Data Utility and Privacy Protection Trade-off in K-Anonymisation

Grigorios Loukides
School of Computer Science
Cardiff University
Cardiff, UK
g.loukides@cs.cf.ac.uk

Jianhua Shao
School of Computer Science
Cardiff University
Cardiff, UK
j.shao@cs.cf.ac.uk

ABSTRACT

K-anonymisation is an approach to protecting privacy contained within a dataset. A good k-anonymisation algorithm should anonymise a dataset in such a way that private information contained within it is hidden, yet the anonymised data is still useful in intended applications. However, maximising both data utility and privacy protection in k-anonymisation is not possible. Existing methods derive k-anonymisations by trying to maximise utility while satisfying a required level of protection. In this paper, we propose a method that attempts to optimise the trade-off between utility and protection. We introduce a measure that captures both utility and protection, and an algorithm that exploits this measure using a combination of clustering and partitioning techniques. Our experiments show that the proposed method is capable of producing k-anonymisations with required utility and protection trade-off and with a performance scalable to large datasets.

1. INTRODUCTION

A vast amount of data about individuals is being collected and stored worldwide. Such data can contain private information about individuals, for example, their credit card numbers, shopping preferences and medical records. When the data is released for studies such as lifestyle surveys, business analysis and healthcare research, privacy protection becomes a serious concern. Unfortunately, simply removing unique identifiers (e.g. credit card numbers) from data is not enough, as individuals can still be identified using a combination of non-unique attributes such as age and postcode.

K-anonymisation has been proposed to address this concern [21]. Assume that a table T consists of two groups of attributes: *quasi-identifiers* (QIDs) that contain information that can potentially be used to identify individuals (e.g. age and postcode), and *sensitive attributes* (SAs) that contain sensitive information about individuals (e.g. their shopping preferences or diagnosed diseases). A *k-anonymisation* of T is a view of T such that each tuple in the view is made identical (through some form of data generalisation) to at least $k-1$ other tuples w.r.t. QIDs. For example, Table 1 is 3-anonymised w.r.t. *Age* and *Postcode*. Clearly, k-anonymisation helps prevent linking sensitive information to individuals but can affect data utility.

Id	Age	Postcode	Disease
t_1	[10-25]	[NW10-30]	Cancer
t_2	[10-25]	[NW10-30]	HIV
t_3	[10-25]	[NW10-30]	Flu
t_4	[10-25]	[NW10-25]	Bronchitis
t_5	[10-25]	[NW10-25]	Flu
t_6	[10-25]	[NW10-25]	Pneumonia
t_7	[10-30]	[NW20-30]	Bronchitis
t_8	[10-30]	[NW20-30]	Flu
t_9	[10-30]	[NW20-30]	HIV

Table 1: A 3-anonymisation of a dataset

For a given table, many k-anonymisations are possible. An ideal solution should maximise both data utility and privacy protection in anonymised data, but this is computationally not possible [18]. Various metrics have been proposed to capture what a good k-anonymisation should be and methods for deriving them heuristically [21, 15, 26]. A popular approach is to derive k-anonymisations that retain as much data utility as possible, while satisfying a minimum level of protection required [18, 16]. This is illustrated in Figure 1.

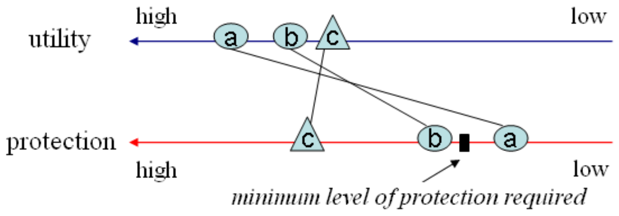

Figure 1: Utility-driven k-anonymisations

Here we have 3 possible k-anonymisations (a, b and c) of the same dataset. a will not be considered because it does not have the minimum level of protection required. Both b and c can be considered, but since b offers better data utility, existing methods attempt to produce b instead of c.

This approach is good for the data user as it seeks to maximise utility, but it may not represent the best interest of the data owner. Thus, alternative terms of utility/privacy trade-off need to be considered. Suppose, for example, that the data user is required to pay for a dataset based on its utility and is willing to pay an x amount for the dataset if an anonymisation has a required level of utility w.r.t. a certain measure, e.g. classification accuracy [13]. Knowing that the utility offered by anonymisation c is still sufficient for the user's requirement (see Figure 2), the owner may want to produce c instead of b, as it offers better protection while still satisfying the user's utility requirement. Another example comes from location privacy preserving systems, where how location information is anonymised through data generalisation must satisfy both quality of service and identification protection requirements [10]. Since providing a certain level of protection may result in unacceptable quality of service in such applications, a desired trade-off between utility and privacy requirements is necessary.

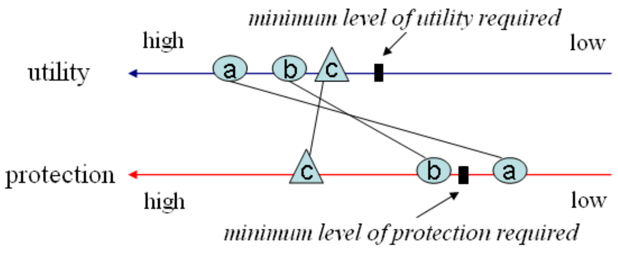
minimum level of utility required
high
low
utility
a
b
c
protection
c
b
a
high
low
minimum level of protection required

Figure 2: Tradeoff-driven k-anonymisations

In fact, data users and owners have often conflicting requirements, thus detailed control of the level of utility and protection of anonymised data is necessary to help them reaching a compromise [9]. In this paper, we consider how k-anonymisations with an optimal trade-off between data utility and protection may be produced, providing anonymisers with an intuitive mechanism to choose a different trade-off in case of different utility/protection requirements.

Our first contribution is a distance-based quality criterion that handles both QIDs and SAs on equal terms, a key requirement to trade-off utility and protection when k-anonymising data. We show that our notion of distance on QIDs attempts to capture the accuracy of aggregate query answering, one of the main tasks anonymised data is used for [15, 25] and generalises many existing utility measures [17, 7]. We also explore the relation between adversarial background knowledge and the protection offered by k-anonymisation, based on a more general definition of protection than that of [18, 16, 24, 25]. More specifically, we consider *range disclosure*, the case where sensitive information is estimated given ranges of values. For example, the second group in Table 1 includes $\{Bronchitis, Flu, Pneumonia\}$, allowing an attacker to estimate that an individual suffers a respiratory problem (see Figure 3). Such inferences are not prevented by existing protection mechanisms, posing a privacy danger in many real-world settings [23]. In response, we propose an intuitive and accurate probabilistic measure to capture protection under this threat model.

We also design an efficient heuristic algorithm for k-anonymising data with a desired utility/privacy trade-off. Our algorithm optimises the weighted sum of the amount of generalisation of QIDs and the amount of protection of SAs, as they are captured using our measure. Furthermore, it uses a flexible multi-dimensional local recoding strategy and combines clustering with partitioning to achieve both quality and efficiency in the k-anonymisation process. Our analysis and experimental results show that our approach is able to produce k-anonymisations with required trade-off between utility and protection and with a performance scalable to large datasets.

The paper is structured as follows. In Section 2 we discuss how data utility and privacy protection are captured. We present our method in Section 3 and experimentally evaluate it in Section 4. We briefly discuss the related work in Section 5 and conclude in Section 6.

2. MEASURING UTILITY AND PROTECTION

We first explain how data utility may be measured. Observe that data utility loss may occur when a table is k-anonymised through data generalisation. For example, the following query can no longer be answered accurately using Table 1, since *Age* and *Postcode* values have been generalised into ranges (note that we treat postcodes as interval values here):

```
Q1: select count(*)
    from   Table 1
    where  age > 25
    and    postcode between NW25 and NW30
```

However, we can attempt to answer Q1 approximately using Table 1. Assuming uniform distribution of data, we can calculate the probability of a tuple in Table 1 being in the answer to Q1 as $p = \frac{Rq \cap R}{R}$, where R_q and R denote the areas covered by Q1 and the qualifying tuples in Table 1 respectively, and $Rq \cap R$ denotes the overlap between R_q and R. For example, R_q in this case represents an area that is bounded by $Age : [26, 30]$ and $Postcode : [NW25, NW30]$. Since we have 3 tuples in Table 1 satisfying the search condition of Q1, the estimated answer to Q1 is $3 \times p$. Intuitively, therefore, maximising the overlap between queries and their coverage by anonymised data is desirable.

More generally, we observe that k-anonymisation partitions a table into groups, and each group is represented by the *minimum bounding rectangle* (MBR) of an m-dimensional hyper-rectangle [3], where m is the number of QIDs, and each side (and its length) correspond to a QID (and the range of its generalised values) in the table. Since R_q is not known when the data is anonymised, it makes sense that we should minimise R, or the MBR of a group, in order to maximise p. Our utility measure is built on this observation.

DEFINITION 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*

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

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 hierarchy 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 an interval-based QID, ranges are naturally captured using the Euclidean distance. For a discrete attribute, however, there is no ordering among its values and distance between them is often defined in terms of their semantic relationships using a user-defined hierarchy [21]. For example, Figure 3 shows one such hierarchy for *Disease*. Given a hierarchy, distance between a group of discrete values can then be defined as the ratio of the number of leaves of the subtree rooted at the closest common ancestor of these values to the domain size [26]. If there is no hierarchy for a discrete attribute, we assume that the distance between any pair of values is equal. Q-diversity can handle all different types of QIDs uniformly, being more general than the Maxsize [7] and Usefulness [17] utility measures, which cannot handle ordered discrete attributes.

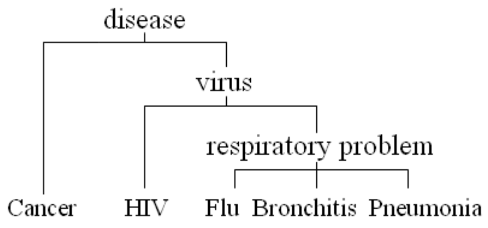

Figure 3: A hierarchy for *Disease*

Based on Q-diversity, we define data utility to be the average Q-diversity across groups over all QIDs, assuming attribute independence. Intuitively, utility reflects the average amount of generalisation that each group of tuples incurs, and a small score is preferred.

DEFINITION 2.2 (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 of g_j will have the same values in each QID after anonymisation. The utility of T under this clustering is defined as*

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

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 $1 \leq i \leq m$.

To explain how privacy protection is captured, we need to consider how sensitive information about individuals may be learned from a k-anonymised table. For our analysis, we assume that information about an individual I is modelled as a single tuple t_I in the anonymised dataset (the reader is referred to [25] for models considering multiple tuples), the attacker has identified the group (called *covering group*) that contains t_I, and he attempts to determine the SA value s_I for I. We note that in practice it is often sufficient for the attacker to obtain the SA value within a range, e.g. salary in \$40-42K or a disease being "virus", in order for the privacy to be considered breached. We therefore allow ranges to be used to model attackers' *estimation power* in our measure of protection.

Some existing frequency-based protection measures, such as l-diversity [18], (a, k)-anonymity [24] and p-sensitive k-anonymity [22], consider attackers with zero estimation power. Thus, they do not prevent range disclosure, since groups with a very small range can be formed, even though salary values are distinct. T-closeness [16] requires grouping SA values with a distribution similar to that of the SA value in the whole dataset, effectively assuming attackers with maximum estimation power when these two distributions are identical. However, SA values can still be accurately estimated, when the original distribution allows such inferences for example.

We assume that the attacker attempts to determine s_I within a specified range having width r and model the protection for I's sensitive information as the probability of a range r covering s_I being disclosed. Observe that a tuple t_j in the covering group for t_I (including t_I itself) contributes to attacker's estimate *only* when the distance between t_j and t_I's SA values is less than or equal to r. For example, when r includes *"respiratory problem"*, all pairs containing values *"Flu"*, *"Bronchitis"* and *"Pneumonia"* can be estimated. We capture this as *pairwise contribution (pc)*. Intuitively, a larger pc score implies a smaller distance between t_j and t_I's SA values, and t_j contributes more to the disclosing of s_I.

DEFINITION 2.3 (PAIRWISE CONTRIBUTION). *Assume that a is an SA, the domain of a is D_a, and v_i, v_j are a pair of values obtained from a. The pairwise-contribution for v_i, v_j is defined as:*

$$pc(v_i, v_j) = \begin{cases} 1 - \frac{|v_i - v_j|}{max(D_a) - min(D_a)} & \textit{interval values} \\ 1 - \frac{s(\{v_i, v_j\}) - 1}{|D_a|} & \textit{discrete values} \end{cases}$$

$max(D_a)$, $min(D_a)$, $s(\{v_i, v_j\})$ and $|D_a|$ have same definitions as those in Definition 2.1.

Assuming that each sensitive value can be estimated equally likely for any individual in a group [25], we measure the protection of sensitive information for the group as the ratio of total contribution the attacker obtains from the SA values in the group for all individuals, to the maximum contribution that the attacker could possibly have from the group. We call this measure *S-diversity*.

DEFINITION 2.4 (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 for a given range r, denoted by $sd(V_a, r)$, is defined as:*

$$sd(V_a, r) = \frac{\sum_{\forall v_i, v_j \in V_a s.t. pc(v_i, v_j) \geq (1-r)} pc(v_i, v_j)}{|V_a|^2}$$

where $pc(v_i, v_j)$ is given in Definition 2.3 and r is normalised to $[0, 1]$.

S-diversity has two important properties. First, it can be applied to SA values in any metric space. Consider for example, $g_1 = \{t_1, t_2, t_3\}$ and $g_2 = \{t_4, t_5, t_6\}$ in Table 1 and their corresponding sensitive values in *Disease*. Both groups are considered to be equally protected by some existing measures, such as l-diversity, which handle only unordered discrete SAs. However, if we consider the distance between these values according to the hierarchy shown in Figure 3 and assume $r = 0.5$, then the probabilities of disclosing any SA value in g_1 and g_2, measured by S-diversity, are $\frac{3 \times 1}{3^2} = 0.333$ [1] and $\frac{3 \times (1 + 2 \times (1 - \frac{3-1}{5}))}{3^2} = 0.733$ respectively.

[1]The pairs $(t_1, t_1), (t_2, t_2), (t_3, t_3)$ have $pc \geq 0.5$ and thus can be estimated. Specifically, these pairs have $pc = 1$, hence the S-diversity is $\frac{3 \times 1}{3^2} = 0.333$.

This suggests that g_2 offers less protection than g_1 does, and is justified as the SA values in g_2 are not as diverse as those in g_1, i.e. they are all respiratory problems (see Figure 3). Second, attacker's estimation power can be controlled by varying r, as opposed to t-closeness which effectively sets r to 1 for all cases. Consider $g_3 = \{t_7, t_8, t_9\}$ in Table 1, for example. g_3 is considered to offer more protection when $r = 0.5$ than when $r = 0.7$ using the S-diversity measure. Again, this is justified because when $r = 0.5$ (a narrower estimate range is used), the value *"HIV"* of t_9 is not considered to help the SA values of t_7 and t_8 to be inferred. In contrast, t-closeness assumes that all SA values can help an attacker's estimation and thus considers g_3 as equally protected in both cases.

Based on this measure, the protection for a table is defined as the average S-diversity of all its groups over SAs. A small score means that the values in SAs are far apart from one another and thus are better protected.

Definition 2.5 (Protection). *Assume that a table T comprised of s SAs $\{a_1, \ldots, a_s\}$ 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 protection of T under this clustering is defined as*

$$protection = avg(\sum_{i=1}^{s} sd(\pi_{a_i}(g_1), r), \ldots, \sum_{i=1}^{s} sd(\pi_{a_i}(g_h, r)))$$

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

Finally, the discussion so far assumed a single covering group for an individual, but overlapping groups can be created in k-anonymisation, as a result of using multidimensional recoding [15]. In this case, individuals may achieve more (but not less) protection than that measured by S-diversity. Also, we note that weights may be used to reflect the significance of each attribute. Due to space restrictions, we do not discuss these further in this paper.

3. GROUPING DATA

In this section, we present our method that combines clustering with partitioning. That is, we first partition data into suitable sub-spaces, and then cluster data in each sub-space separately. We explain how both quality and efficiency may be achieved in generating k-anonymisations with utility and protection trade-off.

3.1 Threshold-Based Greedy Clustering

Clustering algorithms use heuristics and perform greedy search in grouping data. The existing clustering heuristics and search strategies attempt to optimise data utility [26, 5], and therefore are not useful for our purpose. To derive k-anonymisations with "optimal" trade-off between data utility and protection, we propose the following heuristic.

Definition 3.1 (Weighted Tuple diversity). *Let $\tau \subseteq T$ be a set of tuples over a set of attributes $A = \{a_1, a_2, \ldots, a_p\}$. Without loss of generality, we assume that the first m attributes are QIDs and the rest are SAs. The* weighted tuple diversity *of τ w.r.t. A, denoted by $wtd(\tau, A)$, is defined as:*

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

where $\pi_{a_i}(\tau)$ denotes the projection of τ on attribute a_i, $w_u, w_p \in [0, 1]$ are the weights, and we require $w_u + w_p = 1$.

The idea of optimising the weighted sum of an information-loss and a protection measure is not new. It has been successfully applied in [17]. However, our formulation has two interesting properties. First, we achieve a much stronger notion of protection. This is because, as opposed to [17], S-diversity can effectively prevent range disclosure. Second, attributes of any type are treated uniformly, therefore datasets with mixed attributes can be handled. However, deriving optimal k-anonymisations using our heuristic, as specified in Definition 3.2, is NP-hard (proof follows [1]).

Definition 3.2 (Optimal clustering). *Let T be a table consisting of n tuples and p attributes $A = \{a_1, a_2, \ldots, a_p\}$. Given a set of user specified weights w_u, w_p and a range r as defined in Definitions 3.1 and 2.4 respectively, an* optimal clustering *of T is a partition $P = \{c_1, \ldots, c_h\}$ of T such that $|c_j| \geq k, j = 1, \ldots, h$, $\bigcap_{j=1}^{h} c_j = \emptyset$, $\bigcup_{j=1}^{h} c_j = T$, and its average weighted tuple diversity $avg\_wtd(T, A) = \frac{\sum_{j=1}^{h} wtd(c_j, A)}{h}$ is minimal.*

Thus, we introduce a heuristic *threshold-based greedy clustering* algorithm (shown in Algorithm 1) which works as follows. It randomly picks up a tuple t_i as the *seed* of a cluster and removes it from T in step 2. In steps 3-7 it repeatedly finds the closest tuples to this cluster, one at time, and adds them into the cluster until a threshold is reached. We find a tuple t_j that is closest to the cluster in step 4 by computing $wtd(\{c \cup \{t_j\}\}, A)$ according to Definition 3.1. If adding t_j into the cluster will not cause a threshold (δ), which controls the maximally-allowed weighted tuple diversity in a group, to be exceeded (step 6), then t_j will be added into the cluster and removed from T (step 7). If δ is exceeded, then the size of the created cluster is checked in step 8. If it is greater than k, tuples in the cluster are anonymised using a local recoding function, else this cluster is rejected. The process is repeated until all tuples of T are processed. We treat rejected clusters as bad local minima (the reason will be given in Section 3.2). Thus, we un-cluster tuples that belong in rejected clusters after clustering, and put each one into the cluster that will result in a minimal increment in weighted tuple-diversity when the tuple is inserted.

Algorithm 1 Threshold-Based Greedy Clustering

```
1. while $T \neq \emptyset$ do
2.     $c \leftarrow t_i \in T$; $T \leftarrow T - \{t_i\}$;
3.     while true do
4.         find $t_j \in T$ s.t. $wtd(\{c \cup \{t_j\}\}, A)$ is minimum;
5.         $c' \leftarrow c \cup \{t_j\}$;
6.         if ($wtd(c', A) > \delta$)   exit;
7.         $c \leftarrow c'$; $T \leftarrow T - \{t_j\}$;
8.     if ($|c| \geq k$) $k$-anonymise($c$) else reject $c$;
```

The proposed algorithm differs from existing clustering-based methods [5, 26, 6] in two main ways. First, it forms clusters based on both data utility and protection, thus it does not simply try to optimise utility. Second, it uses a quality-based instead of a size-based stopping criterion. Since the quality of anonymisations is not solely determined by the size of clusters, measuring maximally-allowed information diversity within a group is better and more meaningful. On performance, Threshold-Based Greedy Clustering has a time complexity that is similar to existing clustering-based methods, and is quadratic to the cardinality of the dataset.

3.2 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 [11], top-down bisection [26] or a cheap pre-clustering step [19]. 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 [8], which partitions data recursively into subspaces along the median of QID with the largest normalised domain. Algorithm 2 shows our method. Given a set of tuples τ (initially the entire table T), a size constraint s and a protection constraint δ', 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[2] (steps 3-5) until τ cannot be further divided without violating constraints on the minimum partition size k and maximum S-diversity δ' (steps 6-7).

Algorithm 2 Median-Based Pre-Partitioning

```
1. **Pre-partition** $(\tau, s, \delta')$
2.     find attribute $a_j \in QID$ s.t.
           values in $a_j$ have the largest range;
3.     $splitVal \leftarrow find\_median(\pi_{a_j}(\tau))$;
4.     $p \leftarrow \{t \in \tau : \pi_{a_j}(t) \leq splitVal\}$;
5.     $p' \leftarrow \{t \in \tau : \pi_{a_j}(t) > splitVal\}$;
6.     **if** ($|p| \geq s$ and $|p'| \geq s$ and
           $\sum_{i=1}^{s} sd(\pi_{a_i}(p), r) > \delta'$and $\sum_{i=1}^{s} sd(\pi_{a_i}(p'), r) > \delta'$)
7.         Pre-partition$(p, s, \delta')$ $\bigcup$ Pre-partition$(p', s, \delta')$;
8.     **else** return $\tau$;
```

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 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.

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

Theorem 3.1 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 [8], requiring only $O(nlog(n))$ time to execute.

THEOREM 3.1. *Pre-partitioning of a dataset T reduces the complexity of the threshold-based clustering algorithm to $O(s \times n)$, where n is the size of the dataset and s the minimum size of the resultant subspaces.*

PROOF. *(Sketch)* Let $P = \{p_1, ..., p_m\}$ be a partition of T created by applying Algorithm 2. The worst-case time complexity of the clustering-based algorithm when applied to each p_i separately is $O(\sum_{i=1}^{m} |p_i|^2)$, which is minimised to $O(m \times |p_i|^2)$, when all p_i are equal-sized for a fixed m. Since we have $\sum_{i=1}^{m} |p_i| = m|p_i| = n$, the complexity of the clustering-based algorithm with pre-partitioning becomes $O(|p_i| \times n) \approx O(s \times n)$. $\square$

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 (see Algorithm 1). 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 [4]. 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 as well. Consider applying pre-partitioning to the data shown in Table 2 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. This created the anonymisation shown in Table 3, which clearly incurred more information loss if we compare it to that shown in Table 4 which is derived by using clustering alone. This is because, as illustrated in Figure 4, good groups depicted as small rectangles in the 3D space of $\{Age,Height,Postcode\}$, intersect the subspaces $\{S_1, \ldots, S_4\}$ 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.

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. Unclustering tuples of rejected clusters also helps, as they can be grouped together with their

Age	Height	Postcode	Sal.
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 2: Microdata

Age	Height	Postcode	Sal.
[10-15]	[170-175]	[NW30-32]	20
[10-15]	[170-175]	[NW30-32]	20
[15-20]	[170-175]	[NW30-32]	65
[15-20]	[170-175]	[NW30-32]	65
[20-25]	[170-175]	[NW30-32]	20
[20-25]	[170-175]	[NW30-32]	20
[25-30]	[170-175]	[NW30-32]	65
[25-30]	[170-175]	[NW30-32]	65

Table 3: Partitioning Table 2

Age	Height	Postcode	Sal.
[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 4: Clustering Table 2

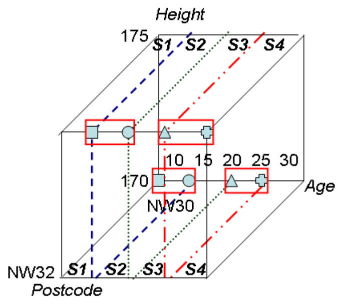

Figure 4: Intersection between clusters and subspaces when $s = k$

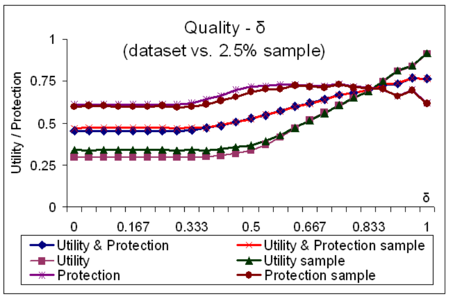

Figure 5: Utility and protection (dataset vs. sample)

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. having an MBR whose perimeter is smaller than that of the MBR of their corresponding subspace) and similar-shaped clusters are less likely to intersect a subspace created by median-based partitioning [20]. Our clustering step specifically minimises the perimeter of the MBR 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 [15], can create subspaces with an arbitrarily low level of protection, even when subspaces are large. For example, the subspaces shown in Table 3 have all the same value in *Salary* and thus offer no protection. We have therefore introduced a protection constraint in Algorithm 2, avoiding to further split a subspace when the sum of S-diversity over all SAs exceeds δ' in either of the resultant subspaces. This makes our partitioning aware of the trade-off between utility and protection and helps subsequent clustering.

As our pre-partitioning strategy remedies the problem of overlapping and unprotected subspaces, pre-partitioning serves well the purpose of maintaining quality of anonymisations produced by our clustering step. Our experimental results show that using an s just 5 times larger than k suffices to maintain the quality of clustering, while improving the performance of anonymisations by many orders of magnitude.

3.3 Thresholds selection

In this section, we discuss how the thresholds used in our algorithm can be set. In order to determine the thresholds δ and δ', we have developed a simple and efficient heuristic, which is based on the following observation. Groups derived by applying clustering on the whole dataset have similar quality to those derived by applying clustering on a sample of this dataset taken using random sampling with replacement, when the same δ is used. This is because groups are typically of similar size and thus are not likely to be missed by sampling. Moreover, even though some groups are missed, quality is not significantly affected, as δ still allows a good grouping for most clusters. Figure 5 depicts the utility and protection measures of applying clustering setting $k = 5, w_u = w_p, r = 1$ and varying δ on the Adults dataset [12] and on a 2.5% sample of it. Observe that both utility and protection measures are well reflected in the sample of data. Based on these observations, we propose the heuristic given in Algorithm 3.3.

Algorithm 3 Sampling-based selection heuristic for δ.

```
1. $S \leftarrow sample(T)$; $i \leftarrow 1$;
2. while $i < max\_iterations$ do
3.     split $[0, 1]$ into $2^i$ equal-length intervals;
4.     choose the mean of each interval as $\delta_r$;
5.     for each $\delta_r$ do
6.         apply Algorithm 1 to $S$
7. choose $\delta_r$ s.t. the sum of utility and
       protection for $S$ is minimum
```

The basic idea is that we run our clustering algorithm on a small random sample of the dataset several times, each time by changing the value of δ, so as to investigate the difference in clusters quality (as it is evaluated by the sum of utility and protection scores). Then, we choose the δ which results in the best clustering. Furthermore, δ' is set to the level of *protection* corresponding to the best clustering found before. This is because the mean and standard deviation of pairwise distances between SA values do not significantly vary across different subspaces when pre-partitioning is used with an s larger than k, as verified by our experiments. We note that a sample less than 2.5% and 4-6 iterations were enough to find fairly good values for thresholds δ and δ' for various levels of k very efficiently. We have also experimentally found that using $s = 5 \times k$ is a fairly good choice for s as mentioned in Section 3.2.

4. EXPERIMENTAL EVALUATION

In this section, we report experimental results. We configured three versions of our algorithm: **TGCU** (Threshold-based Greedy Clustering Utility with $w_u = 1$ and $w_p = 0$) for optimal utility, **TGCP** (Threshold-based Greedy Clustering Protection with $w_u = 0$ and $w_p = 1$) for optimal protection, and **TGCB** (Threshold-based Greedy Clustering Balance with $w_u = 0.5$ and $w_p = 0.5$) for a balanced utility and protection. We compare them to three benchmark algorithms: Mondrian [15] is partition-based and is very efficient; K-Members [5] is clustering-based and can achieve very good data utility; and K-Members-p is a modified version of K-members where we changed its objective function to optimise protection instead of utility, so it can achieve very good protection. Both Mondrian and K-members have been shown to achieve better utility than the global recoding algorithm used by [18, 16, 24], so this algorithm was excluded from our experiments. Our objective is to investigate how the quality of solutions with a trade-off between utility and protection compares to those produced by methods that specifically optimise one of these properties. We also test the impact of parameters used by our method on data quality and examine its efficiency.

4.1 Quality of Anonymisation

For these experiments, we used the Adults dataset [12] which has become a benchmark for k-anonymisation. The dataset is comprised of 8 attributes and 30162 tuples. We configured the dataset as in [13], but left out *Education* and treated *Marital status* and *Occupation* as SAs. *Occupation* was transformed to an interval SA, in the same way as described in [26]. We configured the parameters of our algorithms as follows: subspace size threshold $s = 5 \times k$, thresholds δ and δ' were set by the method discussed in Section 3.3, and estimation range $r = 1$.

For data utility, we compare our method to others in terms of our utility measure (UM), discernability measure (DM) [2] and average relative error (A.R.E.) (the average relative difference between the actual and estimated query result for all queries, when applied to the generalised dataset) [15]. As can be seen in Figure 6, TGCB substantially outperformed Mondrian in terms of UM, achieving a result that is very close to that of K-Members and TGCU. TGCP and K-Members-p performed poorly, as expected, as they do not optimise data utility. Furthermore, all versions of our algorithm achieved a low DM value as shown in Figure 7, even though we do not restrict the size of a group like Mondrian and K-Members do.

For the A.R.E. test, we examined two types of COUNT-query. A type-1 query retrieves the number of tuples satisfying randomly chosen range predicates on $\{Age, Gender, Income\}$, while a type-2 query additionally uses predicates involving $\{Race, WorkClass\}$. The selectivity was 1.5% and 0.5% for type-1 and type-2 queries respectively. We used a workload of 10000 queries for each query type. Estimated results were derived by computing the probability p for each tuple, as explained in Section 2, and summing up the probabilities for all tuples.

Figure 8 reports the *normalised error* in A.R.E. for Mondrian, K-Members and the three configurations of our algorithm, for various k levels. That is, the normalised difference in A.R.E., $\frac{x-b}{a-b}$, where x is the A.R.E. of a method, and a, b are the A.R.E. achieved by the best and worst A.R.E. values for a specific k. K-Members-p performed very poorly in this test since it does not optimise utility, and thus was not included in the computation of the normalised error values. K-Members achieved small A.R.E. values for type-1 queries when k was small, whereas Mondrian did well when k is large. Interestingly, TGCU outperformed both Mondrian and K-Members for all values of k. This is largely attributed to the fact that both Mondrian and K-Members use size-based stopping criteria when forming groups. As a result some "distant" tuples were brought into a group to satisfy size requirement, thereby increasing information loss. It is worth noting that TGCB performed particularly well, being comparable to that of TGCU. Furthermore, TGCP performed well only when k was small, as achieving protection comes at an increased cost of utility when k is large. Then, we examined type-2 queries, which are more difficult to be accurately answered than type-1 queries due to the higher dimensionality and lower selectivity. As illustrated in Figure 9, all configurations of our algorithm outperformed both Mondrian and K-members for type-2 queries. This validates

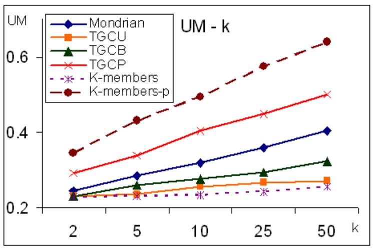

Figure 6: UM comparison (Adults)

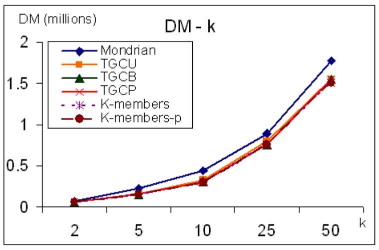

Figure 7: DM comparison (Adults)

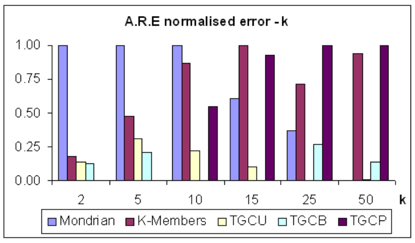

Figure 8: Normalised error in A.R.E. for type-1 queries (Adults)

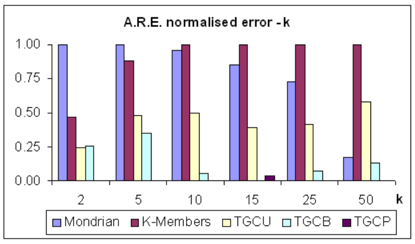

Figure 9: Normalised error in A.R.E. for type-2 queries (Adults)

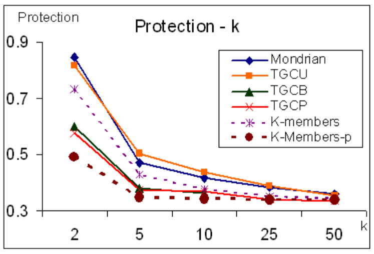

Figure 10: Protection comparison (Adults)

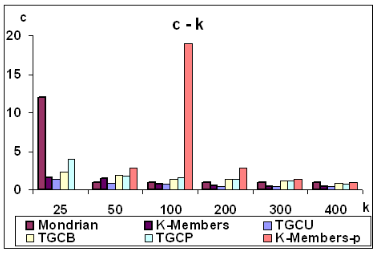

Figure 11: c value in (c,l)-diversity (Adults)

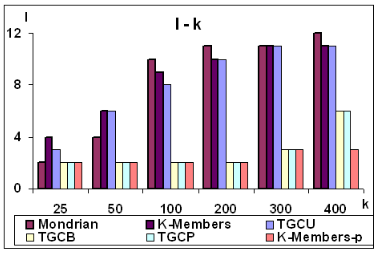

Figure 12: l value in (c,l)-diversity (Adults)

that using clustering with a quality-based stopping criterion can preserve correlations between QIDs better, achieving more utility.

For the same runs, we also evaluated protection using our *protection* measure. Comparing Figures 6 and 10, it is easy to see that optimising data utility adversely affected protection. To compare our method with benchmark algorithms using the (c,l)-diversity measure [18] for protection, we used a sample created by taking the first 5000 tuples of the Adults dataset and treated *Occupation* as the single, discrete SA. Here, l is the number of distinct SA values in a group and c is the fraction of the occurrences of the most frequent SA value in the group and the sum of the occurrences of remaining $l-1$ SA values. Figures 11 and 12 illustrate the results. While Mondrian, K-Members and TGCU created diverse groups, similar SA values were put together and thus protection was low. In contrast, TGCB, TGCP and K-Members-p created groups with few but semantically distant SA values, none of which appeared too frequently.

In summary, it is evident that the combined use of clustering and partitioning is beneficial: TGCU outperforms both Mondrian and K-Members in query answering, while TGCP achieves comparable protection to that of K-Members-p. Interestingly, TGCB demonstrated a performance that is very close to that of TGCU and TGCB, confirming the opportunity for utility and protection trade-off.

4.2 Effect of Data Skewness

It is known that partitioning may affect data utility when data is skewed, as it creates overlapping subspaces [4]. We tested our algorithm on a number of skewed distributions and found that skewness had little effect on data utility. Restricted by space, we only report the result using a synthetic dataset (comprised of 8000 tuples and 5 attributes) with standard normal distribution. The parameters of our method were set as in Section 4.1 and we evaluated the result using UM. As illustrated in Figure 13, TGCU achieved a result very close to that of K-Members and the performance of TGCB was comparable to that too.

Furthermore, it is more likely that a large number of semantically close SA values are grouped together when data is skewed. Thus, Mondrian, K-Members and TGCU achieved very low protection, as shown in Figure 14. In contrast, TGCB was able to trade-off some utility for better protection and thus its result is comparable to K-Members-p and TGCP. We should note, however, that much utility has to be traded-off to achieve strong protection in presence of very high skewness, and alternative techniques based on SA generalisation [25] might be helpful.

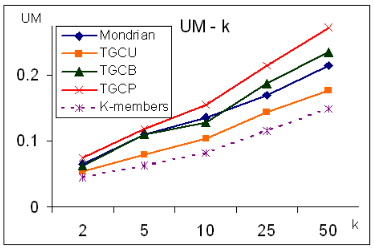

Figure 13: Utility comparison (Normal)

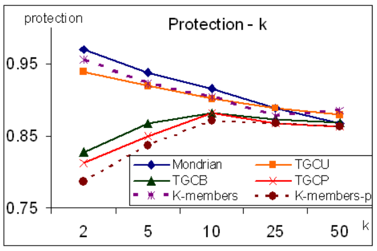

Figure 14: Protection comparison (Normal)

4.3 Effect of Parameters

We first considered the effect of the subspace size threshold s used in pre-partitioning. In this experiment, we varied s from 5 to the size of dataset and used TGCB with $k = 5$. As illustrated in Figure 15, the quality of anonymisations improves as s increases, only until the size of subspace reached a certain threshold ($s = 5k$), and remained stable after this point. This is because the effect of overlapping subspaces is not significant when s becomes much larger than k. Thus, a small s can be used in our method to generate high quality anonymisations efficiently. We then repeated the same experiment on the synthetic dataset used in Section 4.2. Quality again stopped to improve after $s = 5k$, as shown in Figure 16. We in fact have observed similar behaviour for different values of k and using other quality metrics, but restricted by space we do not report these results here.

We also studied how the range r used in S-diversity (see Definition 2.4) affects quality. We used TGCB on the synthetic dataset with the same parameters as in the previous experiment, treating the SA as interval-based and varying r.

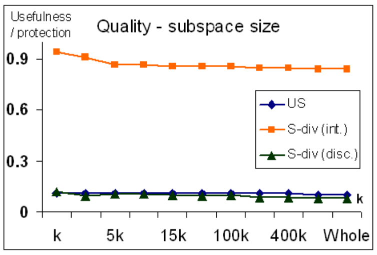

Figure 15: The impact of s threshold (Adults)

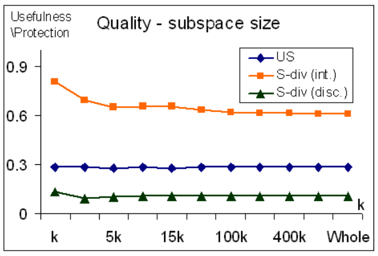

Figure 16: The impact of s threshold (Normal)

As shown in Figure 17, a larger r helped TGCB to trade-off protection for utility. This confirms the benefit of controlling the expected attacker's estimation power offered by our protection measure.

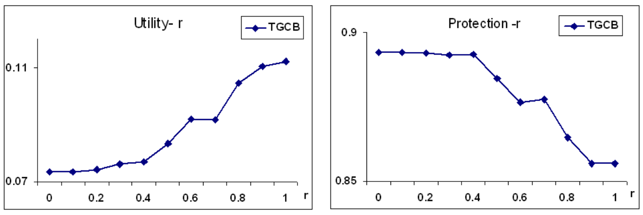

Figure 17: The impact of r in quality (Normal)

Finally, we tested the impact of weights w_u, w_p in quality on the synthetic dataset. All parameters were set as in the previous experiment, except k which was set to 10. Figure 18 illustrates the result. As can be seen, weights are able to lead the algorithm finding a desired trade-off.

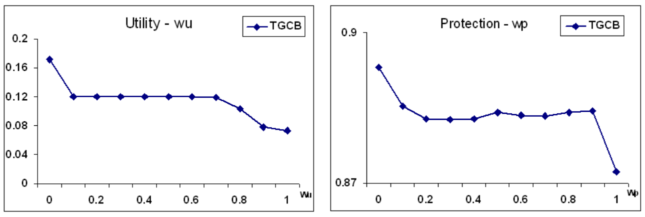

Figure 18: The impact of weights in quality (Normal)

4.4 Efficiency of Computation

The efficiency of our algorithm was studied and compared to the other two methods. For this experiment, we fixed $k = 5$ and ran the algorithms on samples of the Adults dataset with size ranging from 500 to 10000. The results are shown in Figure 19. We observe that K-Members and TGCB without pre-partitioning were slow due to their quadratic time complexity. In contrast, Mondrian is the fastest for its log-linear time complexity. However, the runtime of TGCB (with the pre-partitioning step) is very close to that of Mondrian, when the subspace size s is $5k$, as in our experiments. According to Theorem 3.1, the use of pre-partitioning reduces the time complexity of clustering to $O(s \times n)$, and thus the overall time complexity of our method becomes $O(n \times log(n) + n \times s) \approx O(k \times n)$, when $log(n) < s$ and $s \approx k$. As typically k is a small constant, our algorithm scales very well to large datasets. In order to further test this behaviour, we fixed $k = 5$ and $n = 10000$ and varied s from $2k$ to the size of the dataset. As can be seen in Figure 20, the runtime of our algorithm is not practically affected by s when $s < 100k$, achieving a log-linear perfomance to n.

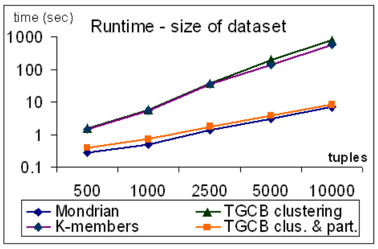

Figure 19: Runtime versus cardinality of dataset

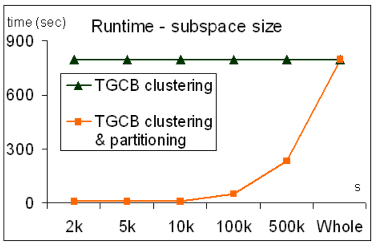

Figure 20: Runtime versus subspace size

5. RELATED WORK

Various measures for determining the quality of a k-anonymisation exist. For data utility, they typically quantify information loss resulted from data generalisation [7, 21, 13, 26], or accuracy degradation when performing certain tasks, e.g. query answering [15] or building classifiers [2], using anonymised data. Our utility measure captures information loss and is similar in principle to [17, 7], but it is more general in that it can be applied to any type of attribute. For protection, Machanavajjhala et al. [18] proposed l-diversity, a frequency-based measure treating SAs having unordered discrete values. Frequency-based measures have also been proposed by Truta et al. [22] and Wong et al. [24]. Li et al. [16] proposed t-closeness, a measure based on data distribution in SAs. All such measures assume that attackers have either zero or maximum estimation power and they are applicable only to single SAs. As a result, none of them is able to protect SA values from being estimated. Our protection measure protects ranges from being disclosed, allowing varying estimation power to be modelled and is applicable to multiple SAs.

K-anonymising data can be achieved through a number of techniques including micro-aggregation [6] and generalisation [21]. Generalisation-based algorithms can be classified into two groups. Some follow the global recoding model, mapping the domain of QIDs into generalised values [14]. As far as achieving trade-off between utility and protection is concerned, these algorithms are rather limited in that they only search QIDs for data groupings with optimal utility and protection requirement is only checked as a constraint that the resultant k-anonymisation must satisfy. The second category of algorithms uses local recoding, which maps the values of individual tuples into generalised ones on a group by group basis. Most existing algorithms search the QID space for data utility optimisation only, while in our previous work [17] we consider balancing utility with a basic protection requirement. All of these algorithms suffer from poor scalability [26, 5, 17]. Our method also uses local recoding, but is driven by a heuristic that attempts to optimise the trade-off between utility and protection from range disclosure and is done very efficiently as a result combining clustering with partitioning.

6. CONCLUSIONS AND FUTURE WORK

Existing methods do not fully consider how data utility and protection may be traded-off while k-anonymising data. This paper addresses this issue by developing a method which optimises the trade-off between these two requirements in a controlled way. We introduce a distance-based measure that captures both utility and protection and can handle attributes of any type uniformly. We also develop an efficient algorithm based on clustering with partitioning. Our extensive experiments verify that our method produces anonymisations with a good trade-off between data utility and protection, and often outperforms methods which specifically optimise one of these two requirements.

There are some interesting directions for future research. First, it would be worthwhile to study how utility and protection can be traded-off when anonymised data is intended for specific data mining applications. Second, releasing the dominating set of anonymised tables with optimal trade-off is worth exploring. We plan to address these issues in the near future.

7. REFERENCES

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

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

[3] 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.

[4] 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.

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

[6] J. Domingo-Ferrer and J. M. Mateo-Sanz. Practical data-oriented microaggregation for statistical disclosure control. *IEEE TKDE*, 14(1):189–201, 2002.

[7] Y. Du, T. Xia, Y. Tao, D. Zhang, and F. Zhu. On multidimensional k-anonymity with local recoding generalization. In *ICDE '07*, pages 1422–1424, 2007.

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

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

[10] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *25th IEEE ICDCS*, pages 620–629, 2005.

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

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

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

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

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

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

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

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

[19] 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.

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

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

[22] T. M. Truta and B. Vinay. Privacy protection: p-sensitive k-anonymity property. In *ICDEW '06*, page 94, 2006.

[23] L. Willenborg and T. Waal. *Statistical Disclosure Control in Practice*. Springer Berlin Heidelberg, 1996.

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

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

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