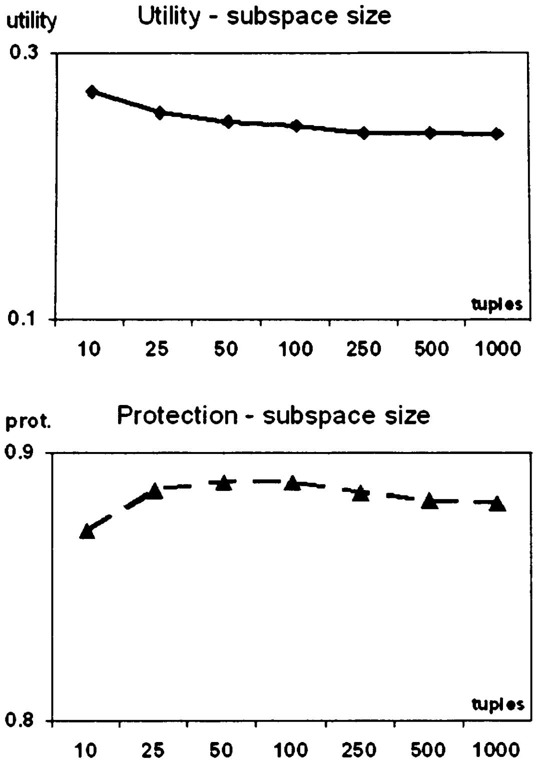


Figure 3.9: s selection heuristic

"right" clusters. However, since Algorithm 2 has no explicit information loss and protection control, a large number of clusters containing tuples that lie far from (close to) most other tuples in their subspace w.r.t. QIDs (SAs) can be created. This can have a significant effect on both data utility and protection. Theorems 3.3.1 and 3.3.2 show the effect of using pre-partitioning with a poor s on data utility and protection respectively.

Theorem 3.3.1. *The utility measure (UM) score achieved by Algorithm 1 when it is applied with pre-partitioning can be up to* $\lfloor \frac{n}{s} \rfloor \times (\frac{1}{\lfloor \frac{n}{2s-1} \rfloor} + (m-1))$ *times larger than the UM score achieved by the same algorithm applied without pre-partitioning, where*

m is the number of QIDs and s the size constraint used in pre-partitioning.

Proof. We will first compute the worst UM score achievable by applying Algorithm 1. This is achieved when pre-partitioning is applied with $s = k$ creating w subspaces by splitting along the same QID a_1 in every iteration (assuming that the domain of a_1 is always the largest) and these subspaces contain all possible values of D_{a_j} for all $j = 2, ..., m$. Note that although we do not know the exact value for the Q-diversity of the splitting attribute in each subspace, we know that the maximum sum for Q-diversity scores over all w subspaces in this attribute is 1. That is pre-partitioning created neighbour subspaces with tuples that have the same value in a_1 across their boundaries. We can also compute the maximum sum for Q-diversity scores over all subspaces in any of the non-splitting attributes. As each subspace contains all possible values of a non-splitting attribute, this sum equals w. Thus, summing up the Q-diversity scores for all the attributes, we get $1 + w \times (m - 1)$. Dividing the sum of Q-diversity scores for all the subspaces by w and taking the average over all m QIDs (see Definition 3.1.2) we get a UM score of $\frac{1}{m} \times (\frac{1}{w} + (m-1))$. Furthermore, since pre-partitioning creates at least $\lfloor \frac{n}{2s-1} \rfloor$ subspaces (each subspace can contain up to $2s - 1$ tuples to satisfy the size constraint s), we have $\frac{1}{m} \times (\frac{1}{w} + (m - 1)) \leq \frac{1}{m} \times (\frac{1}{\lfloor \frac{n}{2s-1} \rfloor} + (m - 1))$.

Now, consider the best UM score that we can get by applying Algorithm 1 on this dataset, which is achieved when the Q-diversity of each subspace (measured over all QIDs) is minimum. That is, the domain of each QID has a size of w, and data is grouped in w clusters such that their values in all QIDs apart from the splitting-attribute a_1 are the same. Then, the best UM score is obtained when the sum of Q-diversity scores for a_1 over all the clusters is 1 (note that this sum cannot be less than 1, as pre-partitioning creates the best possible subspaces for a_1), all tuples in a cluster have the same value in each $a_j, j = 2, \ldots, m$ and all these attributes are numerical (so that their Q-diversity scores are 0). Thus, summing up the Q-diversity scores for all subspaces results in a sum of $1 + w \times (0) = 1$. Dividing the sum of Q-diversity scores for all the subspaces by w and taking the average over all m QIDs we get a UM score of $\frac{1}{m} \times \frac{1}{w}$. Therefore, since at most $\lfloor \frac{n}{s} \rfloor$ clusters can be created, and only a_1 contributes to the UM score, we have $\frac{1}{m} \times \frac{1}{w} \geq \frac{1}{m} \times \frac{1}{\lfloor \frac{n}{s} \rfloor}$.

Dividing the worst with the best UM scores achievable by applying Algorithm 1, we get

$$\frac{\frac{1}{m} \times (\frac{1}{\lfloor \frac{n}{2s-1} \rfloor} + (m-1))}{\frac{1}{m} \times \frac{1}{\lfloor \frac{n}{s} \rfloor}}$$

which gives us $\lfloor \frac{n}{s} \rfloor \times (\frac{1}{\lfloor \frac{n}{2s-1} \rfloor} + (m-1))$. □

Theorem 3.3.2. *The Max-range protection measure (MPM) score achieved by Algorithm 1 when it is applied with pre-partitioning can be arbitrarily larger than the MPM score achieved by the same algorithm when applied without pre-partitioning.*

Proof. Without loss of generality, assume that a dataset is comprised of j SAs and has been partitioned using pre-partitioning into w subspaces, in a way that each subspace has the same value in each SA. In this case, each of the SAs has an S-diversity value of 1 (i.e. $1 - \frac{1-1}{|Da_i|-1}$ or $1 - \frac{0}{max(Da_i)-min(Da_i)}$ for each categorical or numerical SA) for each cluster. Averaging the S-diversity scores over all j SAs, we get an MPM score of $\frac{1}{j} \times \sum_j 1 = 1$, which is the maximum possible MPM score an anonymised table can have.

Now, consider applying clustering on the whole dataset. Observe that the best MPM score is obtained when the S-diversity of each subspace measured over SAs has the smallest possible value. This occurs when the size of the domain of each SA is w (recall that w is the number of subspaces created when pre-partitioning is used) and clustering creates w clusters s.t. each SA contains values that cover its whole domain. In this case, the sum of S-diversity measured over SAs is minimised to $1 - \frac{max(Da_i)-min(Da_i)}{max(Da_i)-min(Da_i)} = 0$, when all SAs are numerical. Thus, the smallest MPM score is $\frac{1}{j} \times \frac{0}{w} = 0$, which is the minimum possible score an anonymised table can have. The proof follows by comparing the MPM scores of 1 and 0 obtained by applying clustering with and without pre-partitioning respectively. □

As an example, consider Tables 3.9 and 3.10, which are 2-anonymisations of Table 3.8 obtained by applying clustering with and without pre-partitioning respectively, using a size constraint $s = k$. Since *Age* has a much larger domain than those of *Height* and *Postcode*, all splits are done around *Age* when clustering is applied with pre-partitioning. This created the clusters shown in Table 3.9. It can be verified that the quality of the clusters produced is lower than those created by using clustering alone (see Table 3.10). We note that the UM scores are 0.75 and 0.25 for Tables 3.9 and 3.10 respectively. That is, clustering with pre-partitioning produced an anonymisation with a UM score that is 3 times larger than the UM score achieved by clustering alone. Although Theorem 3.3.1 suggests that applying clustering with pre-partitioning can achieve a *UM* score that is $\lfloor \frac{n}{s} \rfloor \times (\frac{1}{\lfloor \frac{n}{2s-1} \rfloor} + (m-1)) = \lfloor \frac{8}{2} \rfloor \times (\frac{1}{\lfloor \frac{8}{3} \rfloor} + 2) = 10$ times larger than that achieved by clustering alone, this bound is not reached, since pre-partitioning creates subspaces containing s, but not $2s - 1$ in this example. Furthermore, the MPM score is 1 and 0 for the clusters in Tables 3.9 and 3.10 respectively, thereby confirming that the protection achieved when clustering is applied with pre-partitioning can be arbitrary larger than when clustering is applied without pre-partitioning.

3.3.2 Data Skewness Handling

Data skewness may also deteriorate the quality of data grouping in two cases illustrated in Figures 3.10 (a) and (b) respectively. In Figure 3.10 (a), tuples t and t'

Age	Height	Postcode	Salary (K)
10	170	NW30	20
15	175	NW32	20
15	170	NW30	65
20	175	NW32	65
20	170	NW30	20
25	175	NW32	20
25	170	NW30	65
30	175	NW32	65

Table 3.8: Original data

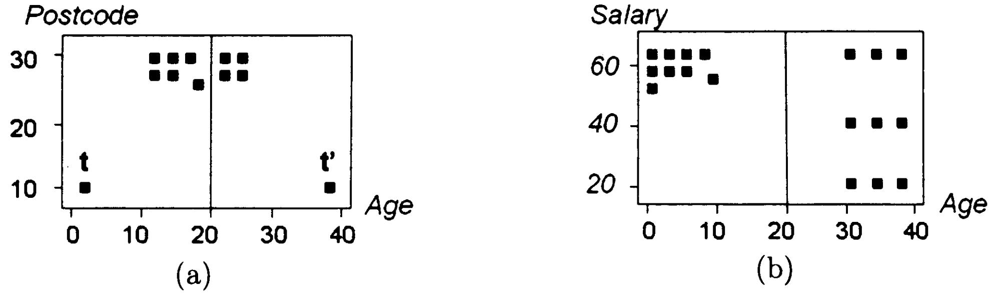
Age	Height	Postcode	Salary (K)
[10-15]	[170-175]	NW[30-32]	20
[10-15]	[170-175]	NW[30-32]	20
[15-20]	[170-175]	NW[30-32]	65
[15-20]	[170-175]	NW[30-32]	65
[20-25]	[170-175]	NW[30-32]	20
[20-25]	[170-175]	NW[30-32]	20
[25-30]	[170-175]	NW[30-32]	65
[25-30]	[170-175]	NW[30-32]	65

Table 3.9: A 2-anonymisation of Table 3.8 by clustering with pre-partitioning

Age	Height	Postcode	Salary (K).
[10-15]	170	NW30	20
[10-15]	170	NW30	65
[15-20]	175	NW32	20
[15-20]	175	NW32	65
[20-25]	170	NW30	20
[20-25]	170	NW30	65
[25-30]	175	NW32	20
[25-30]	175	NW32	65

Table 3.10: A 2-anonymisation of Table 3.8 by clustering without pre-partitioning

which lie across the x-axis force the pre-partitioning algorithm to split along this axis, separating close tuples. Similarly, as shown in Figure 3.10 (b), the pre-partitioning algorithm ignores how SA values are distributed in each subspace, and groups together tuples that have salary values that form a small range in the left subspace. These groups have a large Max-range Weighted Tuple Diversity score, and thus their tuples may not be grouped with other tuples after clustering, even though a generous δ is used.



(a) (b)

Figure 3.10: The effect of data skewness on partitioning

As a result, two types of group may occur: i) there are less than k tuples in a subspace, or ii) there are more than k such tuples, but cannot be grouped together to form a cluster with acceptable quality. These two cases are depicted as small rectangles in subspaces 3 and 2 respectively in Figure 3.11, where k is set to 4. In both cases, the clustering step of the method presented in Section 3.2 suppresses these tuples. However, this can greatly affect data utility when there is a large number of suppressed tuples.

We propose a heuristic which is applied after Algorithm 1 has finished with forming clusters, as illustrated in Algorithm 5. Steps 9-14 replace the steps 9 and 10 in Algorithm 1, while steps 15-16 are additional.

Clusters with at least k tuples are added into set S in step 11, while tuples in the remaining clusters are added into O in steps 12-14. Then, in steps 15-16, the tuples in O are examined in the order they were added into O and each of them is inserted into a cluster from S that is "best" for it, i.e. the one that will result in a minimal increment in Max-range Weighted Tuple Diversity (mr_wtd) when the tuple

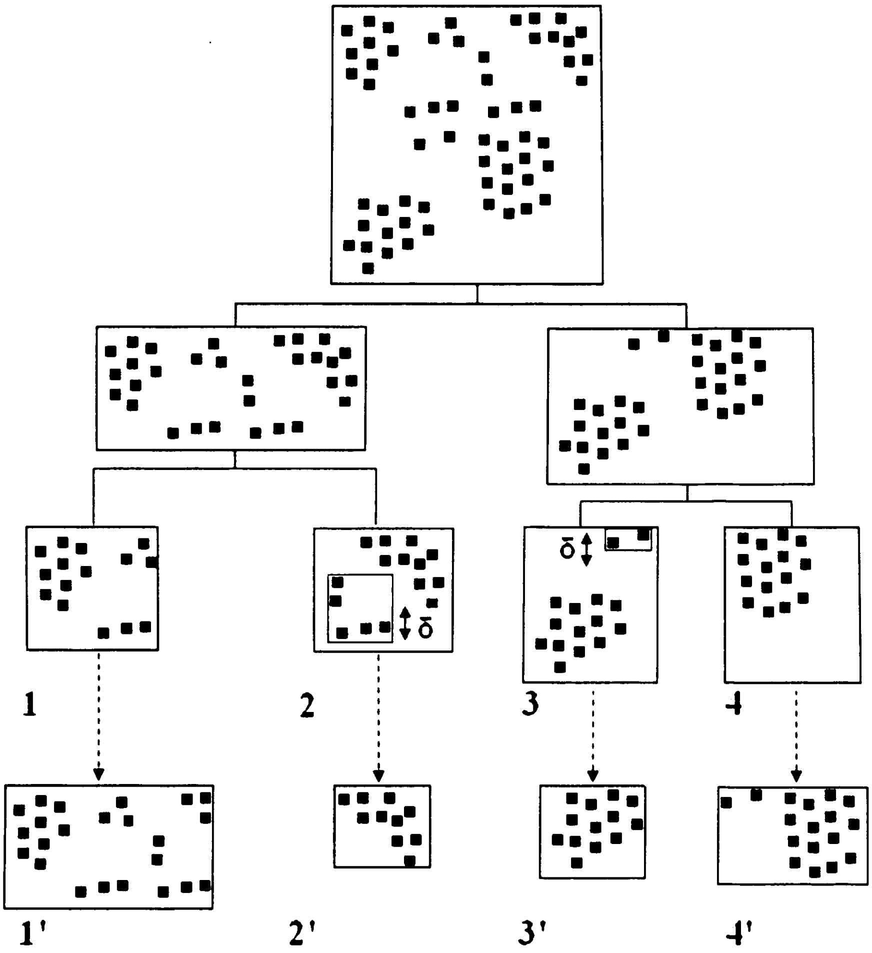


Figure 3.11: The problem of left-over tuples in subspaces

is inserted. Note that we check each tuple with all clusters with at least k tuples, since the nearest neighbours of a tuple can lie in different subspaces when data distribution is skewed [94]. It is also worth noting that choosing thresholds δ and s automatically as described in Section 3.3.1 results in a small number of tuples that need to be handled by Algorithm 5, and thus this heuristic is efficient in practice as shown in our experiments described in Chapter 4. Applying this heuristic to subspaces $1, 2, 3$ and 4 shown in Figure 3.11, for example, results effectively in subspaces $1', 2', 3'$ and

Algorithm 5 Data skewness handling heuristic for Algorithm 1

```
⋮
      **if** $|c| \geq k$
        recode($c$);
        $S \leftarrow S \cup c$;
      **else**
        **for each** $t_i \in c$ **do**;
          $O \leftarrow O \cup \{t_i\}$;
**for each** $t_i \in O$ **do**
  find $c_j \in S$ s.t. $c'_j \leftarrow c_j \cup \{t_i\}$ and $mr\_wtd(c'_j, A)$ is minimum;
```

4′, which retain all the tuples without significant data quality deterioration.

3.4 Summary

Motivated by the fact that existing approaches often sacrifice utility for increased protection and vice versa, we considered the problem of anonymising data with an optimal utility/privacy trade-off. We first introduced measures to capture data utility and privacy protection. These measures are applicable to both numerical and categorical attributes, with or without domain generalisation hierarchies, and are independent of the cardinality and dimensionality of the considered dataset. Therefore, they can be used to anonymise datasets of different characteristics. We also developed an algorithmic framework to generate data with a good utility/privacy trade-off. Our method uses partitioning and clustering heuristics, attempting to combine the efficiency of partitioning in exploring the search space with the high quality of data grouping achievable by clustering. More specifically, we proposed the use of partitioning to help reducing the time complexity of clustering, which is itself efficient to perform, and a clustering algorithm that reduces the effect of overlapping subspaces created by the partitioning step and uses a threshold controlled optimisation strategy.

Finally, we examined how to enhance the applicability of this framework in practice, by developing a set of heuristics to optimally set up parameters and to handle data skewness.

Chapter 4

Evaluation of Trade-Off Constrained k-Anonymisation

In this chapter, we experimentally evaluate our Trade-Off Constrained (TOC) approach to k-anonymisation. Section 4.1 describes the experimental setup and the datasets used in our experiments. Section 4.2 investigates the effectiveness of our sample-based heuristics, while Section 4.3 examines the efficiency of our method. Finally, we evaluate the effectiveness of our method w.r.t. the utility/protection trade-off and w.r.t. each of these properties in Sections 4.4 and 4.5 respectively.

4.1 Experimental Setup and Datasets

We compare our MR Greedy (MRG) method to three benchmark algorithms: Mondrian [77] which is partition-based and very efficient; K-Members [35] which is clustering-based and can achieve very good data utility; and K-Members-p which is a modified version of K-members where we changed its objective function to optimise protection (based on the same protection measure used in our method) instead of utility, so it can achieve very good protection. Both Mondrian and K-members have been shown to achieve better utility than most of the other existing algorithms, e.g. those proposed in [76, 25, 17]. For reference, Table 4.1 summarises the methods included in experiments and their characteristics. All the algorithms were implemented in Java and ran on a Pentium-D 3GHz machine with 1 GB of RAM under Windows XP.

Method	***Type of heuristic***	***Objective***	***Time Complexity***
Mondrian	partitioning	max. utility	$O(n \times log(n))$
K-Members	clustering	max. utility	$O(n^2)$
K-Members-p	clustering	max. protection	$O(n^2)$
MRG	partitioning and clustering	utility/protection trade-off	$O(n \times s)$ [1]

Table 4.1: Evaluated algorithms

Our main objective is to investigate how the quality of solutions with a trade-off between utility and protection produced by our methods compares to those produced by Mondrian, K-Members and K-Members-p, which are designed to specifically optimise one of these properties. We note that we have not considered protection constrained methods [85, 80] in our experiments because these methods have been shown to achieve worse data utility than Mondrian and K-Members, while they do not optimise protection as K-Members-p does. Data utility is evaluated using the Utility Measure (UM), Discernability Measure (DM) [25], and the accuracy of aggregate query answering [77, 124, 123]. We also quantify protection using our MPM measure. In addition, we study the efficiency and scalability of our method.

We have used both benchmarking and synthetic datasets in the experiments. First, we used the Adults dataset [59], which has become a benchmark for k-anonymisation

[1] Recall that s in Table 4.1 denotes the size constraint used in pre-partitioning.

Attribute	Domain size	Type
Age	74	Numerical QID
Gender	2	Categorical QID
Race	5	Categorical QID
Salary	2	Categorical QID
Country	41	Categorical QID
Work class	7	Categorical QID
Marital status	7	Categorical SA
Occupation	14	Numerical SA

Table 4.2: Summary of attributes for Adults dataset

algorithms [64, 77, 127, 35, 76, 96]. This dataset contains demographical information of US citizens, and was configured in a way similar to that described in [64], as shown in Table 4.2. *Occupation* was treated as a numerical attribute by mapping semantically close values into consecutive numbers. For a fair comparison, we used the same recoding model for all the methods. Specifically, we recoded values by using ranges and sets for numerical and categorical QIDs respectively. We also removed tuples with missing values, leaving 30162 tuples in the dataset. Furthermore, we used a normally distributed 8-dimensional synthetic dataset with 100000 tuples, which was generated using the Apache Commons-Math library [2]. The attributes were integers with values in the range $[0, 19]$. 6 attributes were treated as QIDs and the remaining as SAs.

4.2 Sample-based Heuristics Evaluation

In this section, we investigate the effectiveness of our sample-based heuristics (see Section 3.3.1). This was done by measuring the relative difference in quality between the anonymisations produced by our MRG method with our heuristic method and

[2] http://commons.apache.org/math/

with an "optimal" threshold setup. For the optimal threshold setup, we configured parameters empirically by trying a large number of different values and then chose one that resulted in the best quality (i.e. minimum sum of UM and MPM scores). In order to be able to control data characteristics, we used synthetic datasets in this set of experiments. We note however that results of using our heuristics on the benchmarking dataset will also be demonstrated later in Sections 4.4 and 4.5.

Evaluating the selection heuristic for δ

In order to examine the impact of our heuristic for choosing the δ value on anonymisation quality, we measured the difference in quality between anonymisations generated by applying MRG on the entire dataset when δ was set "optimally" and "heuristically" as described above, and then measured the *relative quality loss*, defined as

$$RQL = \frac{|(u_T + p_T) - (u_S + p_S)|}{u_T + p_T}$$

where u_T, p_T and u_S, p_S are the UM and MPM scores obtained from applying MRG with the optimal and heuristic δ threshold respectively. Intuitively, RQL indicates percentage of quality decrease from the optimal. We measured RQL with respect to the sample size, data dispersion (measured using standard deviation), k and number of iterations. Default values for these parameters were 2% of dataset, standard deviation $\sigma = 2.5$, $k = 5$ and 3 iterations respectively.

We first studied the effect of sample size on the efficiency of our heuristic. We generated a set of samples from our synthetic dataset with various sizes ranging from $\sqrt{n}$ to 10%, where n is the cardinality of the synthetic dataset. The results are illustrated in Table 4.3. RQL was particularly low, even for a sample of $\sqrt{n}$ tuples. Hence, the proposed heuristic can find a good δ using a small sample. Obviously, small samples are good for execution efficiency.

	Sample size				
	$\sqrt{n}$	2%	5%	7.5%	10%
RQL	0	0.04	0.04	0	0.04

Table 4.3: RQL w.r.t. sample size.

Then, we examined the impact of data dispersion by varying the standard deviation of the dataset. As can be seen in Table 4.4, increasing dispersion did not substantially affect the quality. For this experiment we used a 2% sample of the dataset, $k = 5$ and 3 iterations. The ability of our heuristic to handle data dispersion was also verified using the Adults dataset in our experiments, which achieved a low RQL of 0.043.

	Standard deviation			
	0.5	1	2.5	5
RQL	0	0	0.041	7.62E-4

Table 4.4: RQL w.r.t. data dispersion.

We also ran MRG with various values of k, to investigate how cluster size affects quality. Table 4.5 illustrates RQL for several values of k. Our heuristic achieved a negligible loss in quality, even when clusters were very small. This is because most clusters were of similar size and diameter and thus were well represented in the sample.

	Minimum cluster size k				
	2	5	10	50	100
RQL	1.47E-5	0	1.47E-5	0	0.0075

Table 4.5: RQL w.r.t. k.

The effect of the number of iterations used to derive the δ value from the sample was examined as well, by varying the maximum allowed number of iterations in Algorithm 3. As depicted in Table 4.6, our heuristic found a δ equal to the optimal when this parameter was set to 6, and a good enough result was obtained when 3 iterations were used. This again suggests that our heuristic is efficient, as only a small number of δ values need to be checked.

	Number of iterations				
	2	3	4	5	6
RQL	0.041	2.99E-5	2.99E-5	2.99E-5	0

Table 4.6: RQL w.r.t. number of iterations.

Evaluating the selection heuristic for s

The effectiveness of our heuristic for setting up the size threshold s used in Algorithm 2 was also tested. Recall that this heuristic implies applying MRG on the same sample used to select δ many times. The first time s is set to k, and each time the value of s is doubled until quality converges. Table 4.7 reports the RQL scores between setting up s optimally (by applying the aforementioned process on the entire dataset) and heuristically on the sample. Each column of the table includes RQL values after a number of iterations for a specific minimum cluster size k. Since RQL values were particularly low, our heuristic can be considered as capable of accurately finding a good value for s. We also note that quality converged after 4 iterations in all cases, suggesting that selecting s using this heuristic can be done efficiently.

Minimum cluster size k					
It.	2	5	10	50	100
1	2.53E-4	0.0169	0.0257	0.0551	0.0556
2	2.57E-4	0.0188	0.0260	0.0413	0.0552
3	2.49E-4	0.0116	0.0218	0.0402	0.0505
4	0	4.27E-4	0.0106	0.0221	0.0204
5	0	4.23E-4	0.0106	0.0221	0.0204

Table 4.7: RQL w.r.t. k in selecting s

4.3 Efficiency Evaluation

We also evaluate the runtime performance of Mondrian, K-Members and MRG (with and without the pre-partitioning step). We do not show the result for K-Members-p in this experiment, as its performance was identical to K-Members because these two algorithms differ only in the measure they optimise. For this experiment, we used random samples of the Adults dataset with a cardinality n ranging from 500 to 10000, $k = 5$, and $w_u = w_p = 0.5$. The efficiency of the algorithms is mainly determined by the cardinality of the dataset, and not by the overhead of metric computation [35, 77, 127]. Thus, different datasets and values of k, w_u and w_p could have been used, but they would not have affected the results of the comparison significantly. Parameters δ and s were set using our sample-based heuristics. A 2.5% sample was used and quality converged after 3 iterations when setting s.

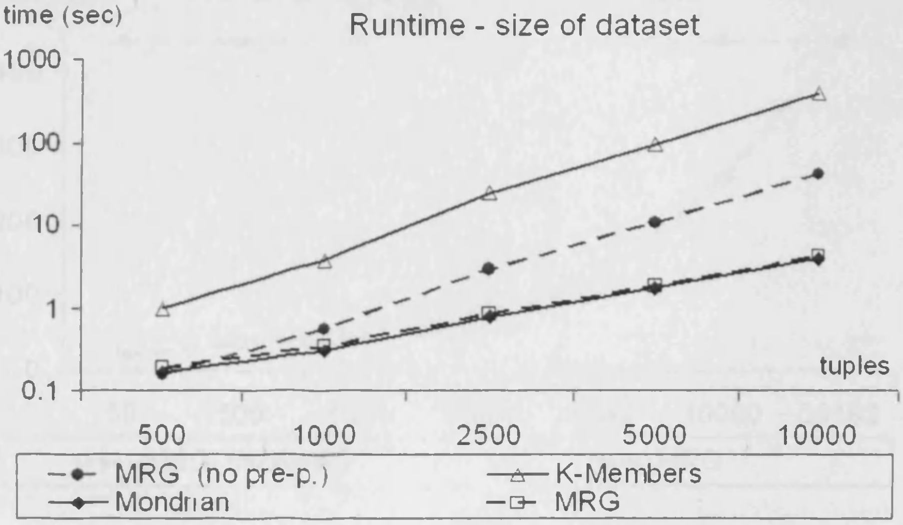


Figure 4.1: Runtime vs. size of dataset

As illustrated in Figure 4.1, Mondrian is faster by at least one order of magnitude than the methods which use clustering-based heuristics (i.e. K-Members and MRG without pre-partitioning). This is because, in contrast to these algorithms which have a quadratic time complexity, the complexity of Mondrian is log-linear with respect to the cardinality of the dataset. However, MRG without pre-partitioning is at least 6 times faster than K-Members. This is attributed to two factors. First, our method uses sorting to speed up cluster formation, having a time-complexity of $O(\frac{n^2}{|c|} \times log(n))$, which is better than the $O(n^2)$ complexity of K-Members when $|c| > log(n)$. Second, seeds in our method are selected randomly rather than based on a furthest-first heuristic as in K-Members. In fact, K-Members requires $O(n \times f)$ time to select seeds [35, 22], where f is the number of seeds, which is not negligible when the number of seeds is comparable to the cardinality of the dataset as in this experiment. Finally, our MRG method with the pre-partitioning step is only 8% slower than Mondrian.

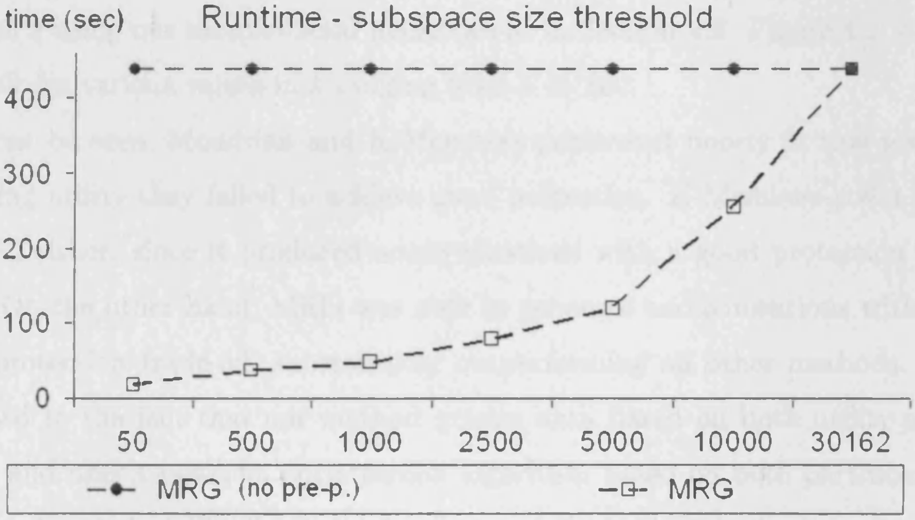


Figure 4.2: Runtime of MRG vs. size threshold s

To study the improvement in efficiency that the pre-partitioning step brings to our method, we applied MRG to the Adults dataset (30162 tuples) setting k to 10. In this experiment, the number of subspaces created by pre-partitioning was reduced in each execution and varied from 512 (size constraint s is set to 50) to 1 (size constraint s is set to 30162). Figure 4.2 shows that the efficiency improvement is significant when pre-partitioning is used. For instance, pre-partitioning reduced the runtime of our method from 7.5 minutes to 18 seconds when s was set to 50.

4.4 Effectiveness of Utility/Protection Trade-off

We then examined whether MRG is able to produce anonymisations with a good utility/protection trade-off, as it is captured by the weighted average of UM and MPM scores. The individual scores achieved in these measures will be studied later in Section 4.5.1. For this experiment, we set the weights w_u and w_p to 0.5, assuming that anonymisers treat utility and protection as equally important. Furthermore, we set δ and s using our sample-based heuristics as in Section 4.3. Figure 4.3 illustrates the result for various values in k ranging from 5 to 100.

As can be seen, Mondrian and K-Members performed poorly in this test, as by optimising utility they failed to achieve good protection. K-Members-p did not perform well either, since it produced anonymisations with a good protection but low utility. On the other hand, MRG was able to generate anonymisations with a good utility/protection trade-off, substantially outperforming all other methods. This is attributed to the fact that our method groups data based on both utility and protection, and uses a powerful optimisation algorithm based on both partitioning and clustering heuristics. Finally, it can be observed that using pre-partitioning did not significantly affect the quality of anonymisations. This is because partitioning and clustering heuristics are combined in a way that the improvement in efficiency brought by partitioning does not harm the quality of anonymisations constructed by clustering.

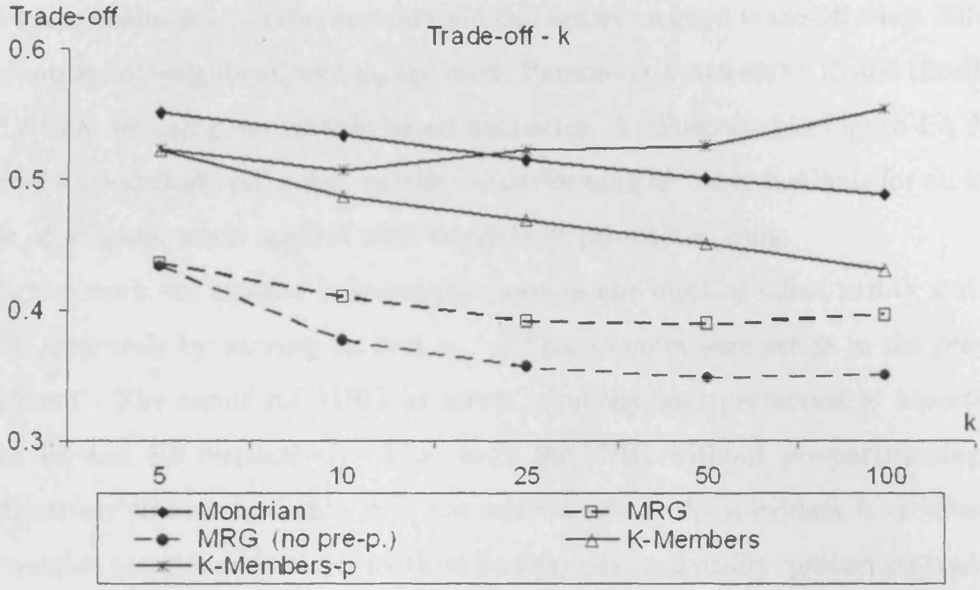


Figure 4.3: Utility/protection trade-off for equal w_u and w_p values

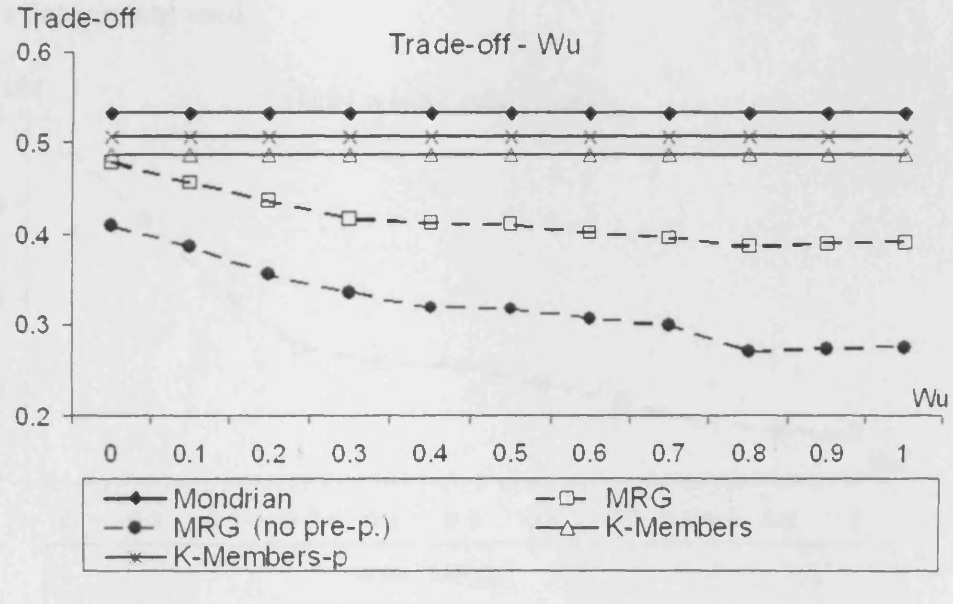


Figure 4.4: Utility/protection trade-off for various w_u and w_p values

We also examined whether our method can achieve a good trade-off when different combinations of weights w_u and w_p are used. Parameter k was set to 10 and thresholds s and δ were set using our sample-based heuristics. As illustrated in Figure 4.4, MRG achieved a good trade-off, substantially outperforming all other methods for all tested values of weights, when applied with or without pre-partitioning.

Furthermore, we studied how weights used in our method affect utility and protection *separately* by varying w_u and w_p. All parameters were set as in the previous experiment. The result for MRG in terms of utility and protection is reported in Figure 4.5 and 4.6 respectively. The result for MRG without pre-partitioning was quantitatively similar, and thus it is not reported here. As is evident from these figures, weights are able to lead our method finding a desired utility/protection trade-off. The same experiment was also repeated using the synthetic dataset. The results reported in Figures 4.7 and 4.8 were qualitatively similar to those derived when the Adults dataset was used.

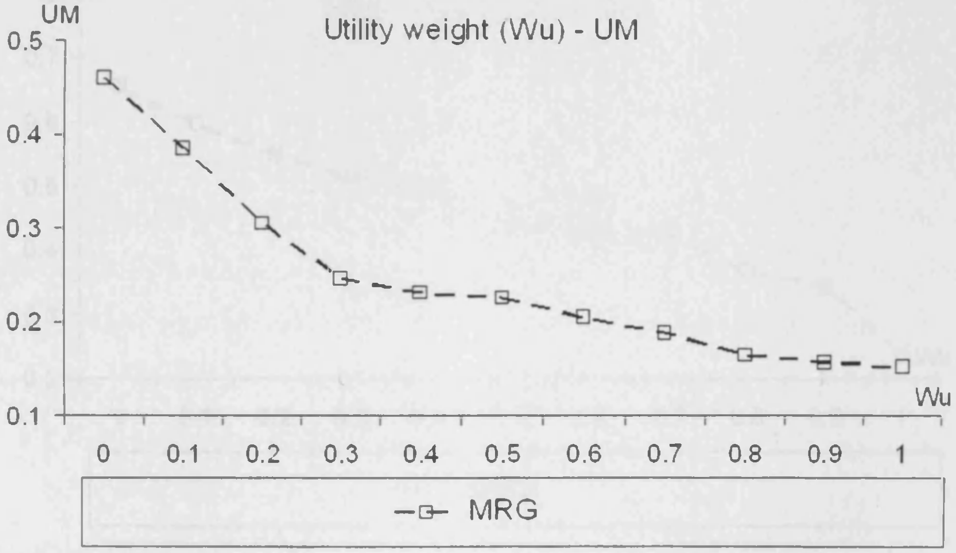


Figure 4.5: The impact of weights on utility (Adults dataset)

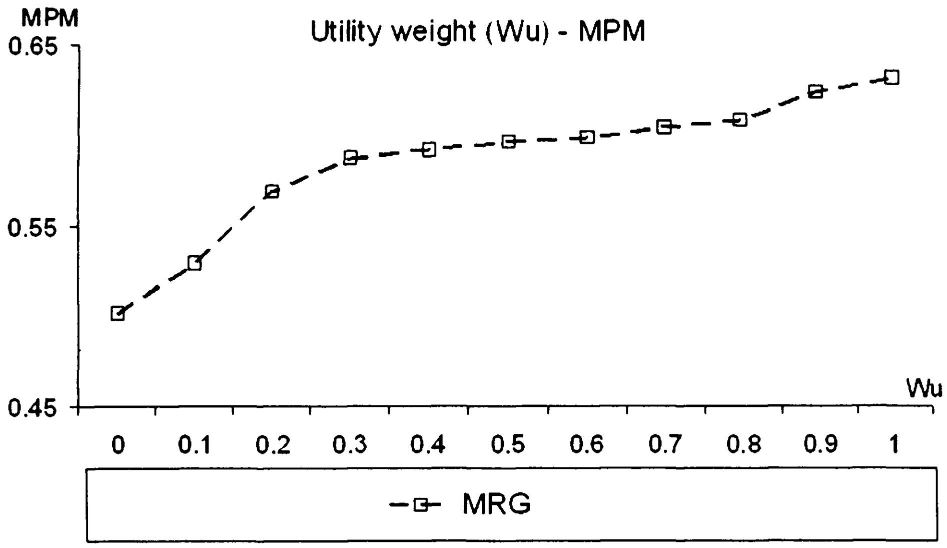


Figure 4.6: The impact of weights on protection (Adults dataset)

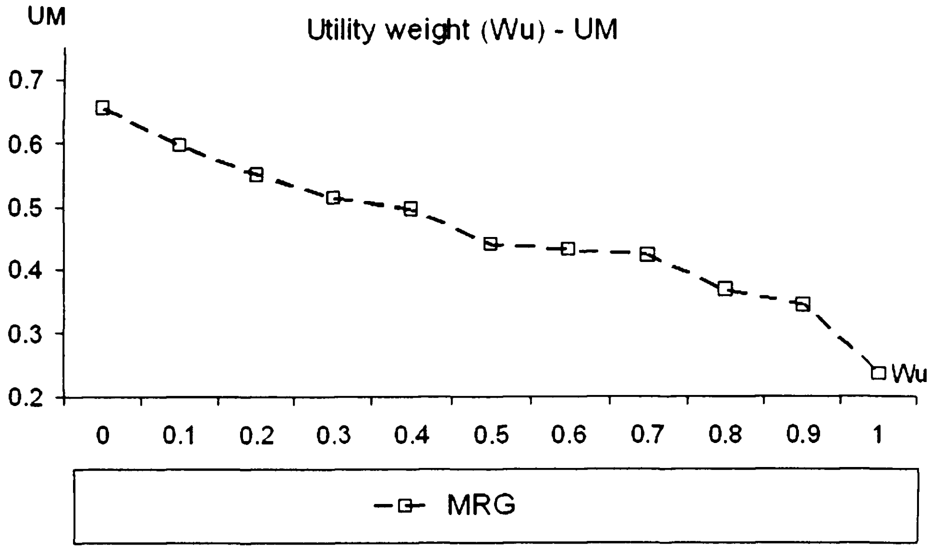


Figure 4.7: The impact of weights on utility (synthetic dataset)

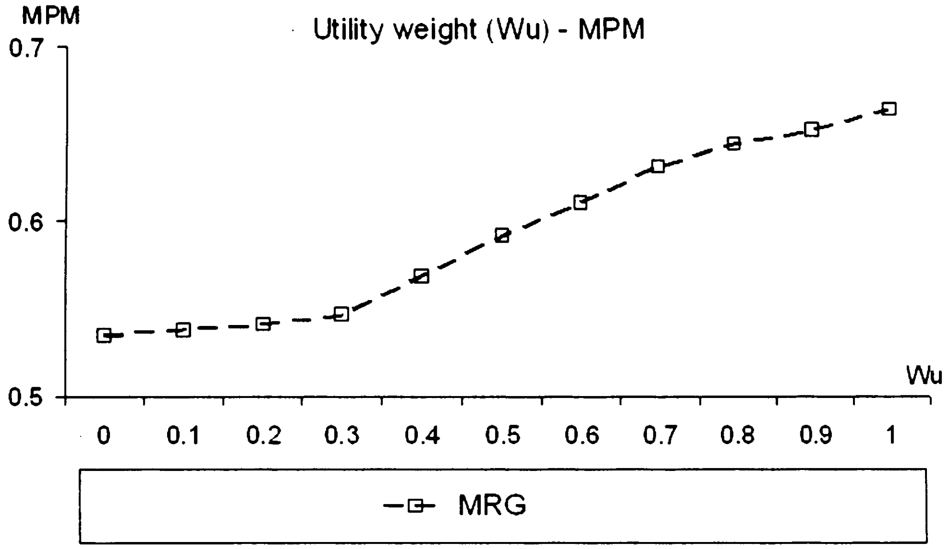


Figure 4.8: The impact of weights on protection (synthetic dataset)

4.5 Utility and Protection Evaluation

We investigated the quality of anonymisations derived by MRG in terms of utility and protection separately. In Section 4.5.1, we captured utility and protection using three measures: UM, DM and MPM. We chose to use both DM and UM, as they capture data utility in fundamentally different ways; DM is based on group size and UM on reconstruction probability. Utility measures based on DGH height are not used, since they cannot be applied to the Adults dataset. This is because QIDs in Adults do not have associated DGHs. We also note that protection criteria, such as l-diversity [85], t-closeness [80] and the measure used in [124] are not applicable, since none of them can handle categorical and numerical SAs simultaneously. For completeness, we also used task-based criteria to capture quality in Section 4.5.2. We used aggregate query answering as an indicative task and measured the utility following [77]. We did not use the CM measure, as we are not specifically interested in classification accuracy.

4.5.1 Effectiveness w.r.t. standard metrics

Figures 4.9 and 4.10 present the results for the UM and DM measures respectively, for Mondrian, K-Members and our method with and without pre-partitioning. The weights w_u, w_p were set to 0.5 and the thresholds δ and s were set as in the previous experiment. The result for K-Members-p was not included in these figures, as it was substantially worse than that achieved by all other methods. As illustrated in Figure 4.9, MRG substantially outperformed Mondrian w.r.t. the UM measure. For instance, the UM score achieved by our method for $k = 25$, is comparable to that of Mondrian when it ran with $k = 5$. Figure 4.10 shows that MRG outperformed Mondrian using the DM measure as well although by a small margin. This is because Mondrian specifically restricts the size of the group to $2 \times k - 1$. Furthermore, the scores for MRG in terms of both UM and DM were close to that achieved by K-Members.

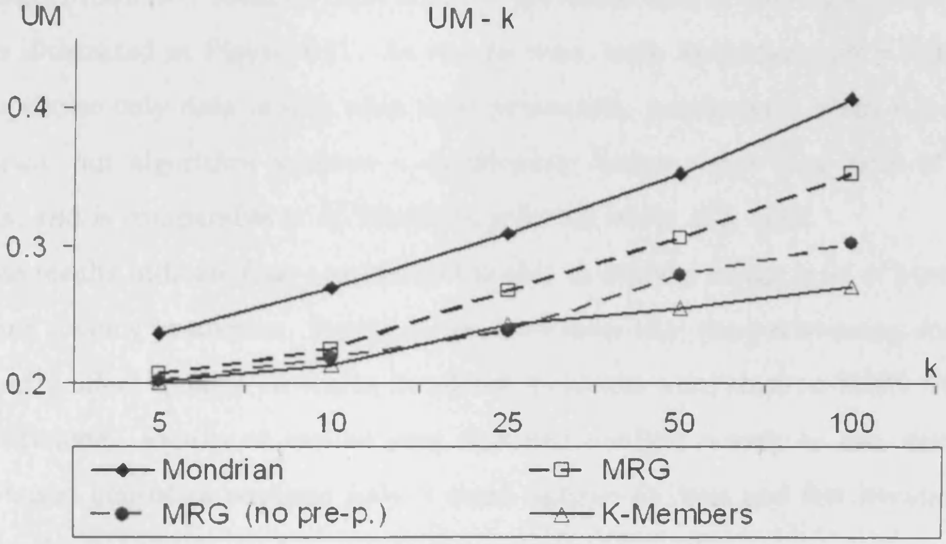


Figure 4.9: UM comparison

We also studied the level of protection offered by the MRG method using the

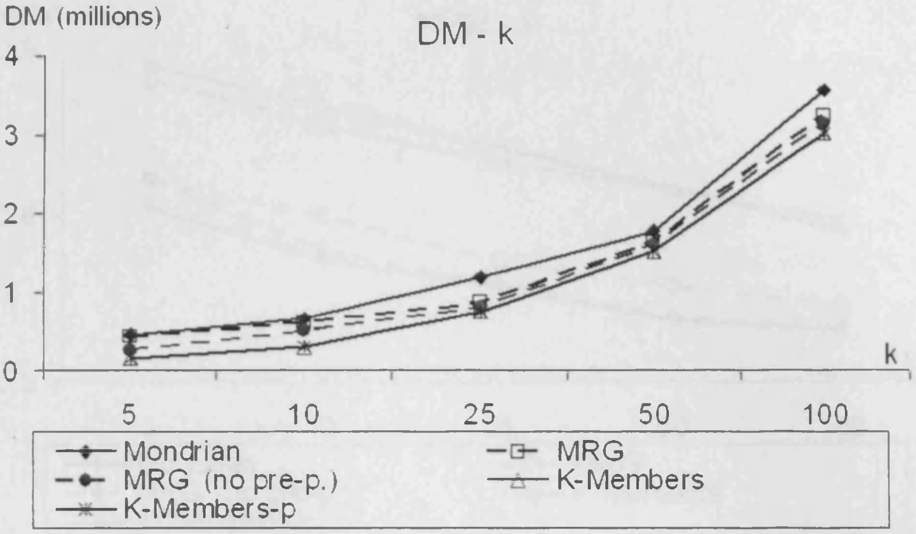


Figure 4.10: DM comparison

Max-range Protection Measure (MPM), for the same run of the algorithms. The result is illustrated in Figure 4.11. As can be seen, both Mondrian and K-Member, which optimise only data utility, offer little protection, particularly when k is small. In contrast, our algorithm achieves a significantly better result than both of these methods, and is comparable to K-Members-p for all levels of k used.

These results indicate that our method is able to achieve a high level of both data utility and privacy protection. Furthermore, they show that pre-partitioning does not significantly affect quality, as scores in all the measures were close to MRG without pre-partitioning. Finally, it can be seen that our method is easy to use, since our sample-based heuristics required only a small sample of data and few iterations to configure the thresholds used in our method reasonably well.

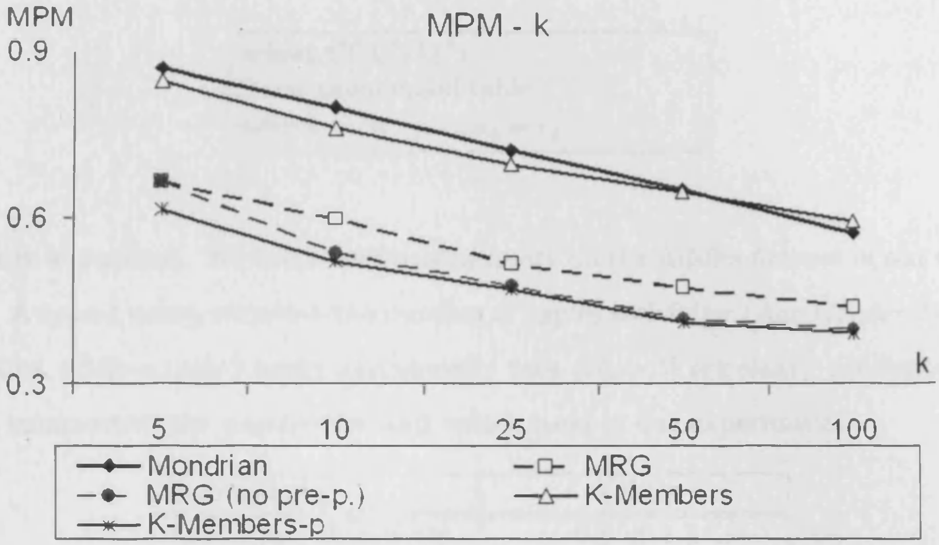


Figure 4.11: MPM comparison

4.5.2 Aggregate query answering accuracy

We now demonstrate the performance of MRG using query answering as an indicative application. To examine the effect of balancing utility and protection, we configured our method using i) equal weights for utility and protection ($w_u = w_p = 0.5$), ii) ($w_u = 1, w_p = 0$) for maximum utility, and iii) ($w_u = 0, w_p = 1$) for maximum protection. We refer to these configurations as MRGB, MRGU and MRGP respectively. MRGU optimises utility, as Mondrian and K-Members do, whereas MRGP optimises protection, as K-Members-p does. MRGB treats data utility and privacy protection as equally important. Other configurations are possible, but are not reported here, as the impact of weights on query answering is similar to that reported in Section 4.4. We considered COUNT-queries for our experiments following [77, 124, 123, 71]. Each of these queries has the following form.

Attributes $a_1, ..., a_d$ are QIDs, and the number of attributes used in a query is controlled by parameter d (query dimensionality) and their values $r_1, ..., r_d$ are selected

```
select COUNT(*)
from anonymised table
where a_1 = r_1, ..., a_d = r_d
```

uniformly at random. We used two types of query on the Adults dataset in our experiments. A type-1 query retrieves the number of tuples satisfying $\{Age, Gender, Income\}$ predicates, while a type-2 query additionally uses $\{Race, Work\ class\}$ predicates. Table 4.8 summarises the parameters and values used in our experiments.

Parameters	Values
Estimated Selectivity	0.5 or 1.5
Query Dimensionality	3 or 5
Query Workload Size	10000
Predicate Selection	uniformly at random

Table 4.8: Parameters and values used in query answering experiments

Given a workload of queries $W = \{q_1, ..., q_l\}$, a table T and its anonymisation T^*, we capture query answering accuracy by computing the average relative error (A.R.E.) [124]. Let $act(q_i), est(q_i)$ be the actual and estimated result derived when a query $q_i \in W$ is applied on T and T^* respectively. The relative error is defined by $\frac{|act(q_i)-est(q_i)|}{act(q_i)}$. To derive $est(q_i)$, we compute, for each tuple $t \in T^*$, the probability p that t satisfies q_i using $p = \frac{R_{q_i} \cap R}{R}$, where R_{q_i} and R denote the areas covered by q_i and the qualifying tuples in T^* respectively, and $R_{q_i} \cap R$ denotes the overlap between R_{q_i} and R. After computing the probability p, $est(q_i)$ is set to the sum of such probabilities of all tuples. Then, the A.R.E. score for table T^* w.r.t. the workload W can be computed as

$$ARE(W, T, T') = \frac{1}{l} \times \sum_{i=1}^{l} \frac{|act(q_i) - est(q_i)|}{act(q_i)}$$

Figure 4.12 reports the *normalised* A.R.E. for a workload containing 10000 type-1 queries, for various k levels, for all the methods except K-Members-p (its performance in query answering was significantly worse than that of all other methods, hence not included here) [3]. The normalised A.R.E. is defined to be $\frac{x-b}{a-b}$, where x is the A.R.E. achieved by a method, and a, b are the best and worst A.R.E. values achieved for a specific k. As can be seen, all configurations of MRG outperformed Mondrian and K-Members when k was up to 15. This result confirms that our method does not incur an excessive amount of information loss, which is largely attributed to the quality-based stopping criterion used. This is in contrast to both Mondrian and K-Members, which can form groups containing $2 \times k - 1$ tuples at most. As result, these methods may exclude some "close" tuples from a group to satisfy size requirement, thereby increasing information loss. For k larger than 15, the performance of MRGB was slightly worse than Mondrian, but still better than K-Members, while MRGP performed poorly in this test. This is because MRGP traded-off some utility for increased protection. It is also worth noting that the normalised ARE for MRGB and MRG without pre-partitioning were similar, confirming that pre-partitioning does not significantly affect anonymisation quality.

For type-2 queries, which are more difficult to be accurately answered than type-1 queries due to the higher dimensionality and lower selectivity, all configurations of our algorithm outperformed both Mondrian and K-members, as illustrated in Figure 4.13 [4]. This is because using clustering with a quality-based stopping criterion can preserve correlations between QIDs better, which minimises the level of information loss that is incurred to generalise QID values.

[3] The zero scores in normalised A.R.E. are not shown in Figure 4.12.

[4] The zero scores in normalised A.R.E. are not shown in Figure 4.13.

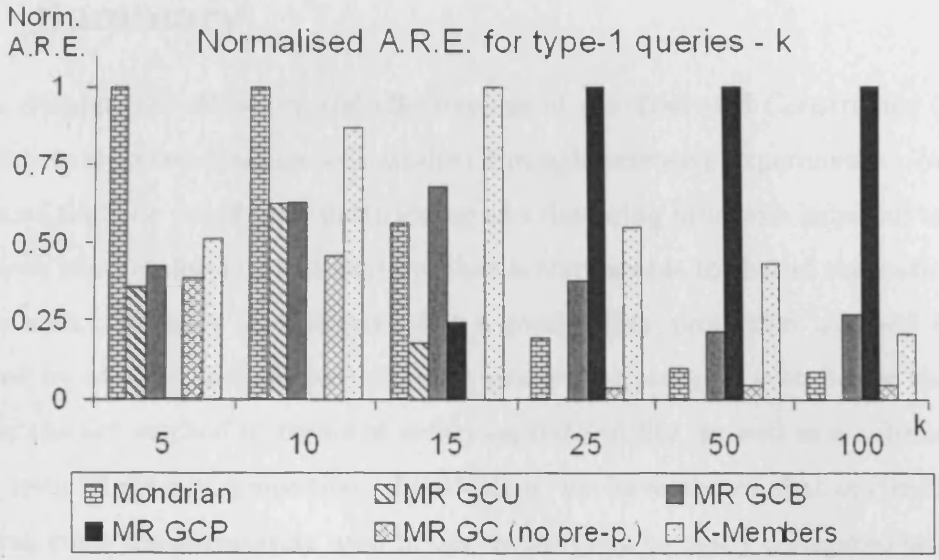


Figure 4.12: Normalised A.R.E. for type-1 queries

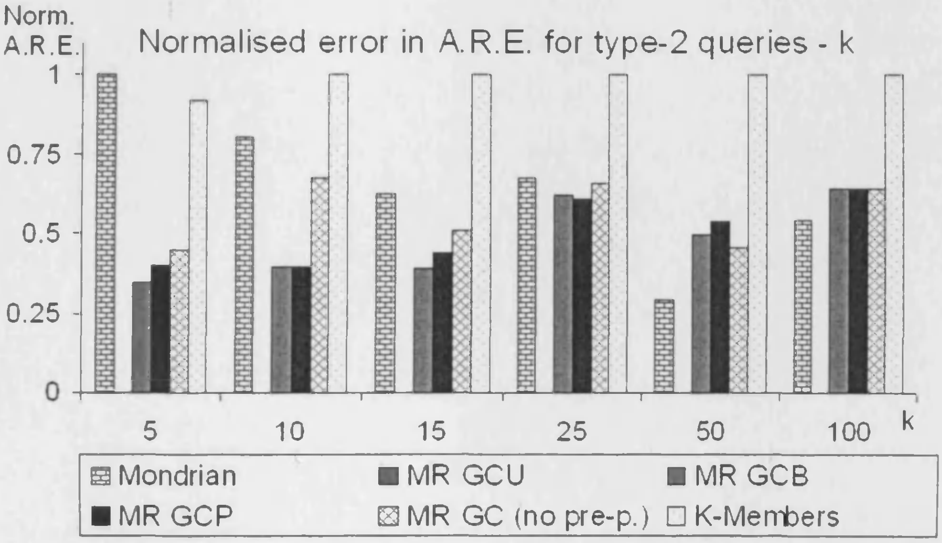


Figure 4.13: Normalised A.R.E. for type-2 queries

4.6 Summary

In this chapter, the efficiency and effectiveness of our Trade-Off Constrained (TOC) approach to k-anonymisation was studied through extensive experiments. We have confirmed that the use of both partitioning and clustering heuristics helps our method to achieve good scalability and runtime that is comparable to that of the state of the art methods. We have also showed that a good utility/protection trade-off can be achieved by our method. In particular, it performs equally or even better than the state of the art method in terms of achieving data utility, as well as a substantially better level of privacy protection. In addition, we have shown that our method is practical, since the parameters used in our method can be easily configured using our sample-based heuristics.

Chapter 5

Range Disclosure and Quality Guarantees

In Chapter 3, we introduced a Trade-off Constrained (TOC) method which uses uniformly formulated criteria for capturing data utility and privacy protection, and is capable of producing anonymisations with balanced utility and protection. This method however has two limitations. First, it may still allow sensitive information to be disclosed in the form of ranges. This is because it attempts to create groups that have a large maximum range, but ignores how the values are distributed within the group. When a group contains a large number of similar SA values, an attacker can still infer a sensitive range for an individual, even though the maximum range of this group is large. Second, the TOC method cannot guarantee that anonymised data will have a minimal required level of both data utility and protection. This is because groups incurring excessive information loss or containing badly protected data can still be created, even when their utility/protection trade-off is acceptable. This may not be acceptable in practice.

This chapter addresses these two issues. In Section 5.1, we analyse the problem of range disclosure in k-anonymised data and present our protection mechanism. Section

5.2 discusses heuristics and algorithms that can guarantee a minimal level of utility and protection required when anonymising data.

5.1 Range Disclosure

To guard against the problem of range disclosure, anonymised data should make the inference of any sensitive range for an individual "sufficiently" difficult. Note that this protection requirement cannot be met by using our MPM measure, the mean squared error [78], or t-closeness [80]. For example, if we have $\{t_1, t_2, t_6, t_{11}\}$ in Table 5.1 as an anonymised group, then the interval is sufficiently large ($[1K - 90K]$), but it is still possible to infer *low income* for any individual represented in this group with a probability of $\frac{3}{4}$. If the requirement is such that no one should be able to infer that someone has a low income with a probability higher than 0.5, then this is not acceptable.

The personalised anonymity approach proposed by Xiao and Tao [124] can achieve protection for range disclosure. It allows individual range disclosure protection requirements to be specified, and met by generalising SA values when necessary. For instance, if the requirement is that no one should be able to infer whether an individual has *low* or *high income* with a probability higher than 0.5 when the data in Table 5.1 is 3-anonymised, their method will produce Table 5.2. This offers guarantees for required range disclosure protection, but may generate data that has an unacceptably low level of utility. In Table 5.2, for example, some income values have been generalised completely, i.e. replaced by $[0 - 99]K$, the most general value in the hierarchy given in Figure 5.1. This may render the anonymised data practically useless.

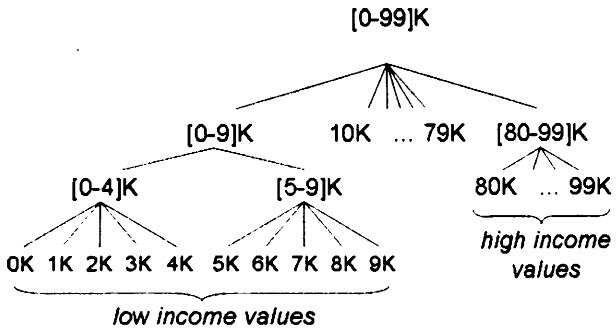


Figure 5.1: Hierarchy for the *Income* attribute

Id	*Age*	*Income*
t_1	30	1K
t_2	22	4K
t_3	29	22K
t_4	29	23K
t_5	31	24K
t_6	35	9K
t_7	37	0K
t_8	39	9K
t_9	41	25K
t_{10}	42	26K
t_{11}	44	90K
t_{12}	45	90K

Table 5.1: Original microdata

Id	*Age*	*Income*
t_1	[20-30]	[0-99]K
t_2	[20-30]	[0-99]K
t_3	[20-30]	[0-99]K
t_4	[20-30]	[0-99]K
t_5	[30-40]	[0-99]K
t_6	[30-40]	[0-99]K
t_7	[30-40]	[0-99]K
t_8	[30-40]	[0-99]K
t_9	[41-45]	25K
t_{10}	[41-45]	26K
t_{11}	[41-45]	90K
t_{12}	[41-45]	90K

Table 5.2: Anonymised data using *Personalised anonymity*

In this thesis, a criterion called *Range Disclosure* that can quantify protection against range disclosure is proposed. In what follows, we first discuss range disclosure and then describe this new criterion.

Let $T(a_1, ..., a_m, sa)$ be a table containing microdata, where $a_1, ..., a_m$ are QIDs and sa is an SA. Multiple SAs are possible, but present additional challenges that are beyond the focus of this thesis (see [80] for an extended discussion). We assume that each tuple in T represents only one individual and takes a value from a finite domain D_{a_i} (D_{sa}) in each a_i (sa). If a_i (sa) is categorical, we assume that there is a domain hierarchy H_{a_i} (H_{sa}) associated with it, similar to the one shown in Figure 5.1. Domain hierarchies are specified by the anonymiser.

Central to our notion of range protection is the concept of sensitive range.

Definition 5.1.1 (Sensitive Range). *Given a value u in sa, the* Sensitive Range *for u, denoted by $SR(u)$, is defined as:*

- *an interval $[w, v]$ s.t. $w \leq u \leq v$ and $w, v \in D_{sa}$, if sa is numerical, or*
- *a node on the path from the root to u in H_{sa}, if sa is categorical.*

So, for a numerical value, its sensitive range is simply an interval containing it, and for a categorical value, its sensitive range is a more general concept for this value according to the hierarchy H_{sa}. For example, the sensitive range for the value $1K$ in the categorical SA *Income* in Table 5.1 can be $[1K, 1K]$, $[0K, 4K]$, $[0K, 9K]$ or $[0K, 99K]$, i.e. any node on the path from the root ($[0-99]K$) to the leaf ($1K$) with respect to the hierarchy shown in Figure 5.1. In contrast, if *Income* is considered as a numerical attribute, the sensitive range for this value can be any interval in its domain containing the value $1K$, e.g. $[0K, 7K]$ or $[1K, 99K]$.

Given T, an anonymiser can specify what range information should not be disclosed about each individual, by setting a minimum sensitive range around the SA value associated with the individual. We assume that this is done by anonymisers using a policy that has been decided before anonymisation [85, 80], e.g. the sensitive range for every distinct value in a numerical sa to have a certain length.

Given a k-anonymised table T^* two types of range attack are possible. The first involves an attacker attempting to identify a sensitive range for an individual with high probability by using T^* only. The second involves an attacker attempting to infer the range information associated with an individual by using additional background knowledge to eliminate the ranges that are known to be irrelevant to this individual. These cases are referred to as positive and negative range disclosure respectively, and are analogous to those studied for the value disclosure problem in [85]. We illustrate them using the following example.

Assume that an attacker uses the 6-anonymisation of Table 5.1 given in Table 5.3 and attempts to infer range information for t_2. The sensitive ranges for values $0K, 1K, 4K, 9K$ and $90K$ denoting *low* and *high* income respectively are set to their closest ascendants shown in the hierarchy in Figure 5.1, while for other values their sensitive ranges are set to be the values themselves. Using Table 5.3 and through positive disclosure attack, an attacker may identify the sensitive range $[0K, 9K]$ for t_2 with a probability of $\frac{2}{6}$. Furthermore, using the same table and through negative disclosure attack, an attacker can infer the sensitive range of t_2 with an increased probability of $\frac{2}{4}$ if he/she knows that the individual represented by t_2 does not have a high income (i.e. in the range $[80K, 99K]$), since this knowledge allows tuples t_{11} and t_{12} in Table 5.3 to be excluded from his/her consideration.

It is worth observing that range disclosure is a more general case than the value disclosure problem we discussed in Chapter 3. In fact, value disclosure becomes range disclosure if for every value u in a categorical *sa* we set the sensitive range $SR(u)$ to the value itself. This implies that by preventing a sensitive range from being disclosed, we effectively prevent any value in this range from being disclosed as well. However, as in the value disclosure case, it may be impossible to release data when an arbitrary amount of adversarial knowledge is assumed [85]. Thus, given a table T, we assume that a data publisher is willing to release a k-anonymisation T^* of T when the risk of associating each tuple $t \in T^*$ to $SR(u)$ is lower than a required level.

Id	*Age*	*Income*
t_1	[22-45]	1K
t_2	[22-45]	4K
t_6	[22-45]	9K
t_8	[22-45]	9K
t_{11}	[22-45]	90K
t_{12}	[22-45]	90K
t_7	[29-42]	0K
t_3	[29-42]	22K
t_4	[29-42]	23K
t_5	[29-42]	24K
t_9	[29-42]	25K
t_{10}	[29-42]	26K

Table 5.3: A 6-anonymisation of Table 5.1

5.1.1 Range Protection Quantification

We now propose a function that can be used to capture the risk of range disclosure. It works by measuring the amount of protection offered to an individual's tuple t in a k-anonymous group G based on the distribution and semantic distance of SA values included in the sensitive ranges contained in G, and takes into account both positive and/or negative disclosure attacks.

We observe that the amount of protection offered to t is small when these ranges have a large number of semantically close values. This is because the more tuples are associated with one sensitive range, the greater the danger for privacy breach, since they either allow the sensitive range to be established with high probability or to be excluded from attacker's consideration. In addition, it is natural to expect that an attacker with knowledge about a value u is more likely to have knowledge about a value u' that is closely related to u. Thus, if one of these ranges is "narrow" it may be eliminated easily.

To see this, consider *Income* in Table 5.3 as a numerical SA and an attacker who wants to establish how likely the income of an individual represented by t_7 in this table is in the range $[0K, 4K]$. Also assume that the sensitive range for values $22K, 23K, 24K, 25K$ and $26K$ (associated with tuples t_3, t_4, t_5, t_9 and t_{10}) is the interval $[10K, 30K]$. Observe that these tuples are semantically close, so they allow an attacker to eliminate them if he/she knows that an individual's income is not in the range $[22K, 26K]$. This makes it possible for the attacker to infer sensitive range information for t_7.

Before formulating our protection criterion, we define our notion of semantic distance between a pair of values in *sa* using *Pairwise Contribution*, as illustrated in Definition 5.1.2.

Definition 5.1.2 (Pairwise Contribution). *Given a pair of values (u, u') obtained from D_{sa}, the Pairwise Contribution for (u, u') is defined as:*

$$pc(u, u') = \begin{cases} 1 - \frac{|u-u'|}{max(D_{sa})-min(D_{sa})} & \textit{numerical values} \\ 1 - \frac{s(\{u,u'\})-1}{|D_{sa}|-1} & \textit{categorical values} \end{cases}$$

where $max(D_{sa})$ and $min(D_{sa})$ denote maximum and minimum values in D_{sa} respectively, and $|D_{sa}|$ is the size of domain D_{sa}. If there is a hierarchy H_{sa}, then $s(\{u, u'\})$ is the number of leaves of the subtree of H_{sa} rooted at the closest common ancestor of the values u and u'.

Pairwise Contribution is based on the same notion of distance as the *S-diversity* measure (see Definition 3.1.3) and can be computed similarly. For instance, the *pc* score for the values *1K* and *4K* in t_1 and t_2 in the first group of Table 5.3 is $1 - \frac{5-1}{100-1} \approx 0.96$, since the subtree rooted at the closest common ancestor ($[0-4]K$) of these values according to the hierarchy shown in Figure 5.1 has 5 leaves, and there are in total 100 distinct values in the domain of *Income*. If the data publisher treats

Income as a numerical attribute, the *pc* score for these values can be computed as $1 - \frac{4-1}{100-1} \approx 0.96$. Pairwise Contribution penalises semantically close values in a group, since they help prevent range disclosure. For example, when *Income* is treated as a numerical attribute, the *pc* score for the values $1K$ and $90K$ in t_1 and t_{11} in the first group of Table 5.3 is $1 - \frac{90-1}{100-1} \approx 0.1$, which is smaller than 0.96, reflecting the fact that the distance between values $1K$ and $90K$ is larger than that between $1K$ and $4K$.

We also introduce *Disclosure Confidence*, a practical metric to assess the level of risk of associating a tuple t to its sensitive range in a k-anonymous group.

Definition 5.1.3 (Disclosure Confidence (dc)). *Given a k-anonymous group G and a tuple t having a value $u \in G$ w.r.t. sa, the disclosure confidence $dc(u, SR(u))$ is given by*

$$dc(u, SR(u)) = \frac{f(u) \times \sum_{\forall x}(f(x) \times pc(u,x))}{|G|}$$

where x is a value (u included) that lies in $SR(u)$, $f(x)$ and $f(u)$ denote the frequency of x and u in G respectively, $|G|$ is the size of G, and pc is as defined in Definition 5.1.2.

It should be noted that Disclosure Confidence takes into account both the frequency of the SA value u in G, as well as the semantic closeness between u and SA values in other tuples that fall in $SR(u)$. This is because its computation is based on $f(u)$, and the level of semantic similarity between u and any value $x \in G$ in $SR(u)$ captured by $pc(u, x)$. A high score in *dc* implies that many tuples in G whose SA values are semantically close to u will have the same sensitive range as t, and thus t can easily be associated with its sensitive range.

To illustrate the computation of Disclosure Confidence, we evaluate this measure for the income value $22K$ in t_3, which lies in the second group of Table 5.3. We consider *Income* as numerical and assume that the sensitive range for value $22K$ is an interval $[10K, 30K]$. Then, the Disclosure Confidence score can be computed as

$$dc(22K, SR(22K) = [10K - 30K]) =$$

$$= \frac{pc(22K,22K) + pc(22K,23K) + pc(22K,24K) + pc(22K,25K) + pc(22K,26K)}{6} \approx 0.82$$

Furthermore, we make the standard *random worlds* assumption [24, 85]. That is, in the absence of any further knowledge, each t can have any of the sensitive values in G equally likely. Therefore, we measure the level of protection G offers for range disclosure by computing the average level of risk of associating a tuple in G with its sensitive range. We call this measure *Range Diversity.*

Definition 5.1.4 (Range Diversity (rd)). *Given a k-anonymous group G containing distinct values $u_1, ..., u_l$ w.r.t. sa and their sensitive ranges $SR(u_1), ..., SR(u_l)$, the* Range Diversity *of G, denoted by $rd(G)$, is defined as:*

$$rd(G) = \frac{1}{|G|} \times \sum_{i=1}^{l} dc(u_i, SR(u_i))$$

where $dc(u_i, SR(u_i))$ is given in Definition 5.1.3 and $|G|$ denotes the size of G.

Based on Range Diversity, we propose a metric called *Worst-Group Protection* (*WGP*) to capture the amount of protection for the least protected group in T^*. Limiting WGP can naturally be used to ensure that T^* offers a sufficient level of protection for range disclosure.

Definition 5.1.5 (Worst-Group Protection). *Given a k-anonymous table T^* comprised of groups $\{g_1, g_2, \ldots, g_h\}$, the Worst-Group Protection for T^*, denoted with $WGP(T^*)$ is defined as $WGP(T^*) = max(rd(g_1), \ldots, rd(g_h))$.*

5.2 Heuristics and Algorithms For Guaranteeing Quality

As mentioned in the Introduction, there are applications in which some minimal data utility and protection requirements have to be met. Consider for example a pharmaceutical company which requires the anonymised data it gets from a hospital to have a minimal level of data utility, e.g. to allow a certain level of classification accuracy when building classifiers from the data. Since anonymisations with the level of data utility below what is required are practically useless for the company, the hospital will need to observe that this requirement is met. In addition, released data must not give away sensitive information about individuals, according to the stated privacy policy that the hospital promises to its patients. Thus, the hospital requires this data to have a minimal level of protection too.

Unfortunately, existing approaches to k-anonymisation are not able to meet these requirements. The group-size and utility constrained approaches discussed in Section 2.1.3 are inapplicable, since they do not consider protection. The protection constrained approach is not applicable either, since it guarantees the protection but not the utility requirement. The TOC approach presented in Chapter 3 does consider both but provides no guarantees for either.

To generate anonymisations with utility and protection guarantees, we propose a *Utility and Protection Constrained* (UPC) k-anonymisation approach. That is, we require anonymised data to satisfy three constraints: minimum group size k, maximum level of acceptable information loss incurred (U), and minimum level of acceptable protection exercised (P). We develop a method that allows anonymised data with a controlled amount of information loss and protection for range disclosure to be generated. This is achieved by designing a new heuristic, which is incorporated into an algorithm that uses partitioning and clustering in a similar way to our TOC-based method presented in Chapter 3.

5.2.1 UPC k-Anonymisation Heuristics

Before formulating our heuristic, we introduce a modified version of our Utility Measure (UM) given in Definition 3.1.2. This is necessary because the UM and other utility criteria discussed in Section 2.1.1 control the total or the average distortion of the table, not the level of distortion of the most generalised group. On the other hand, we require the level of distortion in individual groups of the anonymised dataset to be limited, so as to ensure that no groups contain data that may violate the utility requirement.

Our alternative criterion is based on *Q-diversity* introduced in Chapter 3, which is reproduced below for convenience of reference.

Definition 5.2.1 (Q-diversity). *Assume that a is a QID, the domain of a is D_a, and $V_a \subseteq D_a$ is a subset of values obtained from a. The* Q-diversity *of V_a, denoted by $Qd(V_a)$, is defined as*

$$qd(V_a) = \begin{cases} \frac{max(V_a)-min(V_a)}{max(D_a)-min(D_a)} & \textit{numerical values} \\ \frac{s(V_a)-1}{|D_a|-1} & \textit{categorical values} \end{cases}$$

where $max(V_a)$, $min(V_a)$, $max(D_a)$ and $min(D_a)$ denote maximum and minimum values in V_a and D_a respectively, and $|D_a|$ is the size of D_a. If a is categorical, then $s(Va)$ is the number of leaves of the subtree of H_a rooted at the closest common ancestor of the values in V_a.

Our modified criterion is called *Worst-Group Utility* (WGU) and is given in Definition 5.2.2. An upper bound on WGU, i.e. the maximum allowed amount of information loss of the most heavily distorted group, can be used to constrain the utility loss of a table.

Definition 5.2.2 (Worst-Group Utility). *Assume that a table T comprised of m QIDs $\{a_1, \ldots, a_m\}$ is clustered into groups $\{g_1, g_2, \ldots, g_h\}$, such that $|g_j| \geq k, 1 \leq j \leq h$, and tuples in each g_j will have the same values in each QID after anonymisation. The worst group utility of T under this clustering is defined as*

$$WGU(T) = max(\frac{1}{m}\sum_{i=1}^{m} qd(\pi_{a_i}(g_1)), \ldots, \frac{1}{m}\sum_{i=1}^{m} qd(\pi_{a_i}(g_h)))$$

where $qd(\pi_{a_i}(g_j))$ denotes the Q-diversity of group g_j w.r.t. a_i, $1 \leq j \leq h$ and $1 \leq i \leq m$.

We now formally define the concept of *UPC-optimal k-anonymisation*, as illustrated in Definition 5.2.3.

Definition 5.2.3 (UPC-optimal k-anonymisation). *Given a table T and two constraints U, P for utility and protection, a partition $\sigma = \{g_1, ..., g_h\}$ of T is a UPC-optimal k-anonymisation if $|g_j| \geq k, j = 1, \ldots, h$, $\bigcap_{j=1}^{h} g_j = \emptyset$, $\bigcup_{j=1}^{h} g_j = T$, and there does not exist a partition ς of T s.t. $WGU(\varsigma) < WGU(\sigma) \leq U$ or $WGP(\varsigma^*) < WGP(\sigma^*) \leq P$, where U and P are in $[0, 1]$, σ^* and ς^* are the k-anonymised versions of σ and ς respectively, and $|g_j|$ denotes the size of g_j.*

A UPC-optimal k-anonymisation is thus a grouping of data with minimal WGU and WGP scores that satisfy some specified utility and protection requirements. Finding a UPC-optimal k-anonymisation requires the generation of Pareto optimal solutions [40], i.e. the k-anonymous partitions which dominate the search space in at least utility or protection and satisfy both constraints. Related problems have been reported in multi-criteria optimisation literature, and are typically solved by consensus clustering techniques [92, 41, 82]. Such methods first find a set of partitions each of which optimises a different optimisation criterion, and then construct a "good" partition with respect to all of the criteria based on these partitions. Consensus clustering techniques have been applied successfully to genomic data integration [45] and to improving traditional clustering techniques [92], but they are not useful to solving our problem, since they assume that the final partition is not "radically different in terms of number of clusters and cluster composition" than the ones that optimise a single criterion [92]. Unfortunately, this is not the case in k-anonymisation, as experiments reported in Sections 4.4 and 4.5 have shown. Therefore, we take a different approach [33, 98]. Instead of combining multiple partitions, we combine our two search objectives into one using the following heuristic.

Definition 5.2.4 (Group Diversity). *Let $\tau \subseteq T$ be a set of tuples over a set of attributes $A = \{a_1, \ldots, a_m, sa\}$, $a_1, \ldots, a_m$ are QIDs and sa is an SA. The* Group Diversity *of τ w.r.t. A, denoted by $gd(\tau, A)$, is defined as:*

$$gd(\tau, A) = \frac{1}{2} \times \sum_{i=1}^{m} \frac{qd(\pi_{a_i}(\tau))}{m} + \frac{1}{2} \times rd(\pi_{sa}(\tau))$$

where $\pi_{a_i}(\tau)$ and $\pi_{sa}(\tau)$ denote the projection of τ on attribute a_i and sa respectively, $qd(\pi_{a_i}(\tau))$ denotes the Q-diversity of τ w.r.t. a_i, and $rd(\pi_{sa}(\tau))$ denotes the Range Diversity of τ w.r.t. sa [1].

The Group Diversity heuristic is similar in principle to the Max-range Weighted Tuple Diversity heuristic (see Definition 3.2.1), but it uses the *Range Diversity* measure providing stronger protection for range disclosure. Using this heuristic, we can define a gd-optimal UPC partition.

[1] We slightly abuse the notation by writing $rd(\pi_{sa}(\tau))$ for a group τ instead of $rd(\pi_{sa}(\tau^*)$, where τ^* is the k-anonymised version of τ. We note that $rd(\pi_{sa}(\tau)) = rd(\pi_{sa}(\tau^*))$, since SA values are not perturbed after k-anonymisation.

Definition 5.2.5 (Gd-Optimal UPC Partition). *Let T be a table consisting of a set of attributes $A = \{a_1, \ldots, a_m, sa\}$, and U and P be specified utility and protection constraints. A* gd-optimal UPC partition *of T is a partition $\sigma = \{g_1, \ldots, g_h\}$ such that $|g_j| \geq k, j = 1, \ldots, h$, $\bigcap_{j=1}^{h} g_j = \emptyset$, $\bigcup_{j=1}^{h} g_j = T$, $WGP(\sigma) \leq U$, $WGP(\sigma^*) \leq P$, and $gd\_max(\sigma, A) = max_{j=1,\ldots,h}(gd(g_j, A))$ is minimal, where σ^* is the k-anonymised version of σ.*

Thus, a gd-optimal UPC partition is a grouping of data with a minimal gd score and satisfies utility and protection requirements. However, finding such a partition is NP-hard, as shown in Theorem 5.2.1.

Theorem 5.2.1. *Finding a Gd-optimal UPC partition is NP-hard.*

Proof. We observe that the problem considered by Aggrawal et al. [19] is a special case of our problem, where utility and protection constraints U, P are set to the maximum allowed level (i.e. $U = 1$ and P=1), QID attributes are categorical and their hierarchies are as considered in Theorem 3.2.1. □

5.2.2 UPC k-Anonymisation Algorithm

We now present an algorithm for deriving a UPC k-anonymisation heuristically. Our solution employs partitioning and clustering similarly to the one given in Section 3.2, but it differs from it in a number of ways.

The method illustrated in Algorithm 6 works in three steps. First, the entire dataset is partitioned into a set of subspaces (step 1), which is the same as the one presented in Chapter 3 (see Algorithm 2). Second, for each subspace $\phi \in \Phi$, Algorithm 6 performs the following steps. It randomly picks up a tuple t_i as the *seed* of a cluster and removes it from ϕ in step 4. In steps 5-10, it repeatedly finds the closest tuples to this cluster, one at time, and adds them into the cluster while utility and protection constraints are satisfied. We find a tuple t_j that is closest to the cluster in step 6

Algorithm 6 Greedy method with Group Diversity heuristic

```
1.  $\Phi \leftarrow$ MBP($T, s$);
2.  for each subspace $\phi \in \Phi$ do
3.    while $\phi \neq \emptyset$ do
4.      $c \leftarrow t_i \in \phi$; $\phi \leftarrow \phi - \{t_i\}$;
5.      while true do
6.        find $t_j \in \phi$ s.t. $gd(\{c \cup \{t_j\}\}, A)$ is minimum;
7.        $c' \leftarrow c \cup \{t_j\}$;
8.        if $((|c'| < k$ and $gd(c', QID) \leq U)$
             or $(gd(c', QID) \leq U$ and $gd(c', SA) \leq P))$
9.          $c \leftarrow c'$; $\phi \leftarrow \phi - \{t_j\}$;
10.       else exit;
11.   if $|c| \geq k$
12.     recode($c$);
13.     $S \leftarrow S \cup c$;
14.   else
15.     for each $t_i \in c$ do;
16.       $O \leftarrow O \cup \{t_i\}$;
17. for each $t_i \in O$ do
18.   find $c_j \in S$ s.t. $c'_j \leftarrow c_j \cup \{t_i\}$ and $gd(c'_j, QID) \leq U$
      and $gd(c'_j, SA) \leq P$ and $gd(c'_j, A)$ is minimum;
19.   if $c_j$ does not exist
20.     suppress $t_i$;
```

by computing $gd(\{c \cup \{t_j\}\}, A)$ according to Definition 5.2.4. In steps 8 and 9, t_j is temporarily added into the cluster and two conditions are checked: i) the cluster size is smaller than k and the utility constraint U is satisfied, or ii) both constraints U and P are satisfied. If either of these two conditions holds, we add t_j into the cluster and remove it from T (step 9). Otherwise, we stop extending the cluster (step 10). If condition (i) fails, there is no need to extend the cluster, as this cannot help utility. This is because $gd(c', QID)$ can only increase as a result of extending the cluster. If condition (ii) fails, there is again no need to extend the cluster, since doing so will increase $gd(c', QID)$, but may not necessarily reduce $gd(c', SA)$. After finishing with creating a cluster, we check its size in step 11. If it is greater than k, tuples in the cluster are recoded, otherwise this cluster is marked (steps 15-16). The process is repeated until all tuples of T are processed.

Marked clusters either contain less than k tuples and lower utility than the required constraint, or at least k tuples and lower protection (higher WGP score) than the protection constraint. Thus, marked clusters can be regarded as bad local optima, as they resulted by extending them in the hope of getting sufficient protection. Therefore, in the final stage, tuples belonging to such clusters are un-clustered and re-grouped (steps 17-18). Each of these tuples is added into a non-marked cluster when its insertion will not violate the quality constraints for the cluster. If there is more than one cluster that a marked tuple can be added into, the cluster that will result in a minimal increment in Group Diversity when the tuple is inserted into it is selected and the marked tuple is added into it. If no such clusters can be found, we suppress the tuple.

Having presented our method, we discuss how it differs from the TOC-based method presented in Chapter 3. First, the search of the method presented in this chapter is guided by the Group Diversity heuristic, which is different from that used in the TOC-based method. This allows better protection for data, as it handles range disclosure too. Second, we use an alternative linkage strategy in clustering. More specifically, we extend a cluster with the best tuple according to Group Diversity, and not with the one that is closest to the seed, as in the TOC-based method. The former strategy is known to be more effective when seeds are randomly selected [127, 84]. Finally, a new stopping criterion is used to guarantee that all clusters will have an acceptable amount of information loss and protection. In particular, in an attempt to avoid creating over-generalised clusters, we stop extending a cluster when its size is less than k and utility gets worse than its associated constraint U. This is because extending the cluster cannot reduce the amount of information loss that will be incurred when data in it is k-anonymised. In addition, we also stop extending the cluster when its size is at least k and any of the constraints U or P is violated. This is because, a cluster needs to have at least k tuples and to satisfy both of these constraints to be released.

On performance, Algorithm 6 has the same time complexity as the TOC-based method, which is $O(n \times s)$, where n is the cardinality of the dataset and s the minimum subspace size created by MBP (a small constant in practice). This is because clustering is performed in each subspace separately, so a significantly smaller number of tuples are checked for insertion into a cluster, compared to clustering-based methods [127, 35, 96] whose complexity is quadratic.

5.3 Summary

In this chapter, we addressed the limitations of the Trade-Off Constrained (TOC) approach by proposing Range Diversity, a measure that can be used to guard against range disclosure, and the UPC approach that allows k-anonymised data with a required level of data utility and privacy protection to be generated. Range Diversity quantifies the amount of protection by taking into account both positive and negative range disclosure, based on frequency and semantic closeness of values in ranges. Furthermore, it can ensure that SA values in both numerical and categorical attributes are protected from range disclosure. Based on the UPC approach, we also developed an effective k-anonymisation method. Our method employs Range Diversity to produce anonymisations that prevent range disclosure and allows original SA values to be released. This allows much more information contained in the original data to be retained. In addition, it can generate anonymisations efficiently as a result of combining partitioning and clustering heuristics.

Chapter 6

Evaluation of Performance Enhancement

In this chapter, we experimentally evaluate the performance enhancement we proposed in Chapter 5. We first examine how our UPC method compares against the Trade-Off Constrained (TOC) approach described in Chapter 3. We then analyse the quality of anonymisations produced by our Utility and Protection Constrained (UPC) method in terms of utility and protection achieved by comparing them to those derived by some protection constrained and benchmark methods that specifically optimise data utility or privacy protection. We also study the effect of policies for specifying protection for range disclosure and constraints used in our algorithm. Finally, we examine the efficiency of our method.

6.1 Experimental Setup and Datasets

To compare the UPC method with existing protection constrained methods, we chose three popular methods: the Incognito algorithm with (c,l)-diversity [85] and t-closeness [80] protection criteria, and the Personalised anonymity method [124] which all attempt to guarantee one specific form of disclosure. Incognito is a global recoding k-anonymisation algorithm that follows a principle similar to the a-priori search heuristic used in association rule mining [21]. When combined with privacy principles such as (c, l)-diversity or t-closeness, it can generate anonymisations with an "optimal" level of utility and bounded level of protection. In contrast, the Personalised method does not employ k-anonymity (i.e. groups can have less than k tuples), but resorts to generalising SA values for protection. This algorithm performs a single-dimensional global recoding search in the space of QIDs first, to generate groups with optimal utility, and then performs generalisation of SA values in each group for protection.

To examine the quality of anonymisations produced by our method in terms of utility and protection achieved, we compared them to those generated by K-Members [35], K-Members-p and Incognito with t-closeness. These algorithms were used as benchmarks, due to their ability to achieve good quality in terms of utility (K-Members) or protection (K-Members-p and Incognito with t-closeness).

We used publicly available implementations of the Incognito algorithm and the Personalised method, and our own implementations of the algorithm proposed in Chapter 5, K-Members, K-Members-p and Mondrian. The environment for implementing the algorithms and performing these experiments is the same as the one used in Chapter 4.

We used both real and synthetic datasets in our experiments. First, we used the CENSUS dataset [1], which contains demographical information of 100000 US citisens. This dataset is comprised of 5 QIDs, namely *Age, Gender, Education, Marital status, Occupation*, and an SA *Income*, and was configured as shown in Table 6.1. The CENSUS dataset is similar to the Adults dataset used in Chapter 4, and has been used in [124, 85] to evaluate the Personalised method and Incognito with l-diversity.

[1] http://www.ipums.org

Attribute	Domain size	Type
Age	79	Numerical QID
Gender	2	Categorical QID
Education	17	Categorical QID
Marital status	6	Categorical QID
Occupation	26	Categorical QID
Income	50	Categorical SA

Table 6.1: Summary of attributes for CENSUS dataset

The main reason for us to use this dataset instead of Adults is that there are publicly available domain hierarchies associated with all its attributes. Thus, we could use CENSUS to compare our method with Incognito and the Personalised method which require hierarchies to work, as well as with benchmark methods. The hierarchy for *Income* is illustrated in Figure 6.1 [2], while the hierarchies for all QID attributes are given in Appendix A. We also note that *Age* is handled as a numerical QID by the Personalised method and our UPC method. However, as the Incognito algorithm cannot handle numerical QIDs, *Age* is handled using a hierarchy shown in Appendix A.

We also used a synthetic dataset containing 50000 tuples, which was generated from a standard normal distribution using the Apache Commons-Math library [3]. This dataset is comprised of 5 QIDs and 2 SAs, all having integer values in [0,19]. All of the attributes are associated with the hierarchy shown in Figure 6.2.

6.2 Trade-Off vs. Guarantees

To demonstrate the effectiveness of our UPC approach, we show how it outperformed the Trade-Off Constrained (TOC) method in cases where generating data with a

[2] The codes for the hierarchy in *Income* (see Figure 6.1) follow the classification used in [124].

[3] http://commons.apache.org/math/

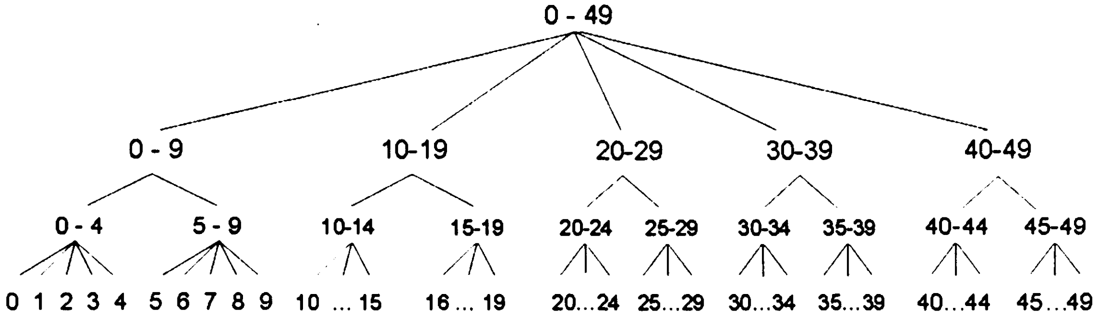


Figure 6.1: Hierarchy for *Income*

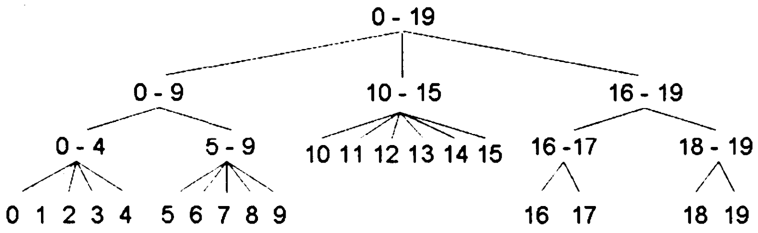


Figure 6.2: Hierarchy for attributes in the synthetic dataset

bounded level of utility and protection for range disclosure is required. We used a data publishing scenario illustrated in Figure 6.3 in experiments.

dataset: CENSUS
data utility requirement (max. acceptable WGU level): 0.7
privacy protection requirement (max. acceptable WGP level): 0.5
Sensitive Range (SR): level-2 [4] ascendant for a value in the hierarchy of Figure 6.1

Figure 6.3: Data publishing scenario

[4] A level-2 ascendant for a value is the ascendant that lies in the path between a leaf value and the root at a height of 2, where height 0 corresponds to a leaf value. For example, the level-2 ascendant for the value 0 in the hierarchy shown in Figure 6.1 is the value $0 - 9$.

The selected data utility and protection requirements should allow the TOC approach trade-off some utility for protection. To meet these requirements, we configured the UPC method by setting $U = 0.7$ and $P = 0.5$. The sensitive ranges were set as in Figure 6.3, and the subspace size constraint s used in Algorithm 2 was set using a sample-based heuristic similar to the one presented in Section 3.3.1. Although we could have used our sample-based heuristic given in Section 3.3.1 to set the threshold δ automatically, we used various manually selected δ values in this experiments. This was done to investigate whether a "good" δ can help TOC satisfy utility and protection requirements, even though it does not help TOC achieve the best utility/protection trade-off. We also configured the TOC method by setting $w_u = 0.5$, $w_p = 0.5$, s using the heuristic presented in Section 3.3.1.

We first examined whether the two methods satisfy the minimum acceptable utility requirement. As can be seen in Figures 6.4 and 6.5, both methods achieved an acceptable utility when k is set to 2 for most cases. However, the TOC method incurred lower information loss according to the UM measure (see Definition 3.1.2) when the threshold δ is at least 0.3. In all other cases, the TOC method suppressed all tuples in the dataset, and thus the WGU and UM scores are undefined and not shown in Figures 6.4 and 6.5 respectively. Furthermore, we report the percentage of suppressed tuples in Figure 6.6. Note that the result for the UPC method is not shown in Figure 6.6, as our method did not suppress any tuples. The results for $k = 5$ are qualitatively similar and are shown in Figures 6.7, 6.8 and 6.9 respectively. Thus, although both methods satisfied the specific utility requirement, the UPC method did so without suppressing any tuples.

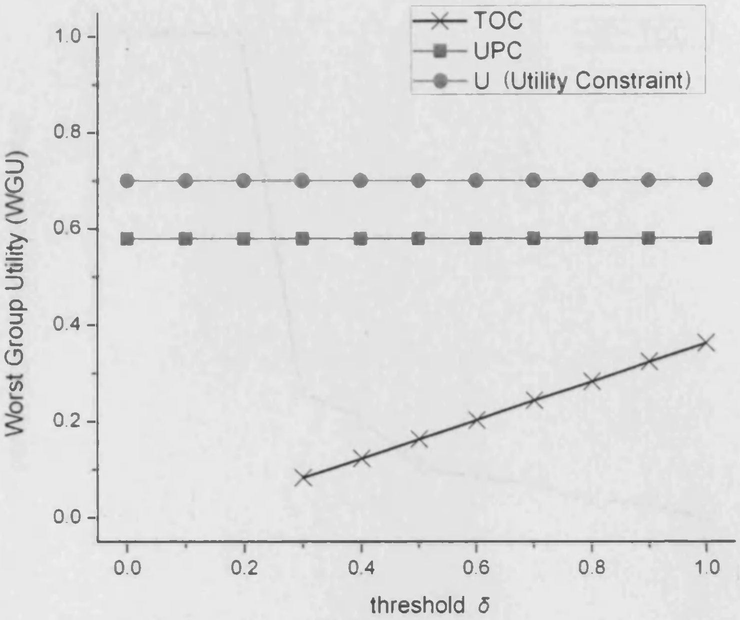


Figure 6.4: Worst Group Utility scores for $k = 2$

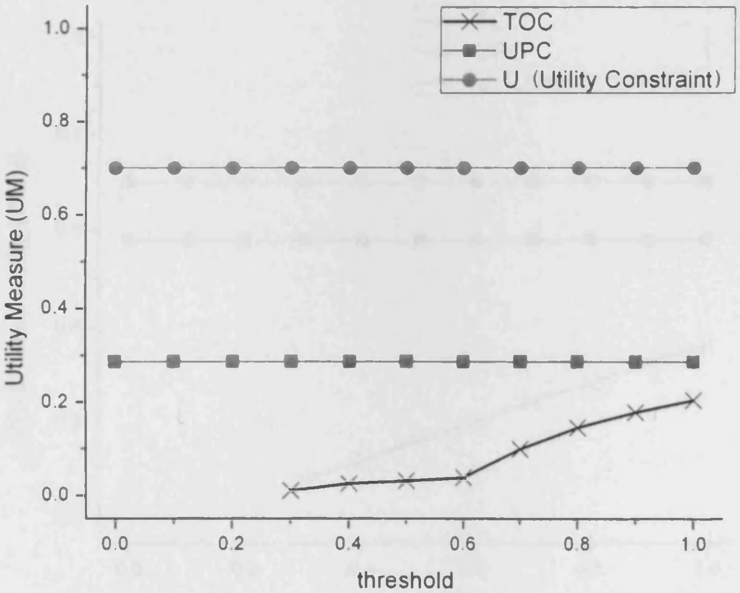


Figure 6.5: Utility Measure scores for $k = 2$

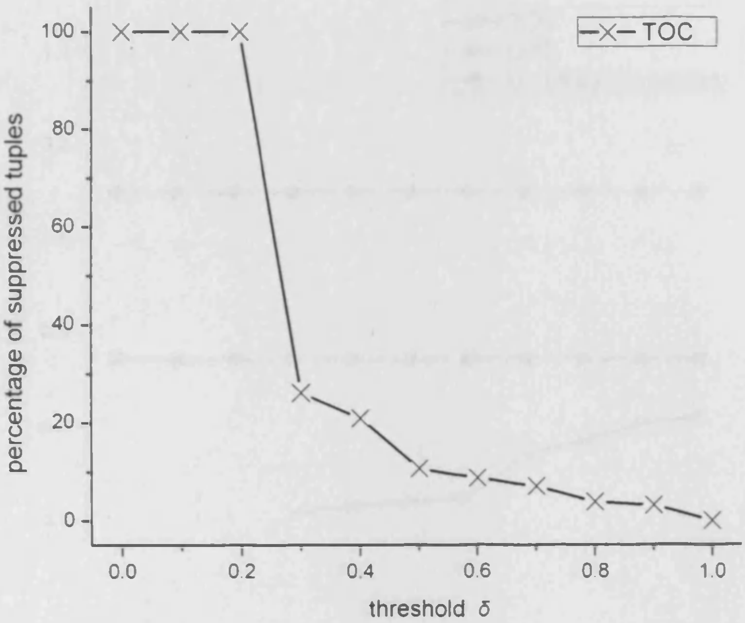


Figure 6.6: Percentage of suppressed tuples for $k = 2$

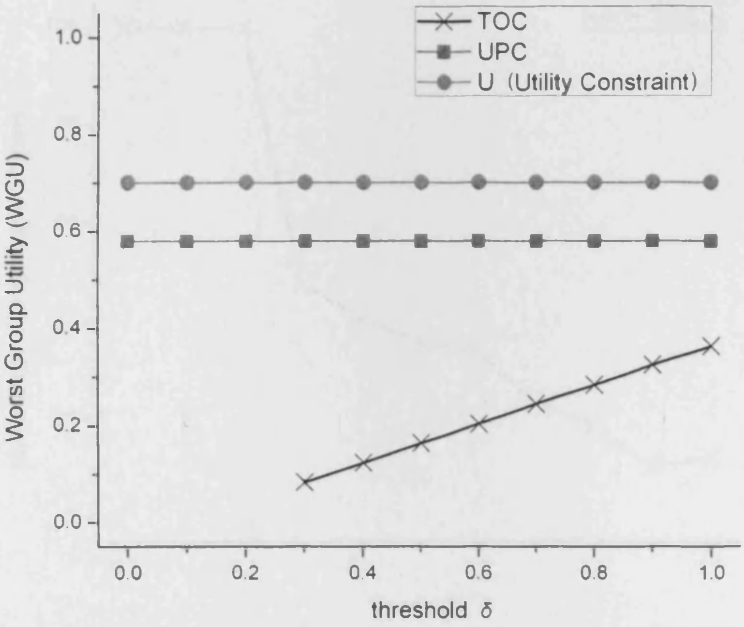


Figure 6.7: Worst Group Utility scores for $k = 5$

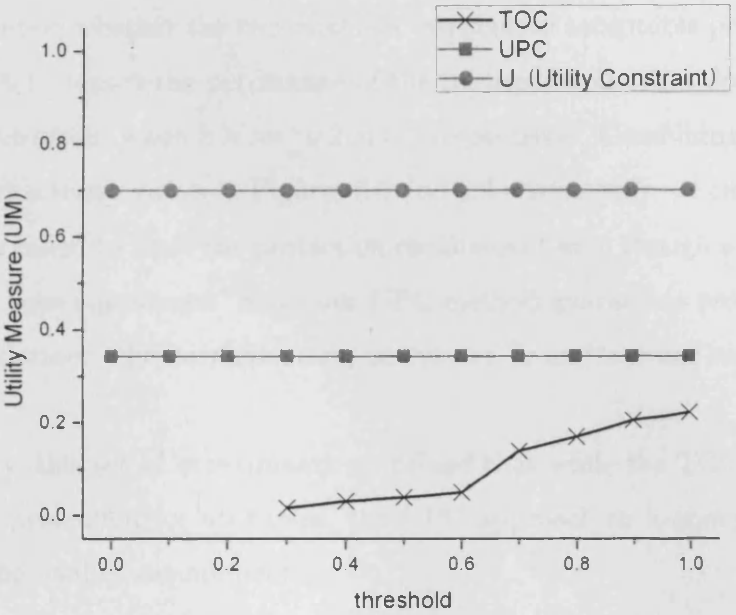


Figure 6.8: Utility Measure scores for $k = 5$

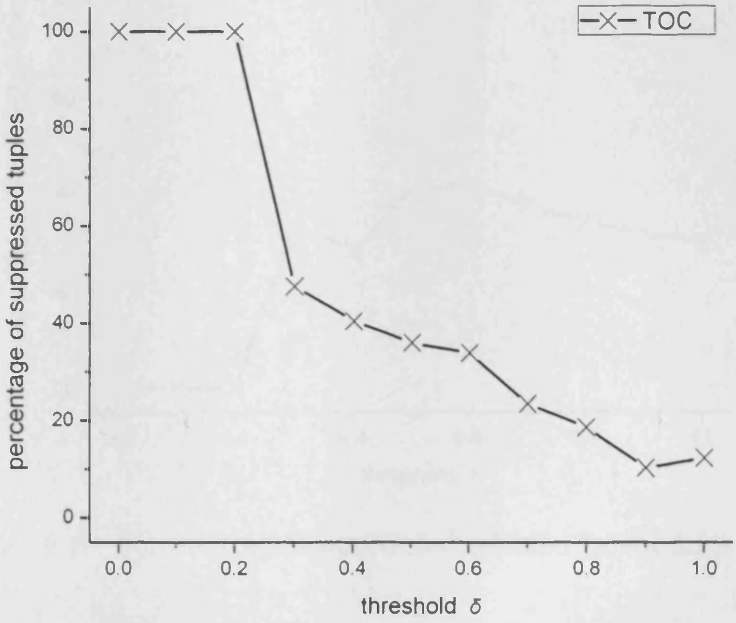


Figure 6.9: Percentage of suppressed tuples for $k = 5$

We also studied whether the two methods can achieve acceptable protection. Figures 6.10 and 6.11 report the percentage of the released tuples that do not meet the protection requirement when k is set to 2 and 5 respectively. Combining the results of these figures with those shown in Figures 6.6 and 6.9 respectively, we can see that the TOC approach failed to meet the protection requirement even though a large fraction of tuples have been suppressed. Since our UPC method guarantees protection for all tuples in the dataset, the corresponding scores are 0, and are not reported in the figures.

In summary, this set of experiments confirmed that while the TOC method may not guarantee protection for all tuples, the UPC approach to k-anonymisation can always meet the quality requirements.

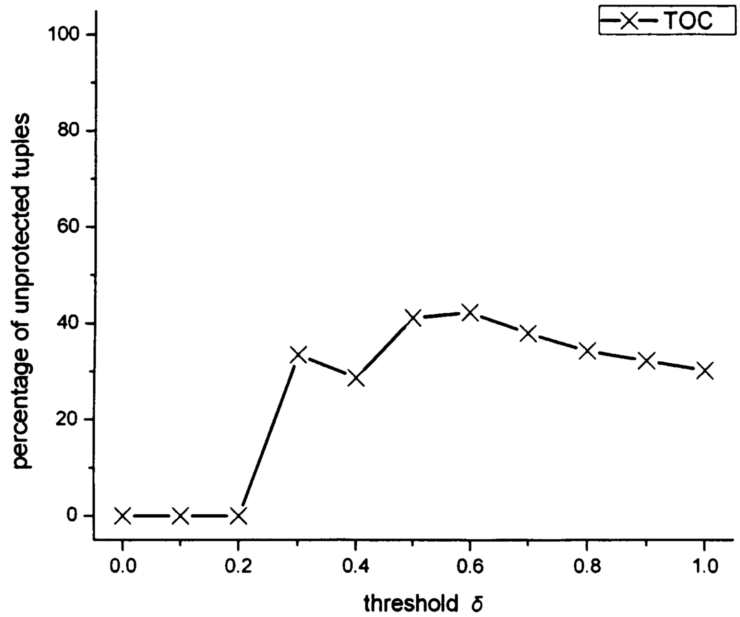


Figure 6.10: Percentage of unprotected released tuples for $k = 2$

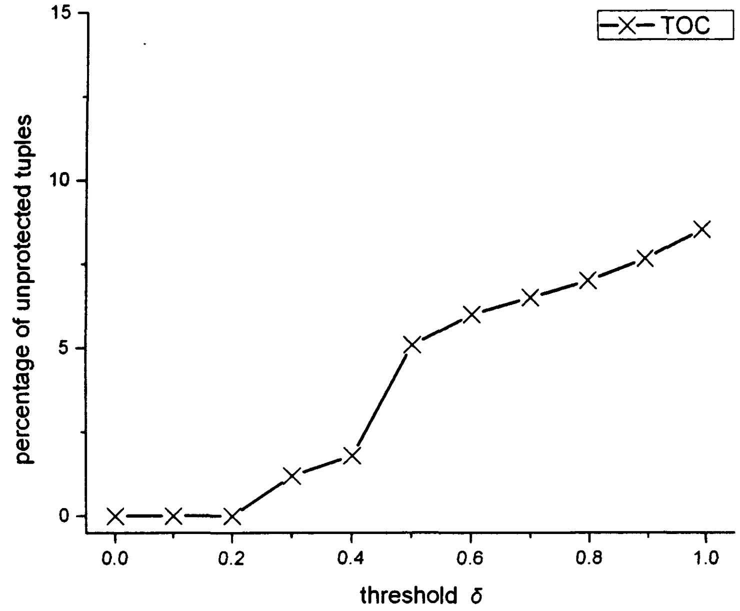


Figure 6.11: Percentage of unprotected released tuples for $k = 5$

6.3 Quality of Anonymisation Against Protection Constrained Methods

For this set of experiments, our UPC method was configured as in Section 6.2, while Incognito ran with $c = 2$ and $l = 2$ for (c, l)-diversity and $t = 0.9$ for t-closeness. The Personalised method was configured as in [124].

6.3.1 Utility of Anonymisations

We first tested the utility of anonymisations using aggregate query answering as an indicative application [77, 127, 124]. We considered two types of COUNT queries as illustrated below.

Attribute q is a QID, selected uniformly at random. Following [124], we chose r

```
select COUNT(*)
from anonymised table
where q = r and s between u and
u + 10
```

Figure 6.12: A type-1 query

```
select COUNT(*)
from anonymised table
where q = r and s = u
```

Figure 6.13: A type-2 query

to be an interval of range 15 for *Age* (covering 19% of the domain for this attribute), a leaf value for *Gender*, or a level-1 value for any other QID. Attribute s denotes the SA *Income* and u is a randomly selected value in [0,40).

We considered a workload of 10000 queries of each type, which was applied on original and anonymised data, and measured the relative error (RE) [77, 127, 124] as in Section 4.5.2. For ease of reference, we briefly describe how to compute RE again. RE is defined as $RE = \frac{|act(q)-est(q)|}{act(q)}$, where $act(q)$ is the answer of a query q on original data and $est(q)$ the estimated answer of the same query on anonymised data. We derive $est(q)$ by counting the probability an anonymised tuple answers q and summing these probabilities for each one of these tuples [77, 124]. Assuming that data is uniformly distributed, the probability a tuple answers q is given by $p' = \frac{Rq \cap R}{R}$, where R_q and R denote the areas covered by q and the qualifying tuples in the generalised table, respectively, and $Rq \cap R$ denotes the overlap between R_q and R.

Figures 6.14 and 6.15 report the RE values for type-1 queries in descending order, when k is set to 2 and 10 respectively. As can be seen, the RE values for our UPC method were an order of magnitude lower than that achieved by the Incognito algorithm for both (c, l)-diversity and t-closeness measures for most queries, and comparable to that of the Personalised method.

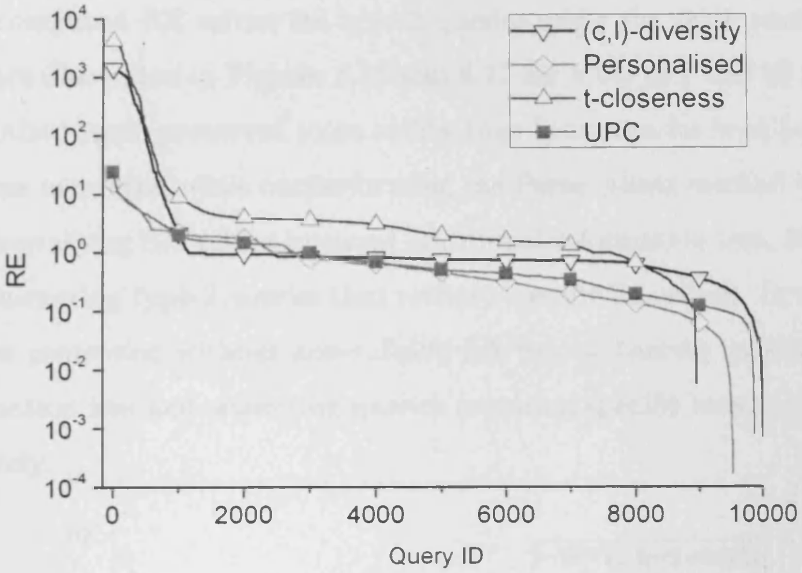


Figure 6.14: R.E. for type-1 queries and $k = 2$

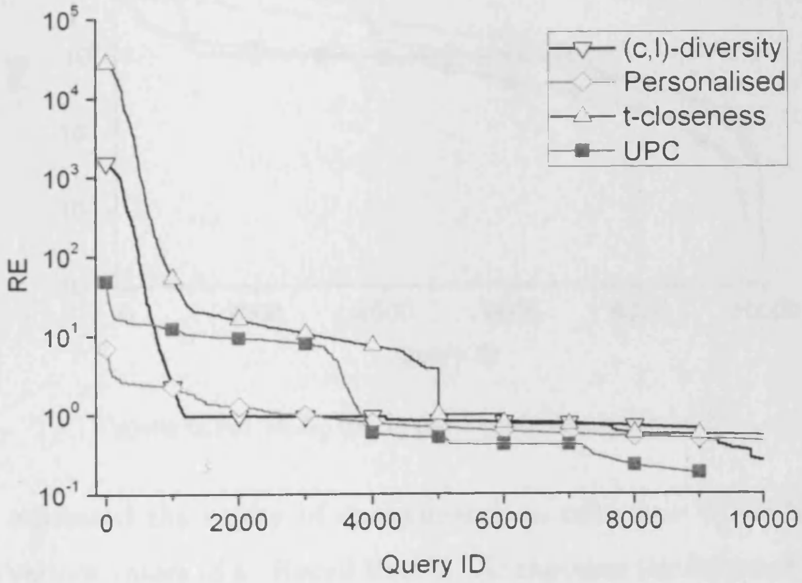


Figure 6.15: R.E. for type-1 queries and $k = 10$

We also computed RE values for type-2 queries using the same anonymisations. The results are illustrated in Figures 6.16 and 6.17 for k set to 2 and 10 respectively. The UPC method again preserved more utility than Incognito for both (c, l)-diversity and t-closeness measures, while outperforming the Personalised method as well. This is because generalising SA values incurred additional information loss, degrading the accuracy of answering type-2 queries that retrieve specific SA values. In contrast, our method offers protection without generalising SA values, thereby avoiding the associated information loss and answering queries requiring specific sensitive information more accurately.

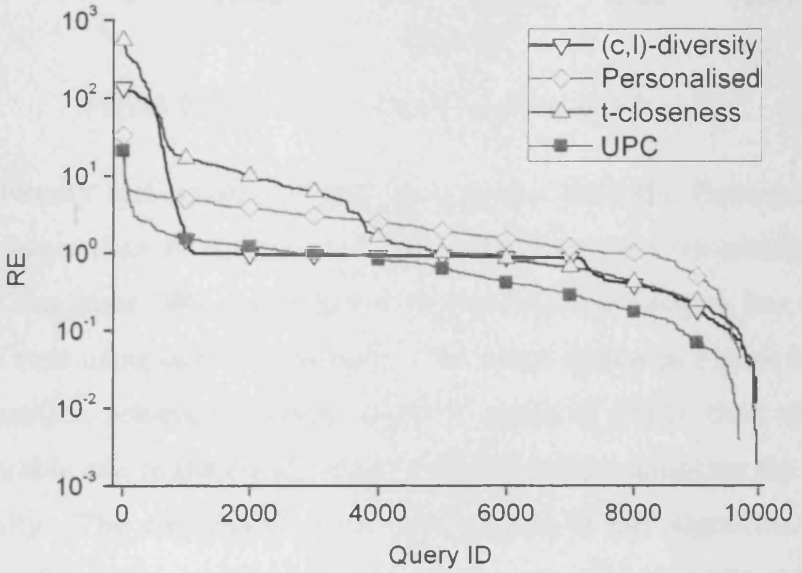


Figure 6.16: R.E. for type-2 queries and $k = 2$

We then evaluated the utility of anonymisations using our Worst-Group Utility measure, for various values of k. Recall that WGU captures the information loss of all QIDs of the most heavily generalised group, as explained in Definition 5.2.2. As can be seen in Figure 6.18, the UPC method achieved a WGU score of less than 0.7 for all values as the threshold U was set to 0.7. This result was similar to the Incognito with

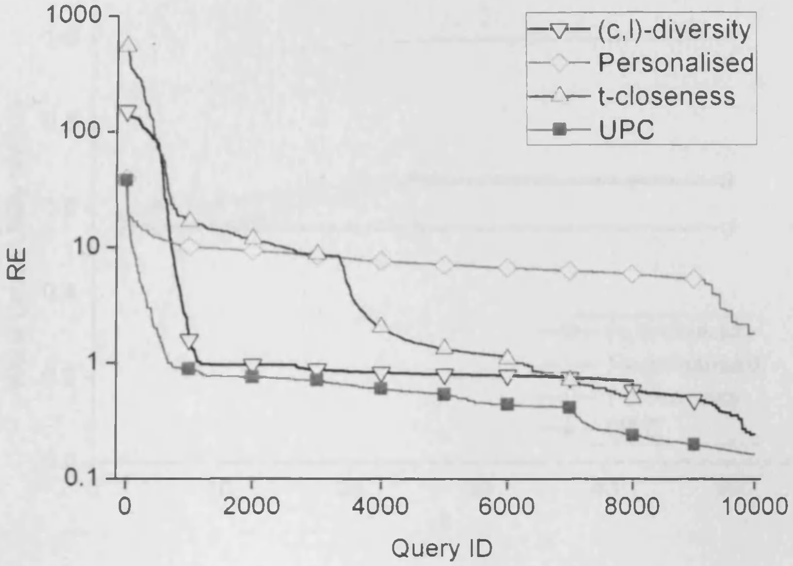


Figure 6.17: R.E. for type-2 queries and $k = 10$

both (c, l)-diversity and t-closeness and much better than the Personalised method when k was larger than 10. In fact, the Personalised method completely generalised all QIDs in these cases. We also measured the average information loss of all groups for the same runs using our UM measure. The result shown in Figure 6.19 confirms that our algorithm achieved a better result in terms of utility than the Incognito, and a comparable one to the Personalised method, despite adopting the requirement of k-anonymity. The remarkably good performance of our algorithm in terms of the incurred information loss is attributed to the use of combined partitioning and clustering heuristics and the powerful local recoding generalisation model [127].

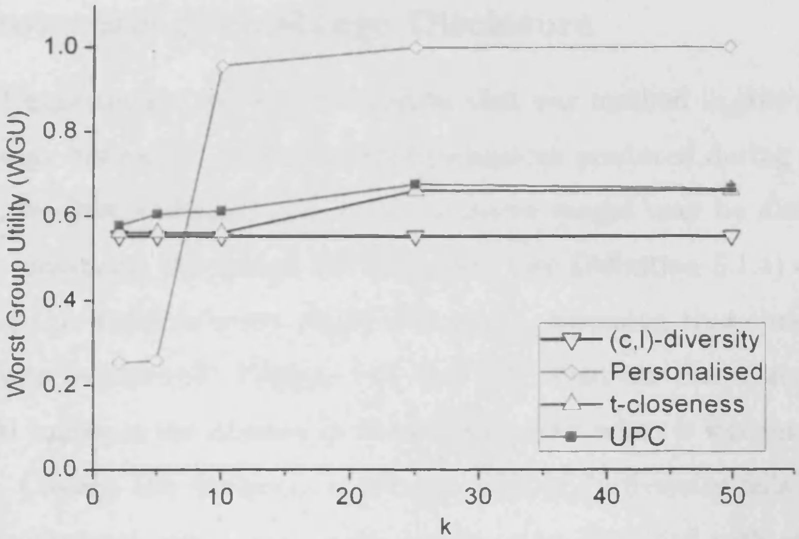


Figure 6.18: Worst Group Utility scores

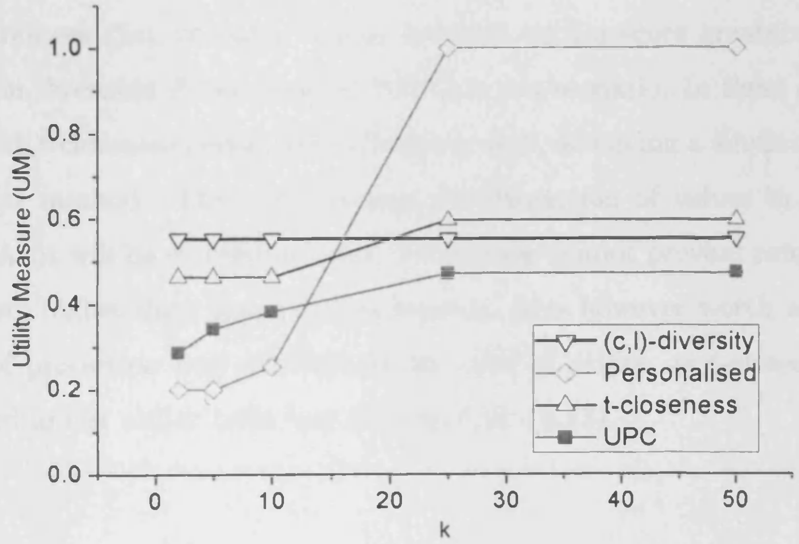


Figure 6.19: Utility Measure scores

6.3.2 Protection from Range Disclosure

In this set of experiments, we aim to confirm that our method is able to offer protection for range disclosure. Given the anonymisations produced during the previous experiments, we first evaluated how easily sensitive ranges may be disclosed. This was done by measuring the Range Diversity (rd) (see Definition 5.1.4) of all groups and assigning this score to every tuple of a group, assuming that these tuples are equally likely to be inferred. Figures 6.20 and 6.21 illustrate the rd scores for each of the 100000 tuples in the dataset in descending order when k was set to 2 and 10 respectively. Clearly, the Incognito algorithm with (c, l)-diversity fails to meet the protection requirement, since every tuple is able to be identified with an rd score of more than 0.85. Similarly, the Personalised method was unable to protect all tuples from range disclosure, since SA values in some groups can be identified with an rd score of 0.9 as well. This is because the level of protection required to prevent range disclosure is higher than that specified by personal preferences. In contrast, our UPC method guarantees that no tuple can be inferred with a score greater than 0.5, as the protection threshold P has been set to 0.5 in our scenario. In these experiments, Incognito with t-closeness performed sufficiently well, achieving a similar level of protection to our method. This was because the dispersion of values in *Income* was rather low. As it will be elaborated later, t-closeness cannot prevent range disclosure in cases where higher data dispersion is present. It is however worth noting that a good level of protection was achieved at the cost of utility, as t-closeness was the worst method in our utility tests (see Figures 6.14 - 6.17).

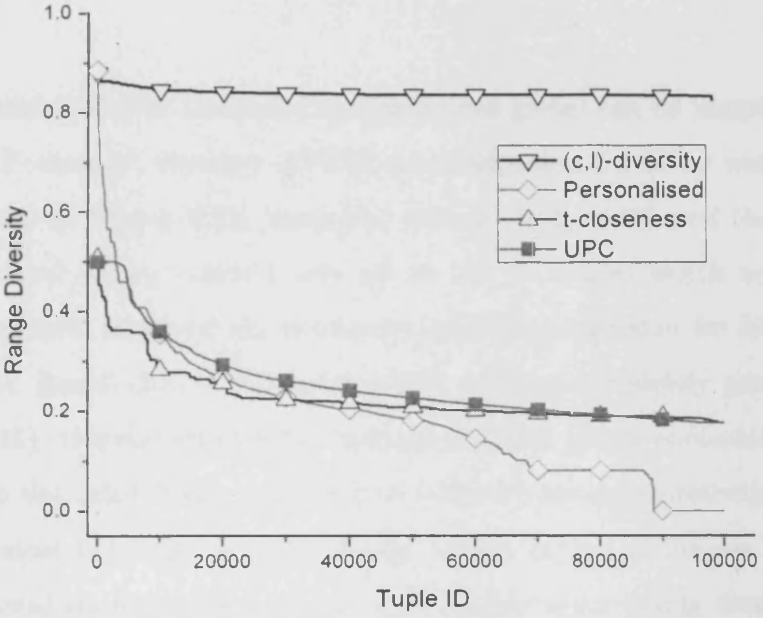


Figure 6.20: Range Diversity scores for $k = 2$

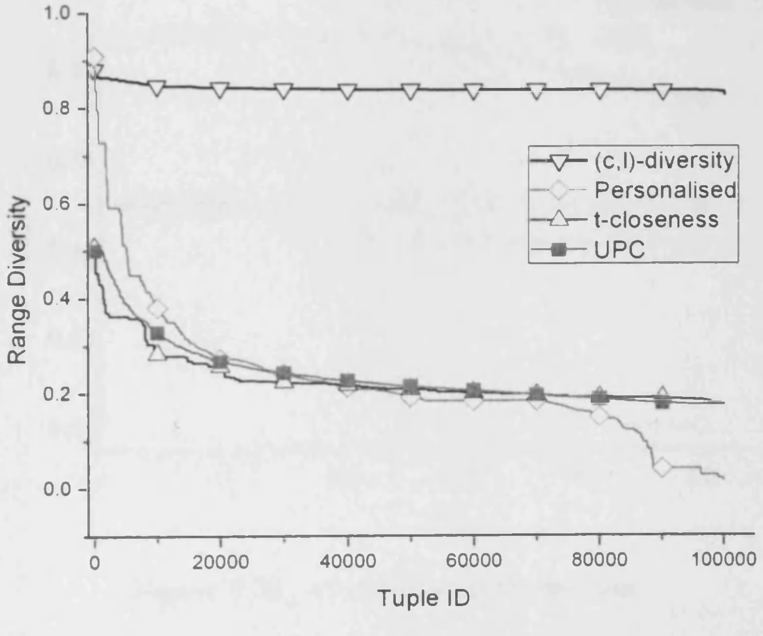


Figure 6.21: Range Diversity scores for $k = 10$

We then measured how likely the less protected group can be identified using our Worst-Group Protection measure (WGP) (see Definition 5.1.5) by varying k from 2 to 50. As shown in Figure 6.22, Incognito with (c, l)-diversity and the Personalised method performed poorly when k was up to 10. It is also worth noting that the Personalised method achieved the maximum possible protection for SA values when k was 25 or 50. Recall that in these cases this method completely generalised QIDs (see Figure 6.19), thereby effectively creating only one group containing all possible SA values. On the other hand, our method achieved stronger protection (the WGP score was at most 0.5) with a much better utility across all values of k, while t-closeness achieved strong protection but considerably lower utility (see Figure 6.19).

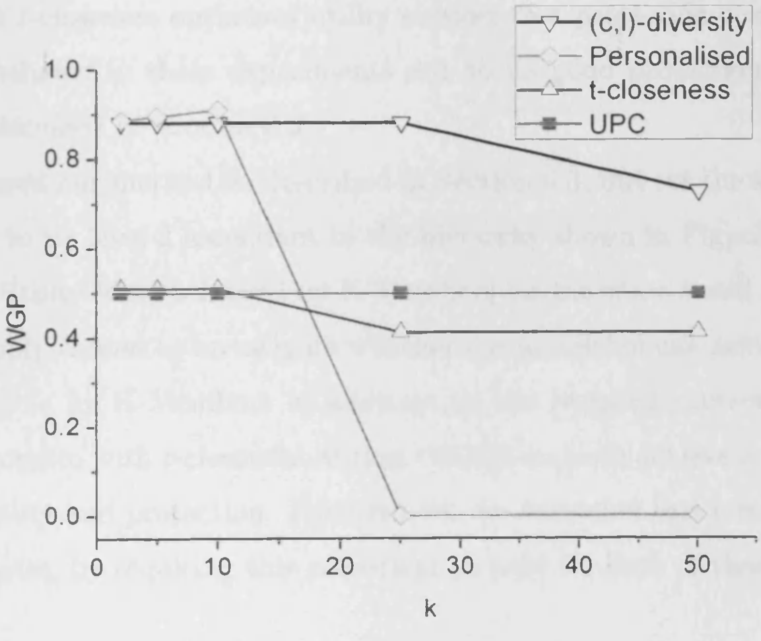


Figure 6.22: Worst-Group Protection

6.4 Quality of Anonymisations Against Benchmark Methods

Having shown that our method is able to guarantee utility and protection better than protection constrained methods, we now proceed to compare it to some other benchmark algorithms in terms of how they fare in achieving better utility and protection. For this set of experiments we used the synthetic dataset as described in Section 6.1, and used K-Members [35], K-Members-p and Incognito with t-closeness. K-Members is a powerful local recoding algorithm based on clustering, which has been shown to outperform both partition-based [77] and other clustering-based [83, 84] algorithms in terms of data utility. K-Members-p is a modified version of K-members, which uses Range Diversity as its objective function. Different from these two algorithms, Incognito with t-closeness optimises utility subject to a protection constraint. However, it was included in these experiments due to its good protection scores in our experiments discussed in Section 6.3.

We configured our method as described in Section 6.3, but set the sensitive range for each value to its level-2 ascendant in the hierarchy shown in Figure 6.2, U to the Worst-Group Utility score achieved by K-Members for the same k and $P = 0.7$. This set-up was simply chosen to investigate whether our algorithm can achieve the utility that is achievable by K-Members in addition to the required protection. We also configured Incognito with t-closeness setting t to 0.8, so as to achieve a "good" trade-off between utility and protection. Furthermore, we extended this method to handle two SA attributes, by requiring this constraint to hold for both of these attributes.

6.4.1 Utility of Anonymisations

The utility of anonymisations was evaluated using a workload consisting of 10000 queries of the form illustrated in Figure 6.23.

select COUNT(*)
from anonymised table
where $q = r$ and $s_1 = u_1$ and $s_2 = u_2$

Figure 6.23: A query applied on synthetic data

These queries are similar to type-2 queries used in Section 6.3. Attribute q is a QID, selected uniformly at random and r is a randomly chosen interval which covers 20% of its domain. Attributes s_1, s_2 are SAs and u_1, u_2 are randomly selected values in [0,19].

We first produced anonymisations using K-Members, Incognito with t-closeness and our UPC method, and estimated the relative error (RE) in query answering, when k is set to 2 and 10. K-Members-p was excluded from this set of experiments as it does not optimise data utility. Figures 6.24 and 6.25 show the results.

It is apparent from these figures that our method was able to preserve more utility than the Incognito algorithm with t-closeness, producing a qualitatively similar result to that of K-Members in terms of utility.

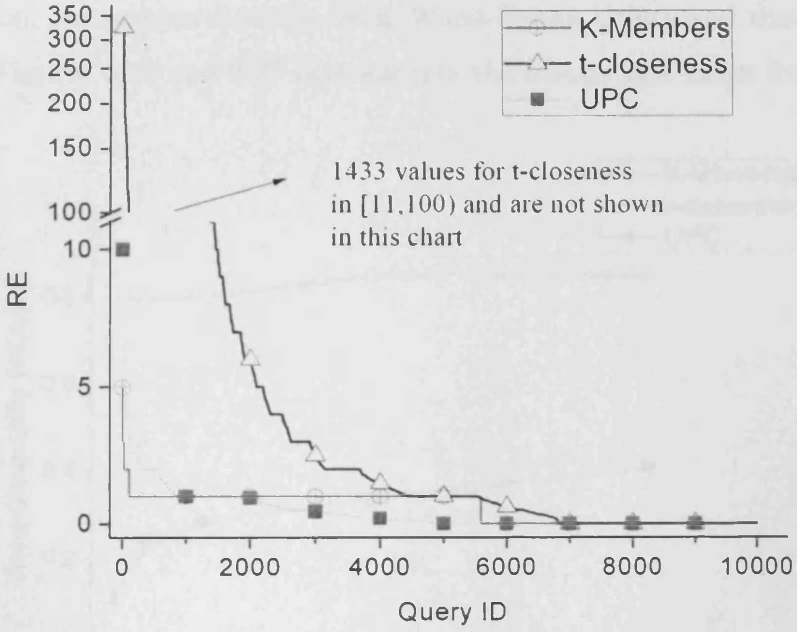


Figure 6.24: R.E. for $k = 2$

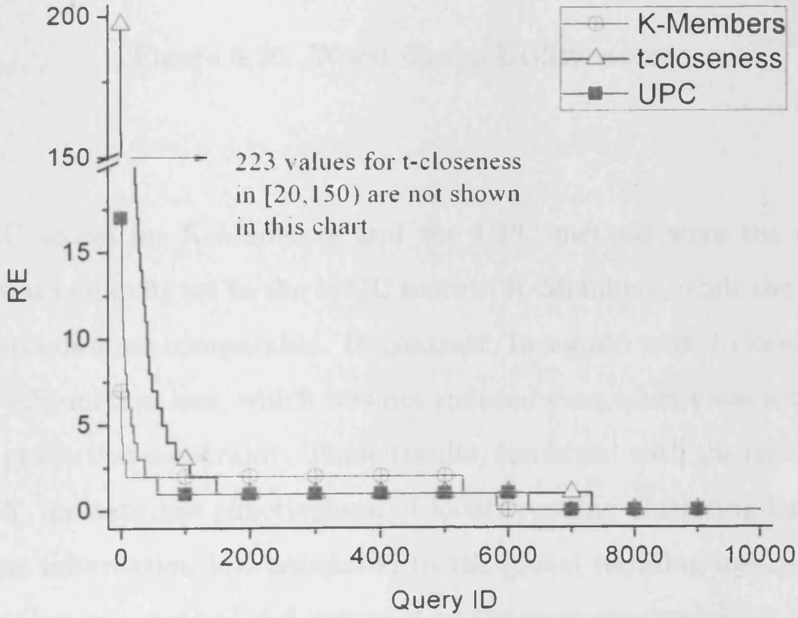


Figure 6.25: R.E. for $k = 10$

In addition, we measured utility using Worst-Group Utility and the Utility Measure (UM). Figures 6.26 and 6.27 demonstrate the results as k varies from 2 to 50.

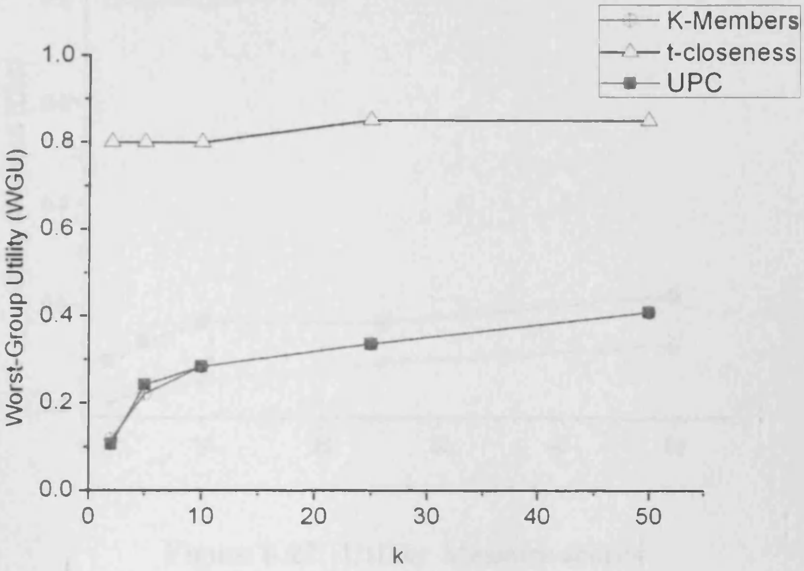


Figure 6.26: Worst Group Utility scores

The WGU scores for K-Members and the UPC method were the same, as the threshold U was manually set to the WGU score of K-Members, while the UM scores of these two methods were comparable. In contrast, Incognito with t-closeness incurred much higher information loss, which was not reduced even when t was set to 1 thereby ignoring the protection constraint. These results, combined with the results of Figures 6.24 and 6.25, indicate the effectiveness of local recoding clustering-based methods for minimising information loss compared to the global recoding methods. It is also worth noting that our method did not need to suppress any tuples.

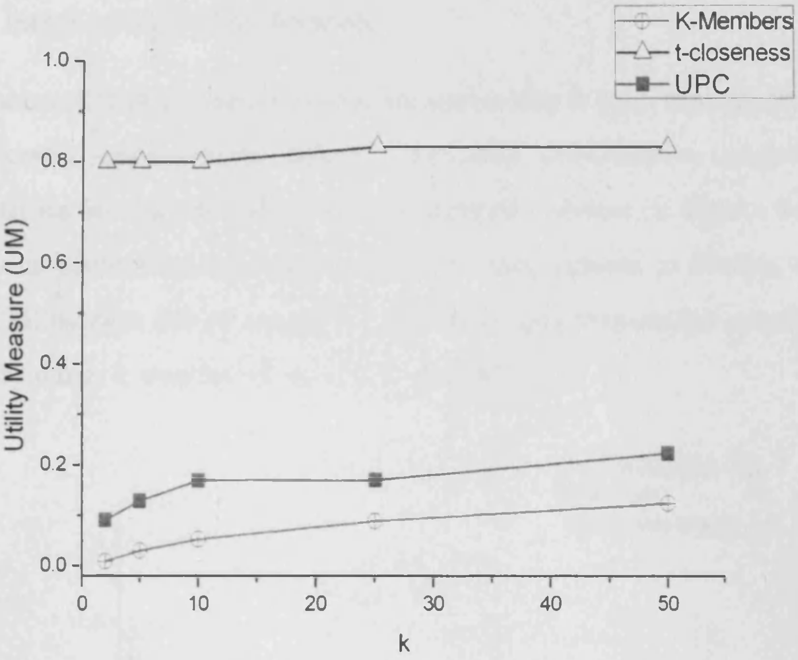


Figure 6.27: Utility Measure scores

6.4.2 Protection for Range Disclosure

To examine how our algorithm performs against benchmark methods in terms of protection for range disclosure, we conducted similar experiments to those in Section 6.3 on the synthetic dataset, using K-Members-p and Incognito with t-closeness as benchmark algorithms. These methods differ in terms of how protection is achieved:

1. K-Members-p optimises our Worst-Group Protection measure attempting to offer protection for range disclosure, while Incognito with t-closeness protects SAs by controlling the distribution in each group (see Section 2.1.2).
2. K-Members-p does not ensure a minimum protection level in all groups, unlike Incognito with t-closeness that requires each group to have an acceptable level of t-closeness.
3. K-Members-p restricts the group size to k, while Incognito with t-closeness

allows larger groups to be formed.

We also assume that the requirement for protection is such that no group can have a Range Diversity score of more than 0.7 and that the sensitive ranges for each SA value is set to its level-2 ascendant in the hierarchy shown in Figure 6.2 (the effect of using various protection requirements will be investigated in Section 6.5). Figures 6.28 and 6.29 illustrate the *rd* scores for the same anonymisations studied in Figures 6.24 and 6.25, when k was set to 2 and 10 respectively.

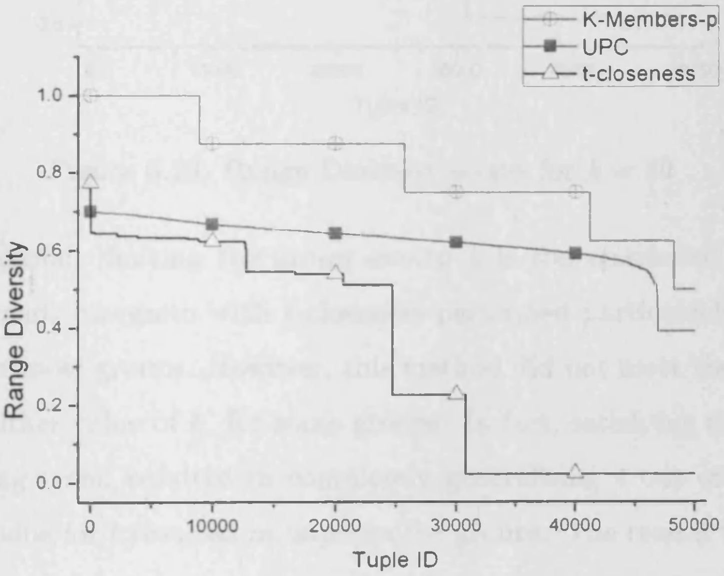


Figure 6.28: Range Diversity scores for $k = 2$

These results show that K-Members-p does not offer protection for range disclosure. More specifically, approximately 83% failed to satisfy the protection requirement when k was set to 2. The result is similar when k was set to 10; 65% of tuples did not meet the protection requirement. The reason for this is two-fold. First, K-Members does not constrain protection, thus some groups of unacceptable protection can be

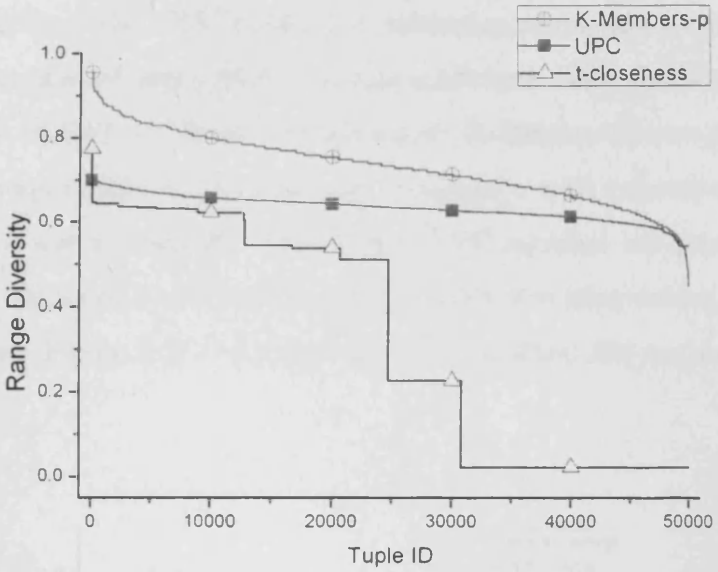


Figure 6.29: Range Diversity scores for $k = 10$

constructed. Second, limiting the group size to k is too restrictive for protection. On the other hand, Incognito with t-closeness performed particularly well in terms of protection for most groups. However, this method did not meet the protection requirement for either value of k, for some groups. In fact, satisfying this requirement required lowering t and resulted in completely generalising 4 out of 5 QIDs, while using a larger value for t resulted in unprotected groups. The reason that t-closeness did not meet the protection requirement in this experiment is related to the highly dispersed data distribution we used to generate our synthetic dataset. As discussed in Chapter 2, t-closeness cannot achieve protection for range disclosure, even when values in all groups follow the distribution of original data with respect to their SAs. It is evident therefore that solely constraining the distribution of SA values in each group is neither sufficient nor practical to achieve protection for range disclosure. In contrast, our algorithm bounds the protection of a group using P, which was set to 0.7, in order to achieve acceptable protection for all groups.

The superiority of the UPC method in achieving protection for range disclosure compared to benchmark algorithms was also confirmed using the Worst-Group Protection measure (WGP). As shown in Figure 6.30, K-Members-p was unable to satisfy the protection requirement in any case, while Incognito with t-closeness was only able to do so when k was at least 25. In contrast, UPC satisfied the protection requirement across all values of k and incurred significantly less information loss than other methods did (see Figure 6.27). Furthermore, our method did not need to suppress any tuples.

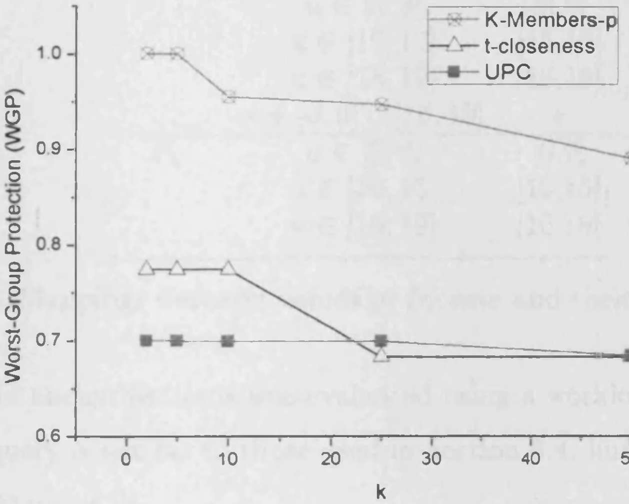


Figure 6.30: Worst-Group Protection

6.5 Effect of Parameters

We first considered the impact of specifying different policies for setting the sensitive ranges on utility. We simulated five different policies P_1 to P_5, which are numbered from 1 to 5 in ascending order of the specified level of protection required. The mappings between an SA value u and its sensitive range for P_1 to P_5 are illustrated in Table 6.2. For this experiment we used a synthetic dataset obtained by discarding one SA of the dataset used in Section 6.4. Our method ran with $U = 0.5$ and $P = 0.5$.

Policy	*SA value u*	*SR(u)*
P_1	$u \in [0, 19]$	u
P_2	$u \in [10, 15]$	[10,15]
	$u \notin [10, 15]$	u
P_3	$u \in [10, 15]$	[10,15]
	$u \in [16, 17]$	[16,17]
	$u \notin [10, 17]$	u
P_4	$u \in [0, 4]$	[0,9]
	$u \in [5, 9]$	[0,9]
	$u \in [16, 17]$	[16,19]
	$u \in [18, 19]$	[16,19]
	$u \notin [0, 9] \cup [16, 19]$	u
P_5	$u \in [0, 9]$	[0,9]
	$u \in [10, 15]$	[10,15]
	$u \in [16, 19]$	[16,19]

Table 6.2: Mappings between values in *Income* and their sensitive ranges

The utility of anonymisations was evaluated using a workload consisting of 10000 queries. Each query is similar to those used in Section 6.4, but retrieves only one SA as depicted in Figure 6.31.

```
select COUNT(*)
from anonymised table
where q = r and s = u
```

Figure 6.31: A query applied on synthetic data

Tables 6.3 and 6.4 show the average and maximum RE values respectively, for various values in k from 2 and 50. Clearly, there is a trade-off between utility and the strictness of policy, suggesting that some utility needs to be given away when increased protection is required. This confirms that not specifying how well certain ranges need to be protected, as practised by the t-closeness may result in unnecessary degradation of the utility. For instance, in the case of P_5 the average and maximum RE illustrated in Tables 6.3 and 6.4 is 10 and 85 times larger respectively, compared to those when P_4 was used instead.

k	P_1	P_2	P_3	P_4	P_5
2	3.728	3.755	3.763	4.164	42.661
5	3.805	3.806	3.906	4.220	42.661
10	3.839	3.861	3.902	4.220	42.646
25	3.855	3.879	3.934	4.247	42.667
50	5.686	5.594	5.621	5.718	43.227

Table 6.3: Average RE values

k	P_1	P_2	P_3	P_4	P_5
2	65.5	66	69.5	148	12693
5	68	64	64	162	12692
10	67.5	67	66	166	12685
25	64	65.5	72	169	12694
50	143	92	97	136	12363

Table 6.4: Maximum RE values

We additionally studied how different policies affect information loss as captured by the utility measure (UM). As can be seen in Table 6.5, most groups continue to retain a similar amount of information across all policies, as the UPC method builds groups with bounded information loss and protection, thereby avoiding overgeneralising data. Furthermore, we note that in this experiment UPC did not need to suppress any tuples.

k	P_1	P_2	P_3	P_4	P_5
2	0.180	0.180	0.180	0.180	0.182
5	0.180	0.182	0.181	0.180	0.181
10	0.180	0.180	0.179	0.179	0.181
25	0.179	0.179	0.179	0.179	0.181
50	0.232	0.233	0.233	0.233	0.234

Table 6.5: Utility Measure scores

Next, we examined the effect of the thresholds U and P on utility. For this experiment k was set to 25, U to 0.3, and P varied from 0 (no range disclosure is allowed) to 1 (no protection against range disclosure is offered). UM scores and the percentage of suppressed tuples are reported in Figures 6.32 and 6.33 respectively. Observe that the information loss incurred to provide protection for most values of P is relatively low, compared to when no protection is required. This implies that protection for a wide range of attackers with sufficient data utility retained is feasible. Furthermore, the UM scores are bounded by U, as long as such anonymisation can be derived from the given dataset. In all other cases the UM scores are undefined and not shown in Figure 6.32.

For the same k, protection was constrained setting P to 0.5 and U varied from 0 (no information loss) to 1 (maximum information loss). Figure 6.34 shows the scores in UM measure in relation to U. Since the protection constraint is particularly tight (compared to the'one used in Section 6.4), low values of U may result in tuple suppression. We report the percentage of suppressed tuples in Figure 6.35. As can be seen, average utility scores are always bounded by U, implying that no more information loss than acceptable is incurred by generalisation to meet the protection constraint P.

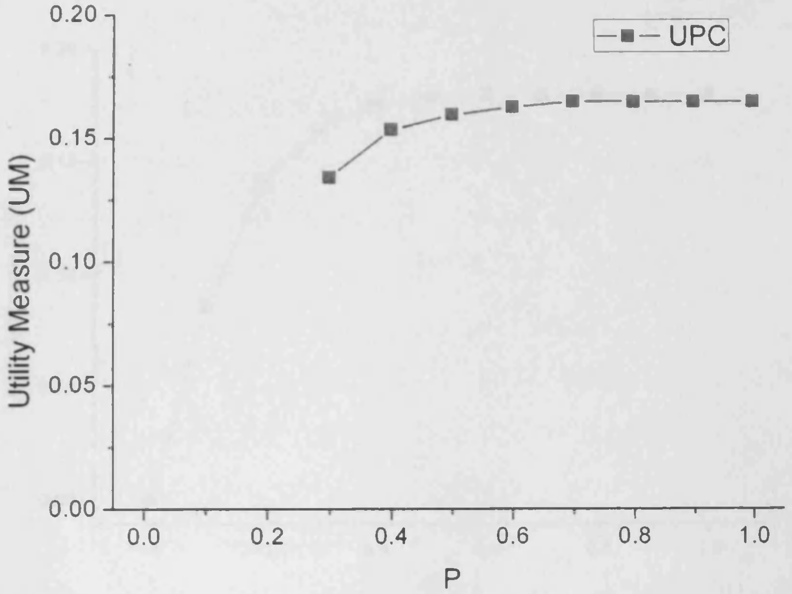


Figure 6.32: The impact of P threshold on information loss

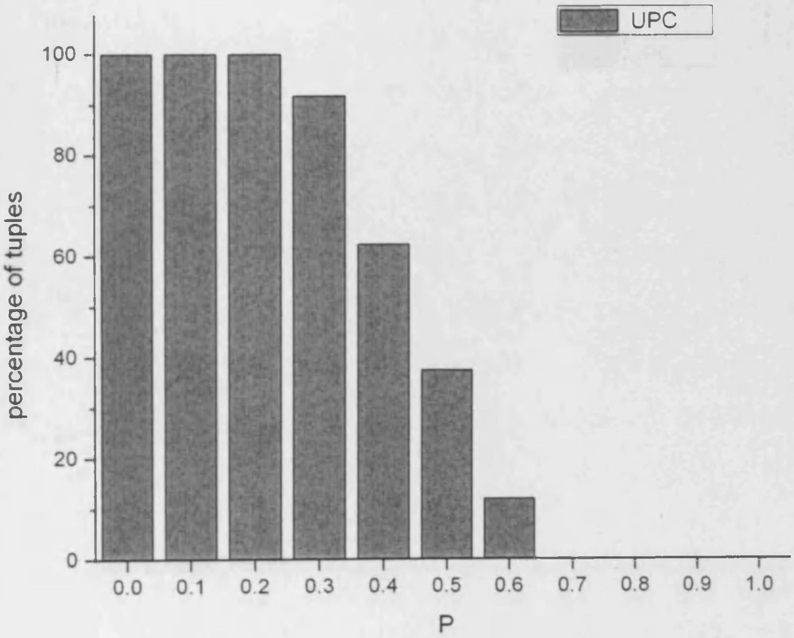


Figure 6.33: The impact of P on tuple suppression

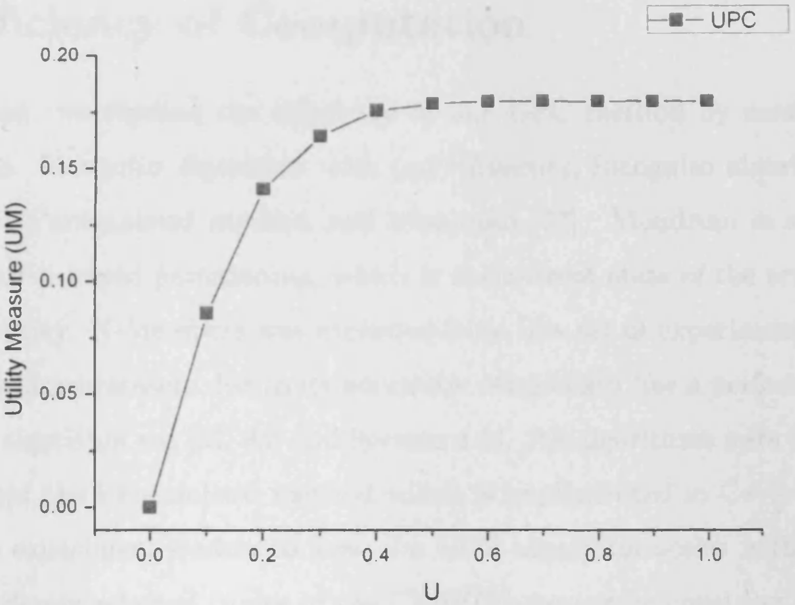


Figure 6.34: The impact of U threshold on information loss

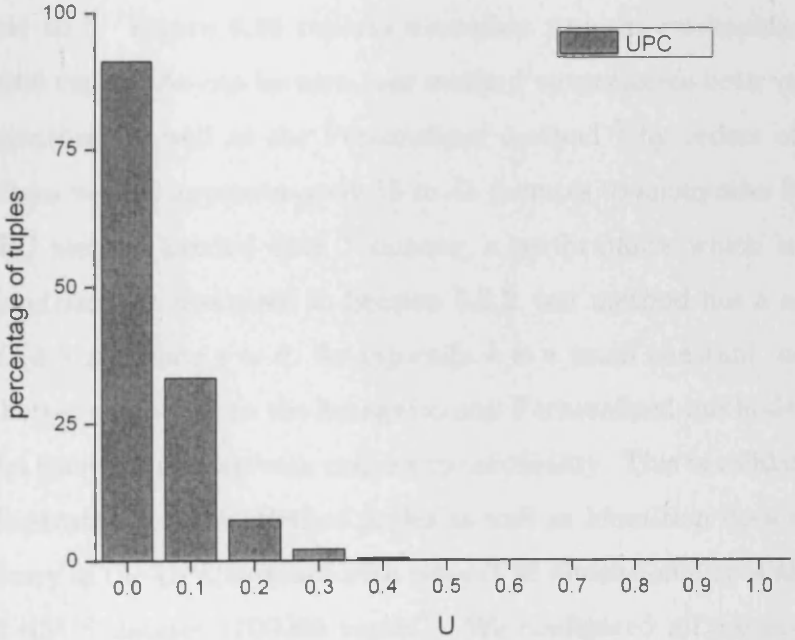


Figure 6.35: The impact of U on tuple suppression

6.6 Efficiency of Computation

In this section, we studied the efficiency of our UPC method by comparing it to four methods: Incognito algorithm with (c, l)-diversity, Incognito algorithm with t-closeness, the Personalised method and Mondrian [77]. Mondrian is an algorithm based on median-based partitioning, which is the current state of the art in terms of runtime efficiency. K-Members was excluded from this set of experiments, since it is inefficient for large datasets due to its quadratic complexity (for a performance evaluation of this algorithm see [35, 84] and Section 4.3). All algorithms were implemented in Java, except the Personalised method which is implemented in C++.

Our first experiment evaluated how the UPC algorithm scales with cardinality. We used randomly selected tuples of the CENSUS dataset to construct the datasets used in these experiments. All the methods were configured as in Section 6.3 and ran with k set to 5. Figure 6.36 reports execution time as cardinality varies from 10000 to 500000 tuples. As can be seen, our method outperforms both versions of the Incognito algorithm, as well as the Personalised method [5] by orders of magnitude. These algorithms needed approximately 15 to 45 minutes to anonymise 500000 tuples while the UPC method needed only 1 minute, a performance which is comparable to that of Mondrian. As discussed in Section 5.2.2, our method has a complexity of $O(s \times n) \approx O(k \times n)$, when $s \approx k$. As typically k is a small constant, our algorithm scales much better compared to the Incognito and Personalised methods, which have an exponential time complexity with respect to cardinality. This is validated in Figure 6.36 which illustrates that our method scales as well as Mondrian does in practice.

The efficiency of the UPC method with respect to dimensionality is also examined using the CENSUS dataset (100000 tuples). We configured all methods as in the previous experiment and varied the number of QIDs from 2 to 7. The result is shown

[5] The Personalised method is favoured against other methods here as it is implemented in C++ which is faster than Java. However, the UPC method is still significantly faster.

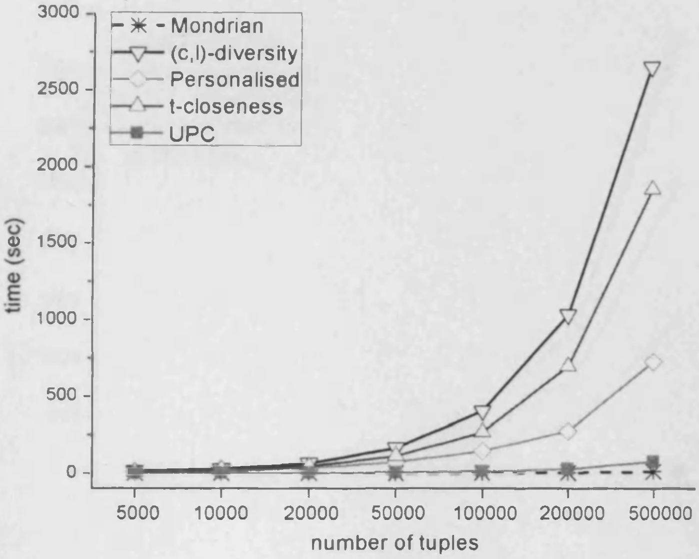


Figure 6.36: Efficiency comparison w.r.t. cardinality

in Figure 6.37. Again, our algorithm significantly outperformed both versions of Incognito and the Personalised method. More specifically, these algorithms required from 22 minutes to 1 hour to anonymise the dataset and their efficiency increased by a factor from 3 to 6 for an increase of dimensionality of 1. In contrast, our algorithm needed only 20 seconds to anonymise the dataset, while it scales linearly to the dimensionality, achieving similar performance to that of the Mondrian algorithm.

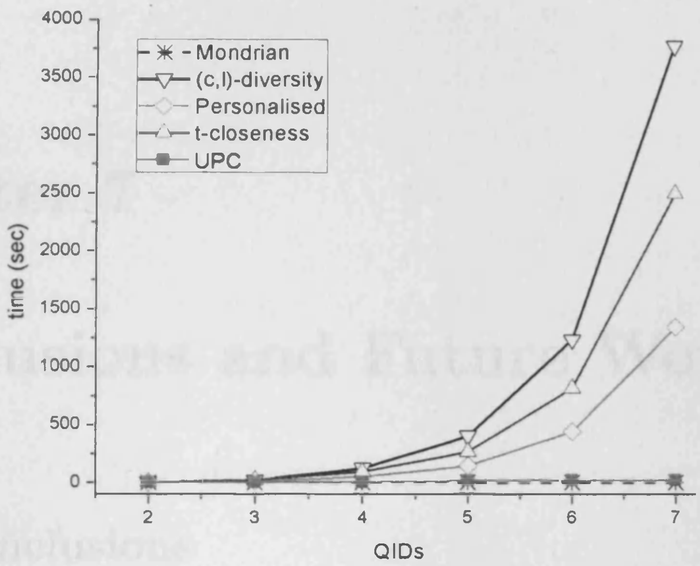


Figure 6.37: Efficiency comparison w.r.t. dimensionality

6.7 Summary

Our extensive experiments using both real and synthetic datasets confirmed that our method is the only one that guarantees protection for range disclosure, while preserving sufficient utility in anonymised data. More specifically, it outperformed existing approaches by orders of magnitude in many cases in terms of utility, and was comparable to K-Members, which is capable of achieving very good utility in k-anonymisation. Furthermore, we showed that our method scales very well to large datasets with respect to both cardinality and dimensionality.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The wide collection and use of microdata in a variety of applications enunciates the need for protecting private information about the individuals represented in the data. In the thesis, we studied this problem under a setting where a data publisher uses k-anonymisation to mask microdata before its release. The contributions can be summarised as follows:

- **k-Anonymisation approaches:** This thesis proposed two novel approaches for k-anonymisation: the Trade-Off Constrained (TOC) approach that allows released data to have a good balance between data utility and privacy protection, and the Utility and Protection Constrained (UPC) approach that guarantees that anonymised data will have a minimum level of utility and protection.

  These two approaches address a significant limitation of existing approaches outlined in Section 2.1.3, as they attempt to produce solutions that do not incur an excessive amount of information loss and protect sensitive information about individuals sufficiently. The experimental results presented in the thesis suggest that anonymisations with a very good level in both data utility and privacy protection can be generated when the TOC or the UPC approach is used.

This result is important for many real-world applications, such as medical data analysis and location-based systems. This is because these applications require anonymised data to be useful for analysis (e.g. to allow users to study the demographic profile of patients suffering a certain disease or to determine a location point of interest accurately), while being protected against sensitive information disclosure (e.g. to prevent a pharmaceutical company from identifying an individual's disease or a location-service provider from learning an individual's exact location). Thus, generating anonymisations that satisfy both of these requirements using our proposed approaches, may alleviate the privacy concerns that individuals may have and help the development of these applications in practice.

- **Utility criteria:** Measuring the level of data utility is essential to ensure that anonymised data is useful for analysis. In our work, we introduced new criteria to capture data utility offered by anonymised data based on the amount of information-loss incurred by generalisation. Our criteria are informative, since they are based on the probability of reconstructing original values from generalised data, and practical, as they can be applied to both numerical and categorical attributes, and are independent of the dimensionality and cardinality of the considered dataset.

  The importance of our criteria is two-fold. First, they can be used by anonymisers or data users to assess the level of data utility provided by anonymised data, regardless of the anonymisation method. For instance, they may suggest that anonymisers should change the way data is anonymised in cases where the utility level is not appropriate for the application, e.g. lowering the level of k. Second, our utility metrics can be used as optimality criteria by existing anonymisation algorithms that employ partitioning [77], clustering [127, 96] or genetic search strategies [64], guiding them towards producing more useful anonymisations.

- **Protection criteria:** While k-anonymisation can make data linkage difficult, it may not prevent sensitive information disclosure completely [85]. Motivated by the need to guard against attacks on anonymised data, we proposed two protection criteria. S-diversity, which attempts to quantify protection by measuring the maximum range between SA values in anonymised groups, and Range Diversity, which allows detailed protection specification and quantifies protection against disclosing sensitive information of individuals through ranges.

  Our criteria make an important step towards accurately measuring the level of privacy protection provided by anonymised data. This is because, as explained in the thesis, they can capture attacks that are not prevented by existing privacy principles. Thus, they can be a valuable tool for anonymisers who want to make sure that a data release is well protected according to their policy. Furthermore, our measures are based on the same notion of distance as our utility criteria, and thus share much of their flexibility, i.e. they can handle attributes of both types, and datasets of different cardinality and dimensionality. Again, this is a feature that anonymisers should find helpful. Finally, we have shown that it is possible to incorporate our protection criteria into a clustering-based algorithm, so that anonymised data with a controlled level of privacy protection can be produced.

- **k-Anonymisation algorithms:** While many algorithms for k-anonymising data have been proposed, they either construct low quality anonymisations fast or focus purely on effectiveness and require a large amount of time to produce a result. Motivated by this, we designed an algorithmic framework that allows the generation of high quality anonymisations efficiently, by using both partitioning and clustering search strategies.

The main technical challenge was to combine these strategies in a way that the speed offered by partitioning in exploring the search space does not harm the quality of data grouping derived by clustering. To address this issue, we made a number of technical contributions. We proposed to use a partitioning strategy that helps reducing the time complexity of clustering and is efficient to perform, we developed a clustering algorithm that specifically reduces the effect of overlapping subspaces created by the partitioning step, and devised a number of heuristics based on sampling that allow our method to be configured automatically.

This algorithmic framework was used to produce anonymisations following both the TOC and the UPC approaches in an effective and efficient way. Extensive experiments confirmed that it outperformed in majority of cases the state of the art method in terms of data utility, and was substantially better in terms of achieving privacy protection. In addition, we have shown that it is practical to use, since it can be easily configured using our sample-based heuristics and scales very well to large datasets.

7.2 Future Work

In this section we outline a number of issues in the area of data publishing that worth further investigation.

Similar to most existing works [106, 25, 76, 124, 80, 71, 127, 35, 96], we have in fact made a number of simplifying assumptions in our study. Removing these assumptions may help the effectiveness and scalability of our methods, allowing them to be used in a wider class of today's applications. However, it involves tackling a number of challenges which we outline below:

- **Static vs. dynamic datasets:** Our methods assume that a static view of a dataset needs to be published. However, releasing a new anonymisation of the same dataset when data is updated (i.e. tuple insertions or deletions occur) may be required. For instance, a hospital may need to release a dataset containing patients' demographic and medical information to a pharmaceutical company periodically. In this case, anonymised data may include tuples that did not appear in previous releases (i.e. when patients that were not hospitalised before are included in the dataset) or it may not include some of the tuples that were released before (i.e. when patients have recovered and left the hospital).

  Unfortunately, dealing with data insertion and deletion turns out to be challenging, since directly anonymising the updated dataset may lead to individual identification when different anonymous releases are stitched together [36, 125]. For example, assume that k has been set to 2 and that three patients are aged $10, 15$ and 16 respectively. Assume also that the tuples representing these patients form a group and that their values have been generalised using the range $[10 - 16]$ before release. Then, consider that the patient aged 16 leaves the hospital, and that data needs to be republished. Obviously, replacing the remaining values (i.e. 10 and 15) with the range $[10 - 15]$ before data is released again is possible, as 2-anonymity still holds. However, the tuple representing the patient who left the hospital may be revealed when an attacker intersects the two data releases. Currently, the only method that can handle dynamic datasets works by inserting fake data values to make sure that combining tables will not reveal any sensitive information [125]. However, inserting fake values may not be acceptable by applications such as medical research, where data needs to be truthful to be useful for analysis [110].

Since anonymising dynamic datasets is important for applications, we believe that studying this problem is worthwhile. Our utility and protection criteria can be used to measure and compare the quality of the generated solutions, but a new model to track possible ways of breaching privacy by combining anonymised dataset needs to be developed. This can possibly be done using some ideas from the work of Byun et al. [36], which studies how dynamic datasets with insertion updates can be anonymised, and the work of Yao et al. [130] which examines whether a set of views derived from a static dataset maintain k-anonymity.

- **Using utility measures in applications:**

  Our work developed utility criteria to capture information loss, but did not focus on how anonymisers can use these criteria in an optimal way. In particular, our Utility Measure (UM) (see Definition 3.1.2) computes the amount of information loss across all QID attributes and values of the dataset, and thus effectively assumes that the entire dataset will be used in applications. However, this assumption does not hold when users are only interested in a part of the dataset (e.g. use queries that need certain attributes or values to be answered). Consequently, our UM measure may not capture the data utility accurately in this case.

  To alleviate this problem, we observe that collaboration between users and anonymiser can be beneficial. More specifically, a user can provide anonymisers with a sample of the queries he/she intends to use. This enables anonymisers to know which QID attributes and values are of more importance to this user and tune anonymisation methods to minimally generalise them, thereby helping this user's analysis. Incorporating a query workload into anonymisation process requires modifying the way both data partitioning and clustering steps in our method work. LeFevre et al. [78] studied how anonymised groups that help answer a query workload comprised of selection and projection queries can be derived using a data partitioning strategy. Extending their methods to work for different types of query (e.g. join queries) and developing similar heuristics for clustering algorithms may be helpful towards achieving this goal.

However, it may not be feasible to expect users to disclose their queries, due to privacy concerns. For instance, an advertising company using a location-based service may not be willing to reveal its queries to a service owned by a rival company. Alternatively, users may be willing to disclose which QID attributes they want to be minimally generalised. Based on this knowledge, anonymisers can perform feature selection using split attribute selection (i.e. specifying the QID attribute along which data is partitioned) or attribute weighting (i.e. specifying which attribute values are more important in measuring similarity between tuples during clustering). Although methods that use feature selection [63] and attribute weighting [127] have been proposed, they require anonymisers to specify a number of parameters which are dependent on data distribution (e.g. the split attribute selection order or attribute weights). Thus, they are generally difficult to use in practice. Furthermore, the method proposed in [63] depends on a certain k-anonymisation algorithm, and cannot be applied when different algorithms are used. Therefore we believe that developing generic and practical feature selection methods that allow utility requirements to be incorporated in anonymisation is worthwhile.

- **Main vs. secondary memory-resident data:**

  Although we currently assume that the table to be anonymised fits in main memory, real-life database applications often deal with large datasets that need to be stored in secondary memory. Reading data directly from the secondary memory in our method is possible, but is very inefficient, since communication between the main and secondary memory is slow due to the large difference in their access times. Thus, our method needs to deal with memory and I/O management effectively to avoid thrashing and performance degradation.

We note that the work of LeFevre et al. [75] proposes a way to modify the partitioning step of our method so that large datasets can be handled efficiently. The main idea is to perform partitioning by computing the split points based on aggregate statistics that are computed efficiently and are small enough to fit in main memory. For instance, computing the median value in an QID a to perform a split in a subspace S can be performed by loading the frequency sets $(u_1, f_1), ..., (u_l, f_l)$ into main memory, where $u_i, i = 1, ..., l$ are distinct values in the domain of a QID a and f_i are their associated frequencies in S [75]. While this allows us creating subspaces with little I/O overhead, our method requires further modifications to be able to handle secondary memory-resident data efficiently. This is because the heuristics proposed in Section 3.3 work by considering all subspaces to find out where each left-over tuple "fits" best (see Section 3.3.2), which essentially means that the entire dataset needs to be transferred into the main memory if our method is directly applied. Finding a compact representation of partitioned dataset that fits in main memory, would be helpful to minimise the overhead of this transfer. However, this is not straightforward, and is considered as a subject for future research.

In this thesis, we have focused on relational data. However, in many cases individuals' private information is contained in unstructured data, i.e. data that lacks a well-defined schema. Common examples of unstructured data include documents, web content and log files. Preventing individual identification in unstructured data is particularly challenging. First, it is difficult to analyse or add a schema to such data in a completely automated way. For example, logs are often stored as text files and are analysed using information retrieval techniques. Second, unstructured data is often distributed across a number of sites which may not be willing to share unperturbed data due to privacy concerns [88]. For instance, assume that the registry and the finance departments of a university want to identify whether a certain employee is entitled to a salary increase. To perform this check, the documents containing this person's employment details which are held by the registry need to be examined together with his/her financial information contained in documents held by the finance department. Preserving privacy requires examining both the employment and financial details of this person without revealing information about other employees. Third, unstructured data is often continuously updated, and sensitive information may be leaked as a result of combining separate releases, as explained above. Thus, the privacy preserving techniques surveyed in Chapter 2, which assume that data is structured, centrally stored and static, are not applicable under this setting. To address this issue, we believe that developing algorithms that are based on the secure function evaluation paradigm [129] that can work on unstructured and continuous data is required.

Appendix A

Hierarchies and Generalisation Codes for QID Attributes In the CENSUS Dataset

This appendix includes the hierarchies and generalisation codes for the QID attributes in the CENSUS dataset, which was used in the experiments presented in Chapter 6.

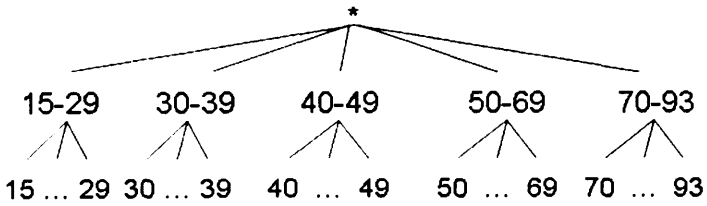


Figure A.1: Hierarchy for *Age*

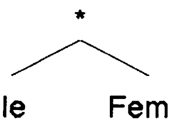


Figure A.2: Hierarchy for *Gender*

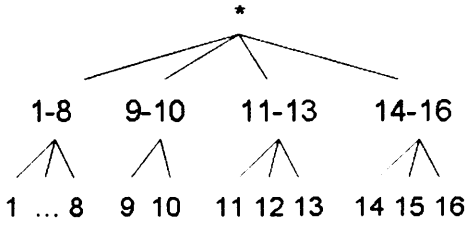


Figure A.3: Hierarchy for *Education*

Symbol	Value
1	Pre-school
2	1st-4th
3	5th-6th
4	7th-8th
5	9th
6	10th
7	11th
8	12th
9	High school grad
10	Some college
11	Bachelors
12	Prof. school
13	Academic assoc. degree
14	Occup. assoc. degree
15	Masters
16	Doctorate

Table A.1: Values for *Education* attribute

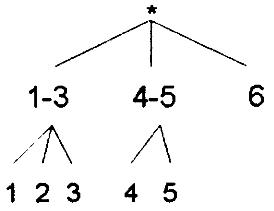


Figure A.4: Hierarchy for *Marital status*

Symbol	Value
1	Spouse present
2	Spouse absent
3	Separated
4	Divorced
5	Widowed
6	Single

Table A.2: Values for *Marital status* attribute

The codes for the hierarchy in *Occupation* shown in Figure A.5 refer to aggregate occupation values [6]. For example, 0 refers to individuals under the age of 16 or

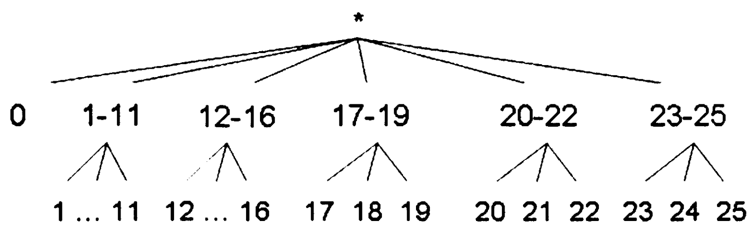


Figure A.5: Hierarchy for *Occupation*

those not in the labor force, who last worked more than 5 years ago.

Appendix B

Related Publications

1. G. Loukides and J. Shao. Preventing range disclosure in k-anonymised data. Under preparation for a journal submission.

2. G. Loukides and J. Shao. An empirical study of utility measures for k-anonymisation. *In BNCOD '08*, pages 15-27, 2008.

3. G. Loukides and J. Shao. Data utility and privacy protection trade-off in k-anonymisation. *In EDBT Workshop on Privacy and Anonymity in the Information Society (PAIS '08)*, 2008.

4. G. Loukides and J. Shao. An efficient clustering algorithm for k-anonymisation. *Journal of Computer Science and Technology*, Vol. 23(2):188-2002, 2008.

5. G. Loukides and J. Shao. Greedy clustering with sample-based heuristics for k-anonymisation. *In the First IEEE International Symposium on Data, Privacy, and E-Commerce (ISDPE '07)*, pages 191-196, 2007. *(Selected as one of the best four papers)*

6. G. Loukides and J. Shao. Clustering-based k-anonymisation algorithms. *In DEXA '07*, pages 761-771, 2007.

7. G. Loukides and J. Shao. Speeding up clustering-based k-anonymisation algorithms with pre-partitioning. *In BNCOD '07*, pages 203-214, 2007.
8. G. Loukides and J. Shao. Capturing data usefulness and privacy protection in k-anonymisation. *In SAC '07*, pages 370-374, 2007.
9. G. Loukides and J. Shao. Towards balancing data usefulness and privacy protection in k-anonymisation. *In IEEE International Conference on Computer and Information Technology (CIT)*, page 15, 2006. *(Excellent paper award)*
10. G. Loukides and J. Shao. Query-driven data anonymisation. *In BNCOD '06 PhD Forum*, 2006.

References

[1] Anonymizer web-site. http://www.anonymizer.com.

[2] DoubleClick web-site. http://www.doubleclick.com.

[3] Electronic Privacy Information Center DoubleClick complain. http://epic.org/privacy/internet/ftc/DCLK_complaint.pdf, 2000.

[4] Health Insurance Portability and Accountability Act of 1996 United States Public Law.

[5] HMRC leak. http://www.itweek.co.uk/itweek/news/2204126/hmrc-leak-raises-prospect-rules.

[6] Integrated public use microdata series, occupation codes. http://usa.ipums.org/usa/volii/00occup.shtml.

[7] Report of The Independent Investigation: breach of data security in the VFS online UK visa application facility, operated through VFS websites in India, Nigeria and Russia. http://www.fco.gov.uk/Files/kfile/IndependentInvestigation26July2007.pdf, 2007.

[8] Stanford encyclopaedia of philosophy. http://plato.stanford.edu/entries/privacy/.

[9] The National DNA Database Annual Report 2005-2006. http://www.homeoffice.gov.uk/documents/DNA-report2005-06.pdf?view=Binary.

[10] U.S. department of health and human services privacy act regulations. http://www.access.gpo.gov/nara/cfr/waisidx_04/45cfr5b/_04.html.

[11] European Union Directive 95/46/EC. *Official Journal of the European Communities*, 281:31, 1995.

[12] Roy Morgan research report for the office of the Federal Privacy Commissioner: Community attitudes towards privacy. http://www.privacy.gov.au/publications/rcommunity/index_print.html, 2004.

[13] U.S. department of health and human services office for civil rights. HIPAA administrative simplification regulation text. 2006.

[14] StorageTek software information guide. http://www.sun.com/storagetek/tape_storage/tape_media/volsafe/VolSafe.pdf, 2007.

[15] N. R. Adam and J. C. Worthmann. Security-control methods for statistical databases: a comparative study. *ACM Comput. Surv.*, 21(4):515–556, 1989.

[16] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB '05*, pages 901–909, 2005.

[17] C. C. Aggarwal and P. S. Yu. A condensation approach to privacy preserving data mining. In *EDBT '04*, pages 183–199, 2004.

[18] G. Aggarwal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu. Achieving anonymity via clustering. In *PODS '06*, pages 153–162, 2006.

[19] G. Aggarwal, F. Kenthapadi, K. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Approximation algorithms for k-anonymity. *Journal of Privacy Technology*, 2005.

[20] R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *SIGMOD '03*, pages 86–97, 2003.

[21] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB '94*, pages 487–499, 1994.

[22] R. Agrawal and R. Srikant. Privacy-preserving data mining. *SIGMOD Rec.*, 29(2):439–450, 2000.

[23] M. T. Almeida, G. Schütz, and F. D. Carvalho. Cell suppression problem: A genetic-based approach. *Comput. Oper. Res.*, 35(5):1613–1623, 2008.

[24] F. Bacchus, A. J. Grove, J. Y. Halpern, and D. Koller. From statistical knowledge bases to degrees of belief. *Artif. Intel.*, 87(1-2):75–143, 1996.

[25] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE '05*, pages 217–228, 2005.

[26] N. Beckmann, H. Kriegel, R. Schneider, and B. Seeger. The r*-tree: An efficient and robust access method for points and rectangles. In *SIGMOD '90*, pages 322–331, 1990.

[27] D. E. Bell and L. J. La Padula. Secure computer systems: Mathematical foundations. MTR-2547, Vol. I, The MITRE Corporation, Bedford, MA, 1 March 1973.

[28] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *ACM Symposium on Theory of Computing (STOC '88)*, pages 1–10, 1988.

[29] S. Berchtold, C. Bohm, and H. Kriegel. Improving the query performance of high-dimensional index structures by bulk-load operations. In *EDBT '98*, pages 216–230, 1998.

[30] E. Bertino, B. C. Ooi, Y. Yang, and R. H. Deng. Privacy and ownership preserving of outsourced medical data. In *ICDE '05*, pages 521–532, 2005.

[31] D. Bertsekas. *Constrained optimization and Lagrange Multiplier methods.* Computer Science and Applied Mathematics, Boston: Academic Press, 1982.

[32] C. Bettini, X. S. Wang, and S. Jajodia. The role of quasi-identifiers in k-anonymity revisited. *CoRR*, abs/cs/0611035, 2006.

[33] S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based constraint satisfaction and optimization. *J. ACM*, 44(2):201–236, 1997.

[34] J. Brickell and V. Shmatikov. Efficient anonymity-preserving data collection. In *KDD '06*, pages 76–85, 2006.

[35] J. Byun, A. Kamra, E. Bertino, and N. Li. Efficient k-anonymity using clustering technique. In *DASFAA '07*, pages 188–200, 2007.

[36] J. Byun, Y. Sohn, E. Bertino, and N. Li. Secure anonymization for incremental datasets. In *3rd Third VLDB Workshop on Secure Data Management*, pages 48–63, 2006.

[37] R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *ACM Symposium on Theory of Computing (STOC '96)*, pages 639–648, 1996.

[38] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90, 1981.

[39] B. Chen, R. Ramakrishnan, and K. LeFevre. Privacy skyline: Privacy with multidimensional adversarial knowledge. In *VLDB '07*, pages 770–781, 2007.

[40] C. A. Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation).* Springer-Verlag, 2006.

[41] W. H. E. Day. Foreword: Comparison and consensus of classifications. *Journal of Classification*, 3(2):183–185, 1986.

[42] D. Denning. *Cryptography and Data Security.* Addison-Wesley, 1982.

[43] J. Domingo-Ferrer and J. M. Mateo-Sanz. Practical data-oriented microaggregation for statistical disclosure control. *IEEE Trans. on Knowledge and Data Engineering*, 14(1):189–201, 2002.

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

[45] V. Filkov and S. Skiena. Heterogeneous data integration with the consensus clustering formalism. In *Data Integration in the Life Sciences (DILS '04)*, pages 110–123, 2004.

[46] M. Fischetti and J. J. S. Gonzalez. Models and algorithms for optimizing cell suppression in tabular data with linear constraints. *Journal of the American Statistical Association*, 95(451):916–928, 2000.

[47] J. Friedman, J. Bentley, and R. Finkel. An algorithm for finding best matches in logarithmic time. *ACM Trans. on Mathematical Software*, 3(3), 1977.

[48] B. C. M. Fung. *Privacy-Preserving Data Publishing.* PhD thesis, Simon Fraser University, Burnaby, BC, Canada, May 2007.

[49] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *ICDE '05*, pages 205–216, 2005.

[50] S. Garfinkel. *PGP: Pretty Good Privacy.* O'Reilly, 1994.

[51] G. W. Gates and N. A. Potok. Data stewardship and accountability at the u. s. census bureau. Federal Committee on Statistical Methodology: working paper.

[52] R. Gavison. Privacy and the limits of law. *Yale law journal*, 89(3):421–471, 1980.

[53] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *ICDCS '05*, pages 620–629, 2005.

[54] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Comput. Surv.*, 38(3):9, 2006.

[55] P. P. Griffiths and B. W. Wade. An authorization mechanism for a relational database system. *ACM Trans. Database Syst.*, 1(3):242–255, 1976.

[56] S. Guha, R. Rastogi, and K. Shim. Cure: an efficient clustering algorithm for large databases. In *SIGMOD '98*, pages 73–84, 1998.

[57] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD '84*, pages 47–57, 1984.

[58] J. A. Hartigan. *Clustering Algorithms.* John Wiley & Sons, Inc., 1975.

[59] S. Hettich and C. J. Merz. Uci repository of machine learning databases. 1998.

[60] B. Hore, S. Mehrotra, and G. Tsudik. A privacy-preserving index for range queries. In *VLDB '04*, pages 720–731, 2004.

[61] House of Commons Health Commity. The Electronic Patient Record. http://www.publications.parliament.uk/pa/cm200607/cmselect/cmhealth/422/422.pdf, 2007.

[62] B. A. Huberman, E. Adar, and L. R. Fine. Valuating privacy. *IEEE Security and Privacy*, 3(5):22–25, 2005.

[63] T. Iwuchukwu and J. F. Naughton. K-anonymization as spatial indexing: toward scalable and incremental anonymization. In *VLDB '07*, pages 746–757, 2007.

[64] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *KDD '02*, pages 279–288, 2002.

[65] E. T. Jaynes. *Probability Theory The Logic of Science.* Cambridge University Press, 2003.

[66] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE Trans. on Knowledge and Data Engineering*, 16(9):1026–1037, 2004.

[67] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *ICDM '03*, page 99, 2003.

[68] K. Kenthapadi, N. Mishra, and K. Nissim. Simulatable auditing. In *PODS '05*, pages 118–127, 2005.

[69] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *SIGMOD '06*, pages 217–228, 2006.

[70] A. Kobsa. Privacy-enhanced personalization. *Commun. ACM*, 50(8):24–33, 2007.

[71] N. Koudas, Q. Zhang, D. Srivastava, and T. Yu. Aggregate query answering on anonymized tables. In *ICDE '07*, pages 116–125, 2007.

[72] V. Koutsonikola and A. Vakali. LDAP: framework, practices, and trends. *IEEE Internet Computing*, 8(5):66–72, 2004.

[73] G. Ramesh L. V. S. Lakshmanan, R. T. Ng. To do or not to do: The dilemma of disclosing anonymized data. In *SIGMOD '05*, pages 61–72, 2005.

[74] E. Larkin. Anonymizer safe surfing suite. http://www.pcworld.com/article/id,128186/article.html, 2007.

[75] K. LeFevre and D. DeWitt. Scalable anonymization algorithms for large data sets. university of wisconsin computer sciences, technical report 1590. 2007.

[76] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In *SIGMOD '05*, pages 49–60, 2005.

[77] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *ICDE '06*, page 25, 2006.

[78] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Workload-aware anonymization. In *KDD '06*, pages 277–286, 2006.

[79] J. Li, R. Wong, A. Fu, and J. Pei. Achieving k-anonymity by clustering in attribute hierarchical structures. In *Data Warehousing and Knowledge Discovery (DaWaK '06)*, pages 405–416, 2006.

[80] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE '07*, pages 106–115, 2007.

[81] T. Li and N. Li. Towards optimal k-anonymization. *Data Knowl. Eng.*, 65(1):22–39, 2008.

[82] T. Li, M. Ogihara, and S. Ma. On combining multiple clusterings. In *ACM Conference on Information and Knowledge Management (CIKM '04)*, pages 294–303, 2004.

[83] G. Loukides and J. Shao. Capturing data usefulness and privacy protection in k-anonymisation. In *SAC '07*, pages 370–374, 2007.

[84] G. Loukides and J. Shao. Clustering-based k-anonymisation algorithms. In *DEXA '07*, pages 761–771, 2007.

[85] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *ICDE '06*, page 24, 2006.

[86] A. Machanavajjhala, D. Kifer, J. M. Abowd, J. Gehrke, and L. Vilhuber. Privacy: Theory meets practice on the map. In *ICDE '08*, pages 277–286, 2008.

[87] B. Malin, E. Airoldi, S. Edoho-Eket, and Y. Li. Configurable security protocols for multi-party data analysis with malicious participants. *ICDE '05*, pages 533–544, 2005.

[88] B. Malin and E.M. Airoldi. Confidentiality preserving audits of electronic medical record access. World Congress on Health (Medical) Informatics (MEDINFO), 2007.

[89] J. M. Mateo-Sanz, A. Martínez-Ballesté, and J. Domingo-Ferrer. Fast generation of accurate synthetic microdata. In *Privacy in Statistical Databases*, pages 298–306, 2004.

[90] A. McCallum, K. Nigam, and L. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *KDD '00*, pages 169–178, 2000.

[91] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *PODS '04*, pages 223-228, 2004.

[92] S. Monti, P. Tamayo, J. Mesirov, and T. Golub. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning*, 52(1-2):91–118, 2003.

[93] M. Naor and K. Nissim. Communication preserving protocols for secure function evaluation. In *ACM Symposium on Theory of Computing (STOC '01)*, pages 590–599, 2001.

[94] B. L. Narayan, C. A. Murthy, and S. K. Pal. Maxdiff kd-trees for data condensation. *Pattern Recogn. Lett.*, 27(3):187–200, 2006.

[95] M. E. Nergiz, M. Atzori, and C. Clifton. Hiding the presence of individuals from shared databases. In *SIGMOD '07*, pages 665–676, 2007.

[96] M. E. Nergiz and C. Clifton. Thoughts on k-anonymization. *Data Knowl. Eng.*, 63(3):622–645, 2007.

[97] J. Nin, J. Herranz, and V. Torra. Rethinking rank swapping to decrease disclosure risk. *Data Knowl. Eng.*, 64(1):346–364, 2008.

[98] A. P. Punnen. On combined minmax-minsum optimization. *Computers and Operations Reseach*, 21(6):707–716, 1994.

[99] V. Rastogi, S. Hong, and D. Suciu. The boundary between privacy and utility in data publishing. In *VLDB '07*, pages 531–542, 2007.

[100] S. P. Reiss. Practical data-swapping: the first steps. *ACM Trans. on Database Syst.*, 9(1):20–37, 1984.

[101] M. K. Reiter and A. D. Rubin. Anonymous web transactions with crowds. *Commun. ACM*, 42(2):32–48, 1999.

[102] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy. Extending query rewriting techniques for fine-grained access control. In *SIGMOD '04*, pages 551–562, 2004.

[103] S. J. Rizvi and J. R. Haritsa. Maintaining data privacy in association rule mining. In *VLDB '02*, pages 682–693, 2002.

[104] W. Rjaibi and P. Bird. A multi-purpose implementation of mandatory access control in relational database management systems. In *VLDB '04*, pages 1010–1020, 2004.

[105] D. B. Rubin. Discussion of statistical disclosure limitation. *Journal of Official Statistics*, 9(2):461–468, 1993.

[106] P. Samarati. Protecting respondents identities in microdata release. *IEEE Trans. on Knowledge and Data Engineering*, 13(9):1010–1027, 2001.

[107] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.

[108] R. Smith and J. Shao. Privacy and e-commerce: a consumer-centric perspective. *Electronic Commerce Research*, 7(2):89–116, 2007.

[109] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.

[110] L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.

[111] Z. Teng and W. Du. Comparisons of k-anonymization and randomization schemes under linking attacks. In *the Sixth International Conference on Data Mining (ICDM '06)*, pages 1091–1096, 2006.

[112] W. Trappe and L. C. Washington. *Introduction to Cryptography: With Coding Theory*. Pearson Education, 2005.

[113] T. M. Truta, A. Campan, and P. Meyer. Generating microdata with p-sensitive k-anonymity property. In *4th Third VLDB Workshop on Secure Data Management*, pages 124–141, 2007.

[114] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *KDD '02*, pages 639–644, 2002.

[115] J. Vaidya and C. Clifton. Privacy-preserving top-k queries. In *ICDE '05*, pages 545–546, 2005.

[116] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *ACM SIGMOD Record*, 3(1):50–57, 2004.

[117] J. Vogel. When cards come collecting. Seattle Weekly, September 23 1998.

[118] Y. Wang and Y. Zheng. Fast and secure magnetic worm storage systems. *the Second International Security in Storage Workshop (SISW '03)*, 2003.

[119] A. Westin. *Privacy and Freedom*. New York: Atheneum, 1967.

[120] A. Westin. Opinion surveys: What consumers have to say about information privacy. prepared witness testimony, the house committee on energy and commerce. 2001.

[121] R. Wong, J. Li, A. Fu, and K.Wang. alpha-k-anonymity: An enhanced k-anonymity model for privacy-preserving data publishing. In *KDD '06*, pages 754–759, 2006.

[122] M. Wright, M. Adler, B. N. Levine, and C. Shields. Defending anonymous communications against passive logging attacks. In *Proceedings of the 2003 Symposium on Security and Privacy (SP '03)*, page 28, 2003.

[123] X. Xiao and Y. Tao. Anatomy: simple and effective privacy preservation. In *VLDB '06*, pages 139–150, 2006.

[124] X. Xiao and Y. Tao. Personalized privacy preservation. In *SIGMOD '06*, pages 229–240, 2006.

[125] X. Xiao and Y. Tao. M-invariance: towards privacy preserving re-publication of dynamic datasets. In *SIGMOD '07*, pages 689–700, 2007.

[126] L. Xiong, S. Chitti, and L. Liu. Topk queries across multiple private databases. In *ICDCS '05*, pages 145–154, 2005.

[127] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. Fu. Utility-based anonymization using local recoding. In *KDD '06*, pages 785–790, 2006.

[128] Z. Yang, S. Zhong, and R. N. Wright. Anonymity-preserving data collection. In *KDD '05*, pages 334–343, 2005.

[129] A. C. Yao. How to generate and exchange secrets. In *IEEE Symposium on Foundations of Computer Science (FOCS '86)*, pages 162–167, 1986.

[130] C. Yao, X. S. Wang, and S. Jajodia. Checking for k-anonymity violation by views. In *VLDB '05*, pages 910–921, 2005.

[131] N. Zhang, S. Wang, and W. Zhao. A new scheme on privacy-preserving data classification. In *KDD '05*, pages 374–383, 2005.

[132] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: An efficient data clustering method for very large databases. In *SIGMOD '96*, pages 103–114, 1996.

[133] J. Zhou and J. Sander. Data bubbles for non-vector data: Speeding-up hierarchical clustering in arbitrary metric spaces. In *VLDB '03*, pages 452–463, 2003.

[134] Q. Zhu and W. W. Hsu. Fossilized index: the linchpin of trustworthy non-alterable electronic records. In *SIGMOD '05*, pages 395–406, 2005.