TRANSACTIONS ON DATA PRIVACY 6 (2013) 1–17

Achieving *k*-anonymity Using Improved Greedy Heuristics for Very Large Relational Databases

Korra Sathya Babu[1], Nithin Reddy[1], Nitesh Kumar[1], Mark Elliot[2] and Sanjay Kumar Jena[1]

[1]Advanced Data Engineering Laboratory, Department of Computer Science & Engineering,

National Institute of Technology Rourkela, India.

[2]Centre for Census and Survey Research, School of Social Sciences,

University of Manchester, UK.

E-mail: {ksathyababu, rnithinr-cs43, niteshk-cs64, skjena}@nitrkl.ac.in, mark.elliot@manchester.ac.uk

Abstract. Advances in data storage, data collection and inference techniques have enabled the creation of huge databases of personal information. Dissemination of information from such databases - even if formally anonymised, creates a serious threat to individual privacy through statistical disclosure. One of the key methods developed to limit statistical disclosure risk is *k*-anonymity. Several methods have been proposed to enforce *k*-anonymity notably Samarati's algorithm and Sweeney's Datafly, which both adhere to full domain generalisation. Such methods require a trade off between computing time and information loss. This paper describes an improved greedy heuristic for enforcing *k*-anonymity with full domain generalisation. The improved greedy algorithm was compared with the original methods. Metrics like information loss, computing time and level of generalisation were deployed for comparison. Results show that the improved greedy algorithm maintains a better balance between computing time and information loss.

Keywords. Datafly, Full Domain Generalisation, Improved Greedy, *k*-anonymity, Privacy, Samarati's Algorithm.

1 Introduction

In the modern era, the concern for privacy dates back to 1948 and the UN Declaration of human rights article 12 [1]. The exponential increase in the collection of personal information has for some time represented a serious threat to privacy as it was conceived in that declaration [2]. With the advancement of technologies for data storage, data mining, machine learning, social networking and cloud computing, the problem has further increased. As a counterbalance to these socio-technical transformations, most nations have developed both general policies on preserving privacy [1] and specific legislation to control access to, and use of, data (for example the Health Insurance Portability and Accountability Act (HIPAA)

[3] of USA, Personal Health Information Protection Act (PHIPA) [4] in Canada, etc).

With improved data analysis tools, researchers can make use of the information from multiple sources improving both the efficiency and the scope of their research. However, with the concomitant increased information flows, the threats to individual privacy are also increasing. One area of concern is the identification of individuals within anonymised data through the use of linkage attacks.

To reduce the risk of such attacks, the techniques of statistical disclosure control are employed. One such approach called *k*-anonymity [5, 6, 7, 8, 9] works by reducing data across a set of *key variables* to a set of classes. In a *k*-anonymised dataset, each record is indistinguishable from at least *k-1* others, meaning that an attacker cannot link the data records to population units with certainty thus reducing the probability of disclosure. However, preserving privacy through statistical disclosure control also reduces the utility of the data. A good privacy preserving technique should ensure a balance of utility and privacy, giving good performance and level of uncertainty [10]. The system described in this article maintains a balance between computing time and information loss (a standard conceptualisation of the reduction of utility). Computing time is the time taken by a method to convert a given dataset into *k*-anonymised form. Anonymising a dataset results in loss of information; Xiao and Tao [11] proposed a metric called information loss (iloss) to calculate the amount of information loss during generalisation. Three metrics, namely iloss, computing time and level of generalisation are used to compare the proposed algorithm with the existing ones.

In the remainder of this paper, section 2 gives preliminaries; section 3 describes the related work. We review the algorithms that give full domain generalisation with record suppression. We observe that there is a trade off between anonymisation achieved and computing time. Section 4 describes the details of the proposed improved greedy approach. Section 5 describes experiments conducted on two benchmark datasets. Section 6 gives the conclusions.

2 Preliminaries

This section defines the various terms used in the paper. The complexity of the problem is also described.

2.1 Definitions

2.1.1 Quasi-Identifier

A quasi-identifier set (QIS) is a set of attributes that can be linked to external information to re-identify individual records. A QIS when combined with corresponding external data can lead to the correct association of a data record with a population unit (also known as *identification disclosure*). Whether an attribute is a QI or not depends on a variety of factors, the most important is the availability of external data with a variable that corresponds to the potential QI.

2.1.2 *k*-anonymity

A relation, T, is comprised of QIs and non-QIs. The QIs need to be anonymised as they can be used to re-identify individual records. Let t be a tuple, then, the value of i^{th} tuple can be represented as $t_i[C]$.

T satisfies *k*-anonymity if for every tuple $t_{i_0} \in T$, there exist *k-1* other tuples t_{i_1}, t_{i_2}, ... $t_{i_{(k-1)}} \in$ T. such that $t_{i_0}[C] = t_{i_1}[C] = ... = t_{i_{(k-1)}}[C]$, $\forall$ C $\subseteq QIS$

2.1.3 Generalisation

A set of values of a particular attribute is called a *domain*. Given two domains S0 and S1, the expression S0 $\leq$ S1 means that values of attributes in S1 are more generalised than those in S0. Generalisation T_1 is *k-minimal* if it satisfies *k*-anonymity and there does not exist another generalisation satisfying these conditions less general than T_1. There are several schools of thought about how generalisation ontologies should be generated and represented (see [12] for a discussion). Figures 1 and 2 show examples for two value generalisation hierarchies.

Figure 1: Domain generalisation hierarchy for residence.

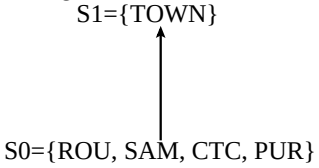
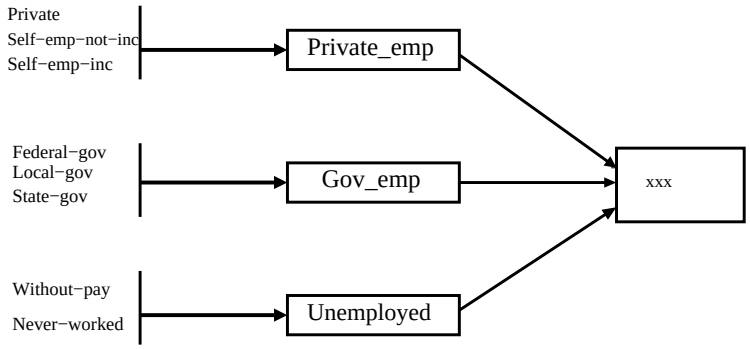


Figure 2: Value generalisation hierarchy for work flow.



2.2 Problem Complexity

A combination of the values of a set of QIs can be viewed as a hierarchical lattice. The *k*-anonymisation algorithms need to search the solution space i.e. the full domain generalisation lattice to find the *k-minimal* solution. In this section, we investigate how the size of the lattice (i.e. the number of nodes in the lattice) depends on the number of quasi-identifiers (r) and the height of generalisation of each QI.

Proposition 1. Let QIS = $\{A_1, ..., A_r\}$ be the set of quasi-identifiers for a private table, with the corresponding domain generalised hierarchy DGH = $\{\text{DGH}_{A_1}, ..., \text{DGH}_{A_r}\}$ and let H = $\{h_1, ..., h_r\}$ be the corresponding height of domain generalised hierarchy such that h_i = height(DGH_{A_i}), for all $1 \leq i \leq r$. Then the size of the full domain generalised hierarchy, $\text{DGH}_{<A_1,...,A_r>}$ is given by:

$$|DGH_{<A_1,...,A_r>}| = \prod_1^r (h_i + 1) \tag{1}$$

Proof. The problem of finding the number of nodes at level n of the lattice of full domain generalisation is same as the problem of finding the number of solutions to the equation:

$$x_1 + x_2 + ... + x_r = n \tag{2}$$

subject to the condition

$$0 \leq x_1 \leq h_1, 0 \leq x_2 \leq h_2, ..., 0 \leq x_r \leq h_r \tag{3}$$

So, the number of nodes at level n is equal to the coefficient of x^n in

$$(x^0 + x^1 + ... + x^{h_1}) \times (x^0 + x^1 + ... + x^{h_2}) \times ... \times (x^0 + x^1 + ... + x^{h_r}) \tag{4}$$

This is because the number of ways in which sum of r integers in (2) subject to the conditions (3) equals n is same as the number of times x^n appears in (4). Also, height($\text{DGH}_{<A_1,...,A_r>}$) = h_1 + h_2 + ... + h_r = h_{max}, since the maximum value attained by n in (1) is h_{max}, according to the condition (3).

Let

$$(x^0+x^1+...+x^{h_1})\times(x^0+x^1+...+x^{h_2})\times...\times(x^0+x^1+...+x^{h_r}) = C_0x^0+C_1x^1+...+C_{h_{max}}x^{h_{max}} \tag{5}$$

where, C_0, C_1, ... , $C_{h_{max}}$ is the number of nodes in the level 0,1, ... ,h_{max} of the lattice respectively. We are interested to find the total number of nodes in the lattice which will be C_0 + C_1 + ... +$C_{h_{max}}$. So, if in equation (5) we let x=1 then:

$$C_0 + C_1 + ... + C_{h_{max}} = (h_1 + 1) \times (h_2 + 1) \times ... \times (h_r + 1) = \prod_1^r (h_i + 1) \tag{6}$$

□

From Proposition 1 it can be shown that the size of full domain generalised hierarchy is exponentially related to the number of QIs of a database.

Proposition 2. Let $f(h, r)$ denote the number of nodes in $\text{DGH}_{<A_1,...,A_r>}$, where QIS = $\{\text{A}_1, ..., \text{A}_r\}$ are quasi-identifiers for a private table T. Then $f(h, r)$ = $O((h + 1)^r)$ where h is the average height of the domain generalised hierarchy of quasi-identifiers in QIS.

Proof. Let $h = \{h_1, ..., h_r\}$ be the corresponding height of domain generalised hierarchies, DGH = DGH_{A_1}, ..., DGH_{A_r} and let h_{avg} be the average height of the domain generalised hierarchies, DGH. That means, h_{avg} = $\frac{h_1+h_2+...+h_r}{r}$. The number of nodes in $\text{DGH}_{<A_1,...,A_r>}$ is given by,

$$f(h, r) = (h_1 + 1) \times (h_2 + 1) \times ... \times (h_r + 1)[\text{From Proposition 1}] \tag{7}$$

and

$$(\sqrt[r]{(h_1+1)\times(h_2+1)\times...\times(h_r+1)}) \leq \frac{(h_1+1)+(h_2+1)+...+(h_r+1))}{r} \tag{8}$$

[From The AM-GM Inequality rule][1]

$$f(h,r) \leq (\frac{(h_1+1)+(h_2+1)+...+(h_r+1)}{r})^r \tag{9}$$

$$\leq (h_{avg}+1)^r \tag{10}$$

$$= O((h_{avg}+1)^r) \tag{11}$$

□

Thus, Proposition 2 shows that the solution space is exponential with respect to r (the number of QIs). Given this, as r becomes large, it becomes impractical to exhaustively search the lattice for a *k-minimal* solution. Thus, the lattice should be searched heuristically in a manner that optimizes the trade-off between computing time and information loss. Samarati's algorithm used binary search to choose levels of the lattice to be searched. In the next few paragraphs, we show that even if binary search is used, Samarati's algorithm is exponential in r. Datafly is polynomial of first order in r, but results in high information loss. Our proposed algorithm is polynomial of second order in r and maintains the best trade-off between computing time and information loss.

If we assume that QIS contains only the attributes having the domain generalised hierarchy of height, h, then, from expression (4), the number of nodes at level n of the full domain generalised hierarchy (FDGH) is the coefficient of x^n in the expansion of $(x^0 + x^1 + ... + x^h)^r$ since, $h_1 = h_2 = ... = h_r = h_{avg} = h$. The expansion of the above expression is given by,

$$(x^0+x^1+...+x^h)^r = \sum_{n=0}^{rh} \left(\begin{array}{c} r \\ n \end{array} \right)_{h+1} x^n \tag{12}$$

$\left(\begin{array}{c} r \\ n \end{array} \right)_{h+1}$ denotes the number of integer compositions of the non-negative integer n with r parts $x_1, ..., x_r$ each from the set $\{0, 1, ..., h\}$ and allows the representation,

$$\left(\begin{array}{c} r \\ n \end{array} \right)_{h+1} = \sum_{\substack{r_0+...+r_h=r \\ 0.r_0+1.r_1+...+h.r_h=n}} \left(\begin{array}{c} r \\ r_0, r_1, ..., r_h \end{array} \right) \tag{13}$$

where $\left(\begin{array}{c} r \\ r_0, r_1, ..., r_h \end{array} \right)$ is a multinomial coefficient, defined as $\frac{r!}{r_0!r_1!...r_h!}$, for non-negative integers $r_0, ..., r_h$. We can verify the representation (13) by noting that for real numbers $y_0, ..., y_h$, it holds that [13]

$$(y_0+y_1+...+y_h)^r = \sum_{\substack{r_0,...,r_h\geq 0 \\ r_0+...+r_h=r}} \left(\begin{array}{c} r \\ r_0, r_1, ..., r_h \end{array} \right) y_0^{r_0}...y_h^{r_h} \tag{14}$$

[1]The AM-GM Inequality rule states that: the arithmetic mean of a list of non-negative real numbers is greater than or equal to the geometric mean of the same list.

Then setting $y_i = x^i$ for all $0 \leq i \leq h$,

$$(x^0 + x^1 + ... + x^h)^r = \sum_{\substack{r_0,...,r_h \geq 0 \\ r_0+...+r_h=r}} \left(\begin{array}{c} r \\ r_0, r_1, ..., r_h \end{array} \right) x^{0.r_0} + ...x^{h.r_h} \tag{15}$$

And comparing coefficients of the right-hand sides of (12) and (15), we get (13). Now we prove that central coefficient $\left(\begin{array}{c} r \\ \lfloor \frac{h_r}{2} \rfloor \end{array} \right)_{h+1}$ is the largest from the remaining coefficient in the expansion of $(x^0 + x^1 + ... + x^h)^r$. This means that the number of nodes in the middle level of the FDGH is highest among all levels.

Lemma 1. Let r ≥ 0 and h ≥ 0 be integers. For all integers n such that $0 \geq n \geq$ hr,

$$\left(\begin{array}{c} r \\ n \end{array} \right)_{h+1} \leq \left(\begin{array}{c} r \\ \lfloor \frac{h_r}{2} \rfloor \end{array} \right)_{h+1} \tag{16}$$

The Proof can be found in [14].

Now we investigate the lower and upper bounds of the central coefficient.

Lemma 2. In the expansion of expression $(x^0 + x^1 + ... + x^h)^r$, the central coefficient $\left(\begin{array}{c} r \\ \lfloor \frac{h_r}{2} \rfloor \end{array} \right)_{h+1}$ satisfies

$$\frac{(h+1)^r}{hr+1} \leq \left(\begin{array}{c} r \\ \lfloor \frac{h_r}{2} \rfloor \end{array} \right)_{h+1} \leq (h+1)^r \tag{17}$$

Proof. The sum of all coefficients in the expansion of the expression is $(h+1)^r$, which can be obtained by setting $h_1 = h_2 = ... = h_r = h$ in Proposition 1. The central coefficient is the largest amongst all coefficients, which is shown in Lemma 1. Since the total number of terms in the expansion of the expression is $(hr+1)$, we get the lower bound of $\frac{(h+1)^r}{hr+1}$ and the upper bound of $(h+1)^r$. Thus, $\frac{(h+1)^r}{hr+1} \leq \left(\begin{array}{c} r \\ \lfloor \frac{h_r}{2} \rfloor \end{array} \right)_{h+1} \leq (h+1)^r$

□

Recently, Eger [15] has found a sharper approximation of the central binomial coefficient by generalising the famous Stirling's approximation to the central binomial coefficient. In the following lemma, we write $a_k \sim b_k$ as a short-hand for $\lim_{k\to\infty} \frac{a_k}{b_k} = 1$.

Lemma 3. For all fixed h,

$$\left(\begin{array}{c} r \\ \lfloor \frac{h_r}{2} \rfloor \end{array} \right)_{h+1} \sim \frac{(h+1)^r}{\sqrt{2\pi r \frac{(h+1)^2-1}{12}}} \tag{18}$$

The Proof can be found in [15].

3 Related Work

Several algorithms have been developed with the purpose of making de-identified data *k*-anonymous [16, 17, 18, 19]. However, this paper is only concerned with the methods

that aim to achieve *k*-anonymity through full domain global recoding, hierarchical generalisation and minimal suppression. Two of the most popular approaches that meet this specification are Sweeney's Datafly algorithm [20] and Samarati's algorithm [21].

Samarati proposed the concept of the domain generalisation hierarchy. The algorithm is based on the axiom that if a node at level h, in domain generalisation hierarchy satisfies *k*-anonymity, then all the levels of height greater than h also satisfy *k*-anonymity. To exploit this, the algorithm uses a binary search over the levels of the domain generalisation hierarchy, i.e. for a DGH of height h, the search starts at a level of $h/2$ and if that level satisfies the *k*-anonymity then it searches at the level $h/4$. Otherwise, it searches at the level $3h/4$ and it goes on the same way till it finds the solution in the lowest possible level. This approach assumes that the optimal level is the one with the least generalisation and, within that level, it takes the node that has the minimum information loss as the solution.

Datafly uses a greedy algorithm to search the domain generalisation hierarchy; at every step, it chooses the locally optimal move. One drawback with Datafly's approach in common with all such hill climbing type algorithms is that it may become trapped in a local optimum [25].

Other methods, such as Incognito [22], aim to find all possible solutions. However, a major drawback of such approaches is that the number of solutions they return is usually very high, and checking the information loss of all of them in order to find the optimal one is impractical. Incognito implements a dynamic programming approach with the idea that if a subset of quasi-identifiers of a relation T is not *k*-anonymous then T cannot be *k*-anonymous. The approach constructs a generalisation lattice of each individual subset of QIS and traverses them by performing a bottom-up, breadth-first search. The number of generalisation lattices constructed for a QIS of order r is 2^r i.e. the power set of QIS. Thus the Incognito algorithm is at least of order 2^r i.e. $\Omega(2^r)$ because at least one check is made for *k*-anonymity for each generalisation lattice.

Emam et al. [23] proposed an algorithm called Optimal Lattice Anonymization (OLA) and showed that it outperforms Incognito. It uses predictive tagging to prune the search space of the lattice. However, if the globally optimal *k*-anonymous lattice lies on or above the middle level of full domain generalized hierarchy, then the algorithm checks all the middle level lattices for *k*-anonymity. We have shown above in Proposition 2 that checking only the middle level of full domain generalized hierarchy is exponential in number of quasi-identifiers.

Kohlmayer et al. [24] proposed the Flash algorithm for finding efficient, stable and optimal *k*-anonymity. They implemented their algorithm in a framework, which holds all the data in a main memory (using dictionary compression). The maximum number of quasi-identifiers for the datasets considered by them is 9. If the number of quasi-identifiers were very high, then it would be difficult to hold all the data items in the main memory.

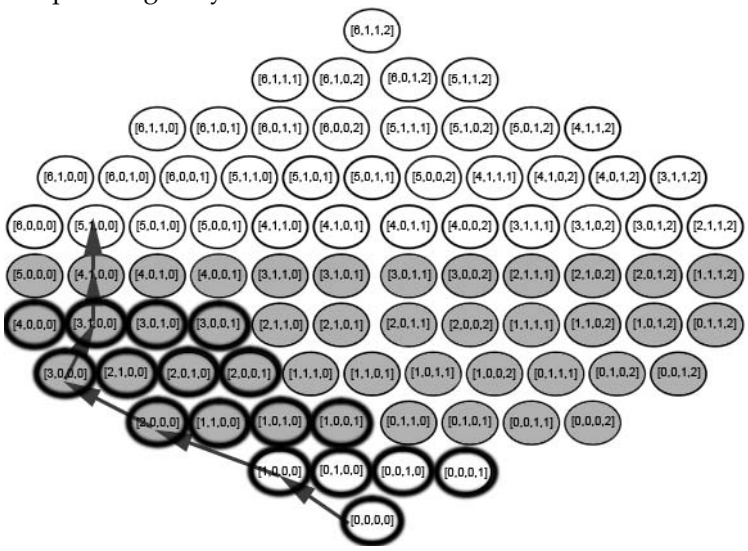
4 The Improved Greedy Heuristic

Sweeney's Datafly starts checking from the lowest level of the DGH and checks for the *k*-anonymity. If it is not satisfied then it generalises the most differentiated attribute, stepping into the next level of hierarchy and repeating until it is satisfied that *k*-anonymity has been achieved. It checks only one attribute in each level, which guarantees that a solution will be found in polynomial time; but not necessarily the optimal one.

Analyzing the algorithms, we observe that both Samarati's and the Incognito algorithms give theoretically optimal solutions but not in polynomial time, whereas the Datafly algo-

rithm finds a solution in a polynomial time but is not necessarily optimal.

Figure 3: A visualization of the three processing strategies: the circles with fillings represent Samarati's method; the arrow represents the strategy for Datafly and the dark circles represent the improved greedy method.



The greedy heuristic that Datafly uses simply generalises the most differentiated attribute. This paper proposes an improved greedy heuristic which gives better solutions than the heuristics of the Datafly algorithm. The algorithm starts by checking for *k*-anonymity from the lowest level. If it is not satisfied, the *k*-anonymity is checked by generalising one attribute at a time simultaneously implementing the minimum suppression. If, by generalising an attribute, the *k*-anonymity is satisfied, the attribute is generalised and declared as the solution to the first phase. If after generalising all the attributes, none of them satisfies *k*-anonymity, the case that is the closest to *k*-anonymity is selected to be generalised. In case of ties the most differentiated attribute is selected. This process continues until a solution is found. In Figure 3, the circles with fillings represent Samarati's method, the arrow represents the strategy for Datafly and the dark circles represent the improved greedy method.

Theorem 1. *The number of nodes in the full domain generalised hierarchy that need to be checked for k-anonymity in Samarati's algorithm is exponential in the number of quasi-identifiers.*

Proof. Samarati's algorithm uses binary search on the levels of the lattice with the nodes in the middle level being searched first. For simplicity we assume $h_1 = h_2 = ... = h_r = h$, then the number of nodes in the middle level of the lattice is given by the central coefficient $\left(\begin{array}{c} r \\ \lfloor \frac{h_r}{2} \rfloor \end{array} \right)_{h+1}$, where $QIS = \{A_1, ..., A_r\}$ is the set of quasi-identifiers and $H = \{h_1, ..., h_r\}$ is the corresponding height of domain generalised hierarchy. In Lemma 2 and Lemma 3 we have shown that the central coefficient is exponential in r. Thus, searching only middle level of the lattice has lower bound $\frac{(h+1)^r}{(hr+1)}$ and the upper bound $(h+1)^r$.

□

Therefore, as the value r becomes large, applying binary search on the levels of the lattice becomes increasingly less effective in reducing the number of potential nodes that need to be checked for *k*-anonymity.

4.1 Description of the Algorithm

Initially, a full domain generalisation hierarchy (FDGH) is constructed (using Samarati's algorithm). The algorithm then starts by checking for *k*-anonymity in the lowest level of the FDGH which has only one node i.e. the node which has all the quasi-identifiers without any generalisation. If *k*-anonymity is satisfied, then the dataset is *k*-anonymous as it is and needs no further anonymisation. But if *k*-anonymity is not satisfied, then *k*-anonymity is checked by generalising one attribute at a time. If none of the nodes at a given level of the FDGHs satisfies *k*-anonymity, then the node that is closest to *k*-anonymity is selected (in case of tie, the node which has generalized the most differentiated attributes is selected) and *k*-anonymity is again checked by generalizing one attribute at a time in the selected node. The previous step is repeated until a node is found which is *k*-anonymous. Theorem 2 shows that the complexity of the proposed algorithm is polynomial.

Theorem 2. *The number of nodes in the full domain generalised hierarchy that need to be checked for k-anonymity in the proposed greedy algorithm is polynomial of second order in the number of quasi-identifiers and Datafly algorithm is polynomial of first order in the number of quasi-identifiers.*

Proof. Let $QIS = \{A_1, ..., A_r\}$ be the set quasi-identifiers and $h = \{h_1, ..., h_r\}$ be the corresponding height of domain generalised hierarchy. The proposed algorithm starts checking nodes from the lowest level, 0 of the full domain generalised hierarchy, $\text{DGH}_{<A_1,...,A_r>}$. In the worst case (at level h_i), r possible generalisations are checked for *k*-anonymity from which the generalisation closest to *k*-anonymity is selected. This worst case value is obtained by generalising the attributes from QIS one at a time from the selected node closest to *k*-anonymity at level h_{i-1}. Thus in the worst case, the algorithm goes to the highest level, h_{max} of the hierarchy, $\text{DGH}_{<A_1,...,A_r>}$ and the number of nodes searched in our proposed algorithm is given by $O(r \times h_{max})$. In Proposition 1, we indicated that the highest level, h_{max} is given by $h_{max} = h_1 + h_2 + ... + h_r$. In Proposition 2, we indicated that $h_{avg} = \frac{h_1+h_2+...+h_r}{r}$. The number of nodes searched is given by $O(r^2 \times h_{avg})$. If we assume $h_1 = h_2 = ... = h_r = h$, it becomes $O(r^2 \times h)$. Thus the proposed algorithm is polynomial of second order in the number of quasi-identifiers.

The Datafly algorithm checks only one generalisation at level h_i of the $\text{DGH}_{<A_1,...,A_r>}$. In the worst case, the algorithm goes to the highest level, h_{max}. So, the number of nodes searched by Datafly algorithm is given by, $O(h_{max}) = O(r \times h_{avg})$. If we assume $h_1 = h_2 = ... = h_r = h$, it becomes $O(r \times h)$ which means the algorithm is polynomial of first order in the number of quasi-identifiers. □

The details of the proposed algorithm are given in Algorithm 1.

5 Empirical Results

The three algorithms were tested on two datasets obtained from UCI Machine Learning Repository: the Adult dataset [26] and the CUPS dataset [27]. The Adult dataset has 32,561 records and 15 attributes of which 8 attributes (namely Age (3), Profession (2), Education (2), Marital status (2), Position (2), Race (1), Sex (1), Country (3)) were considered to be

Algorithm 1: Improved Greedy

Notations: n: number of quasi-identifiers
k: desired anonymity
m: suppression limit
$node$: array of n integers
$array\_node$: array of nodes
$array\_anonymity$: array of integers
$array\_index$: array of integers
$dtree$: two dimensional array `// each row corresponding to a level of DGH and column to the node in the particular level`

Input : T: A table to be *k*-anonymised
Q: a set of n quasi-identifier attributes
a set of dimension tables (one for each quasi-identifier in Q) `// Contains source and destination nodes`

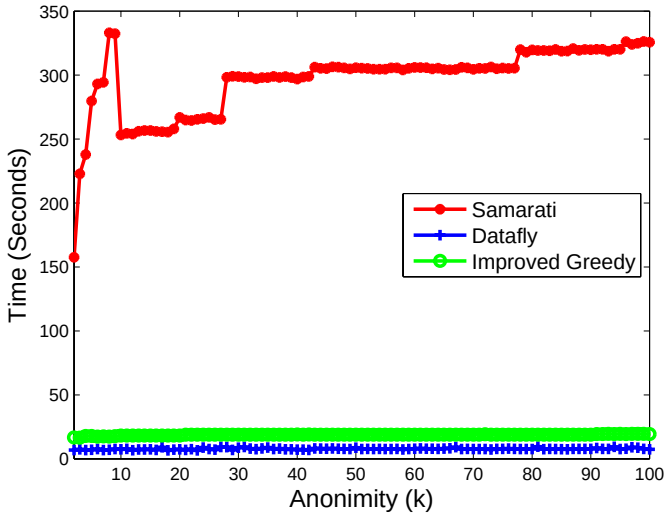
Output : The array of integers which represents the levels of generalisations of the quasi-identifiers of T required to achieve *k*-anonymity

```
Construct the domain generalised hierarchy using Samarati's algorithm for the given tables
node ← array of n zeros
temp ← anonymity(node) // anonymity() is a function which takes the node in the DGH as input and outputs the anonymity of the particular node using suppression limit m
if temp ≥ k then
    return node // T is anonymous as it is given
end
while temp < k do
    array_node ← array of nodes obtained by generalising each quasi-identifier at a time
    array_anonymity ← empty array
    for each level in array_levels append anonymity(node) to array_anonymity
    array_index ← max(array_anonymity) // max() is a function which takes an integer array as input and outputs the array of indices of all the integers with the largest value in the input array
    if len(array_index) = 1 then
        // len() is a function which takes an array as an input and outputs the number of elements in the input array
        node ← array_node[array_index[0]]
        temp ← array_anonymity[array_index[0]]
    else
        index ← diverse(array_index) // diverse() is a function which takes and array of integers as an input, takes into consideration the nodes in array_node with indices in array_index and outputs the index of the node which generalises the most diverse attribute (i.e. the attribute with the most distinct values as used in Datafly algorithm)
        node ← array_node[index] temp ← array_anonymity[index]
    end
end
return node
```

QI[2]. The CUPS dataset has 96367 records and 479 attributes among which, 5 attributes (namely Age (4), Gender (3), Income (4), Cluster (4) and Domain (3)) were considered to be QIs. Whilst we endeavoured for this exercise to maximize the number of attributes that we used as quasi-identifiers, the data quality of the remaining attributes was poor and, we deemed, insufficiently good for our purposes. However, the results presented here are not dependent on the particular attributes chosen; very similar patterns of results are obtained regardless of the cross classification of QIs employed. Since the hierarchies used in our analysis are not supplied in the machine learning repository, the authors considered various hierarchies and chose the ones which were most appropriate based on common sense domain knowledge.

Comparing the time complexities of the algorithms, Samarati's algorithm has an exponential time complexity (shown in Theorem 1) and both the greedy approaches discussed have polynomial time complexities (shown in Theorem 2). From the time complexities, it is obvious that Datafly will have the minimum time of execution followed by proposed improved greedy and Samarati. Figures 4 and 5 compare the execution times of the Datafly, Samarati and improved greedy algorithms for the two datasets. As expected, the execution time of Samarati's algorithm is the highest[3]. The execution times of both greedy approaches are found to be almost the same with Datafly being the faster of the two.

Figure 4: Anonymity vs time for the Adult dataset.

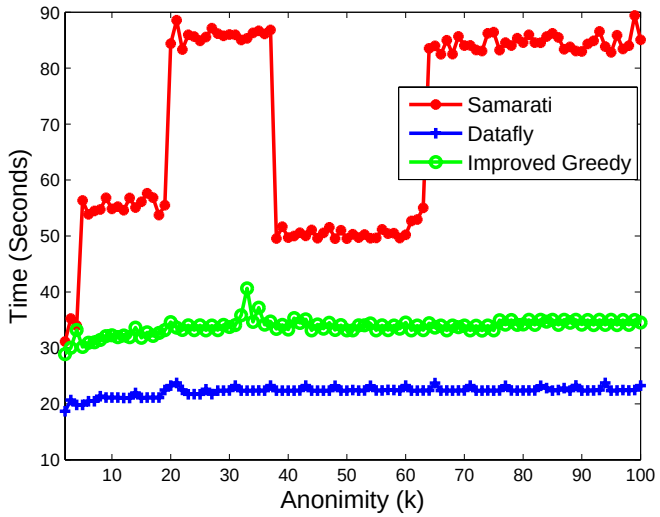


When applying disclosure control to any data product, we need to be mindful of the impact on the utility of that product. If privacy were our only concern, then utility would tend to become zero. In the literature, damage to utility is most frequently operationalised as information loss of which there are many different definitions. According to Domingo-Ferrer and Torra [28], information loss can be obtained by comparing the original data to the masked one, the closer the data is to the original, the lower is the information loss. A

[2]The numbers in the brackets indicate the maximum height in the hierarchy.

[3]The execution time of Samarati's approach varies non-monotonically. This is because the execution time of Samarati's algorithm also depends on the order in which the QIS is arranged in the DGH.

Figure 5: Anonymity vs time for the CUPS dataset.



similar definition is given by Xu et al. in [29]: the information loss measures "how well the generalised tuples approximate the original ones".

A general-purpose metric that is considered here is *iloss*, proposed by Xiao and Tao [11]. This metric is used to find the information loss of transforming a value to a more general one. It assumes that all raw values are at the leaves of the taxonomic tree. The detail of this metric is given in [30]. The formula is given below:

$$iloss(V_g) = \frac{|V_g| - 1}{|D_A|} \tag{19}$$

$$iloss(r) = \sum_{V_g \in r} (w_i \times iloss(V_g)) \tag{20}$$

$$iloss(T) = \sum_{r \in T} iloss(r) \tag{21}$$

Where:
$|V_g|$ is the number of domain values that are descendants of V_g
$|DA|$ is the number of domain values in the attribute A of V_g
w_i is a positive constant specifying the penalty weight of attribute A_i of V_g

The information loss of the *k*-anonymised datasets for the three algorithms is plotted in Figures 6 and 7. We would expect Samarati's algorithm to have the lowest information loss as it searches for the optimal solution in the hierarchy using brute force. So, the ideal would be to achieve the same information loss as Samarati's algorithm but with lower time complexity (i.e. in a polynomial time). The Datafly algorithm, having least time complexity, suffers from very high information loss and the loss is unacceptable if the value of *k* is large. The improved greedy algorithm has time complexity comparable to Datafly and information loss comparable to Samarati's algorithm.

Figure 6: Anonymity vs iloss for the Adult dataset.

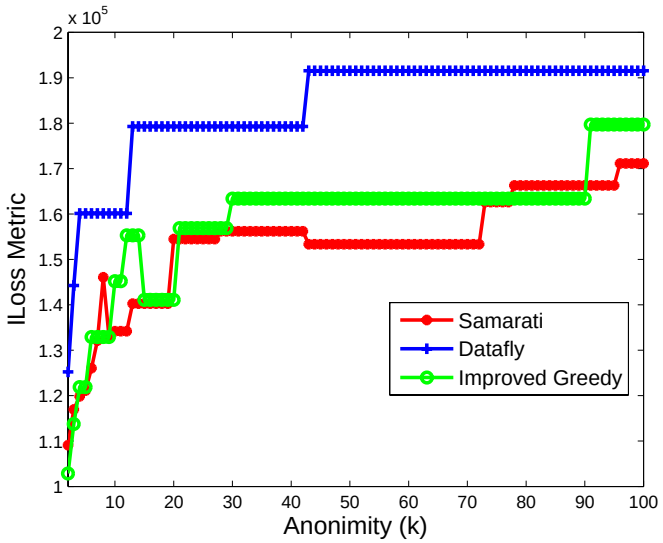
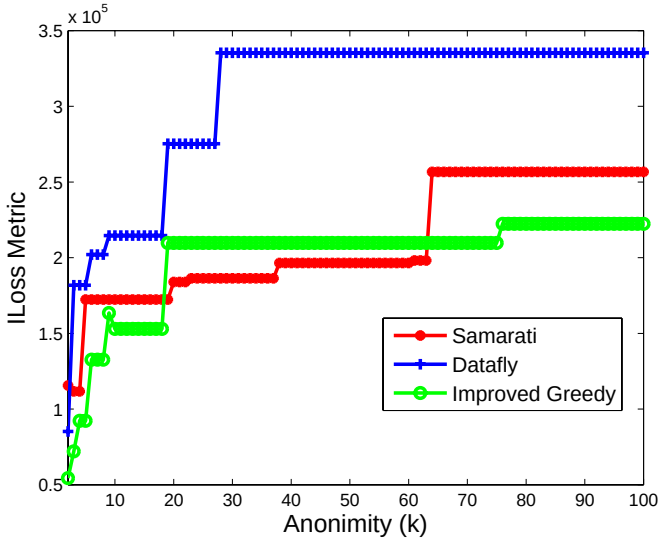


Figure 7: Anonymity vs iloss for the CUPS dataset.



Figures 8 and 9 show the level of generalisation at which a given level of k-anonymity was achieved for the three algorithms. The level of generalisation for the improved greedy algorithm is similar to the Samarati's algorithm for the Adult dataset although slightly inferior for the CUPS dataset. In both cases the improved greedy algorithm performs markedly better than Datafly. The improved greedy algorithm has similar information loss to Samarati's algorithm so it may seem surprising that it performs slightly worse in terms of the level of generalisation. However, as Samarati's algorithm uses a brute force method to find the solution in the lowest possible level, it optimises only with respect to the level of generalisation metric but the information loss within a level of a full domain generalisation hierarchy is not the same for all the nodes. The improved greedy heuristic gives more emphasis on choosing the nodes *within a level*. So, the nodes chosen *at a given level* by our algorithm have, on average, less information loss than those chosen by Samarati's algorithm but when we consider only the level of generalisation Samarati's algorithm performs better. Overall, the information loss is comparable.

Figure 8: Anonymity vs level of generalisation for Adult dataset.

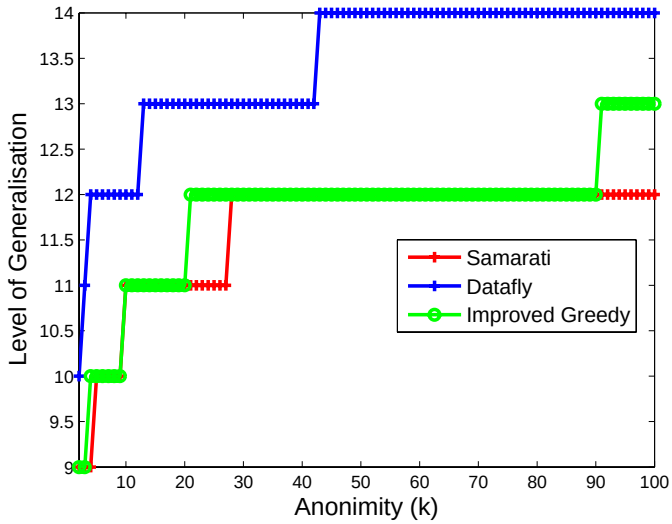


Figure 10 shows the execution times of various algorithms with varying the number of QIs for $k = 10$. This analysis was done for the Adult dataset for 3 to 8 QIs. We can infer that the execution time of Samarati's approach was highest as expected. In addition, as expected, its execution time is found to increase exponentially as the number QIs increases.

6 Conclusion

Before the release of any data product, there is a need to consider a balance between utility and privacy. In this paper, we have proposed an improved greedy heuristic for k-anonymisation that maintains a better balance between privacy and information loss than other similar algorithms. Metrics like iloss, computing time and level of generalisation were used to compare the new algorithm with the established Datafly and Samarati's al-

Figure 9: Anonymity vs level of generalisation for the CUPS dataset.

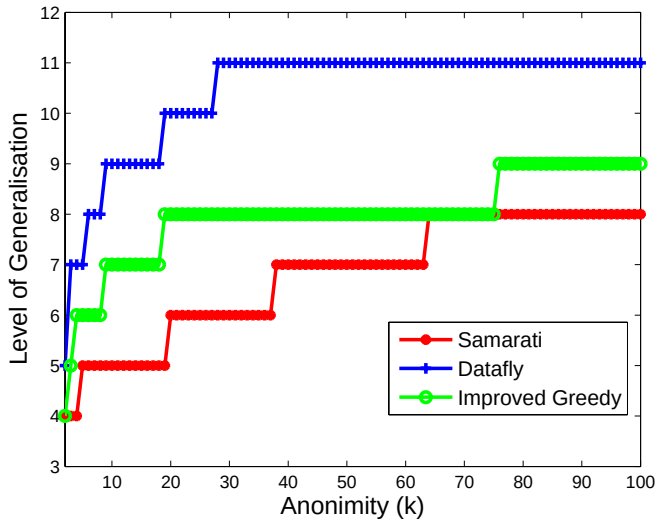
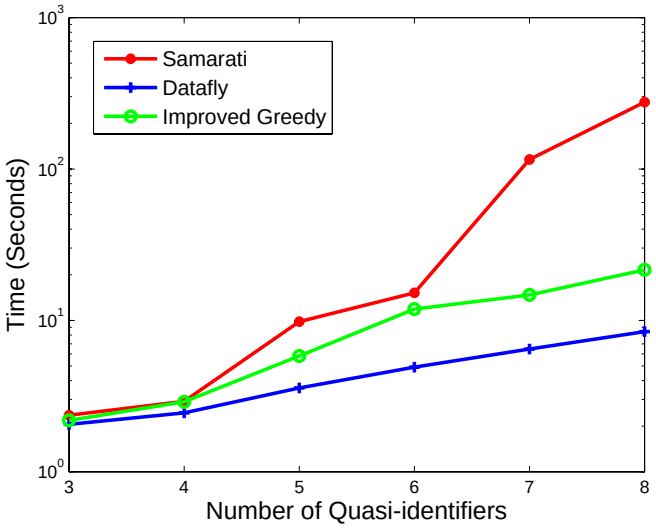


Figure 10: Quasi-identifier vs time for the Adult dataset for k=10.



gorithms. On balance, the new greedy algorithm performed better than either of the established ones.

References

[1] www.un.org/en/documents/udhr/

[2] Purdam K., Elliot M. J. and Mackey E., The Regulation of the Personal: Individual Data use and Identity in the UK, Journal of policy studies, 25(4), 2004, pp.267-282.

[3] Health Insurance Portability and Accountability Act, Available online at http://www.hhs.gov/ocr/hipaa.

[4] Personal Health Information Protection Act, available online at http://www.e-laws.gov.on.ca/html/statutes/english/elaws_statutes_04p03_e.htm.

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

[6] Sweeney L., Achieving *k*-anonymity privacy protection using generalization and suppression, International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10(5), 2002, pp.571-588.

[7] Samarati P. and Sweeney L., Generalizing data to provide anonymity when disclosing information, In Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Seattle, Washington, United States, June 01-04, 1998, pp.188.

[8] Aggarwal G., Feder T., Kenthapadi K., Motwani R., Panigrahy R., Thomas D. and Zhu A., Anonymizing tables, In Proceedings of 10th International Conference on Database Theory, Edinburgh, UK, January 5-7, 2005, pp.246-258.

[9] Bayardo B. and Agrawal R., Data privacy through optimal *k*-anonymity, In Proceedings of the 21st International Conference on Data Engineering, Tokyo, Japan, April 5-8, 2005, pp.217-228.

[10] Verykios V. S., Bertino E., Fovino I. N., Provenza L. P., Saygin Y. and Theodoridis Y., State-of-the-art in privacy preserving data mining, SIGMOD Record, 33(1), 2004, pp.50-57.

[11] Xiao X. and Tao Y., Personalized Privacy Preservation, Proceedings of the 2006 ACM SIGMOD International conference on Management of data, Chicago, IL, USA, June 27-29, 2006, pp.229-240.

[12] Samarati P. and Sweeney L., Protecting privacy when disclosing information: *k*-anonymity and its enforcement through generalization and suppression, Technical Report SRI-CSL-98-04, SRI Computer Science Laboratory, 1998.

[13] Weisstein E. W., Multinomial series. From MathWorld-A Wolfram Web Resource.

[14] Eger S., Asymptotic normality of integer compositions inside a rectangle, CoRR, Arxiv preprint arXiv, abs/1203.0690, 2012, pp.1-7.

[15] Eger S., Stirling's approximation for central polynomial coefficients, CoRR, Arxiv preprint arXiv, abs/1203.2122, 2012, pp.1-5.

[16] Ciriani V., De Capitani di Vimercati S., Foresti S. and Samarati P., Microdata Protection, Secure Data Management in Decentralized Systems, Advances in Information Security, Springer, 2007, pp.291-321.

[17] Truta T. M., Campan A., Abrinica M. and Miller J., A Comparison between Local and Global Recoding Algorithms for Achieving Microdata P-Sensitive *k*-anonymity, Acta Universitatis Apulensis, Alba Iulia, Romania, No.15, 2008, pp.213-233.

[18] Fung B. C. M., Wang K., Chen R. and Yu P. S., Privacy-Preserving Data Publishing: A Survey of Recent Developments, ACM Computing Surveys, 42(4), 2010, pp.1-53.

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

[20] Sweeney L., Guaranteeing Anonymity When Sharing Medical Data, The Datafly System, Proceedings, Journal of the American Medical Informatics Association, Washington, DC, Hanley & Belfus, Inc., 1997, pp.1-5.

[21] Samarati P., Protecting Respondents' Identity in Microdata Release, IEEE Transactions on Knowledge and Data Engineering, 13(6), 2001, pp.1010-1027.

[22] LeFevre K., DeWitt D.J. and Ramakrishnan R., Incognito: Efficient Fulldomain *k*-anonymity, Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, Maryland, USA, June 14-16, 2005, pp.49-60.

[23] Emam K. E., Dankar F. K., Issa R., Jonker E., Amyot D., Cogo E., Corriveau J. P., Walker M., Chowdhury S., Vaillancourt R., Roffey T. and Bottomley J., A globally optimal *k*-anonymity method for de-identification of health data, Journal of American Medical Informatics Association, 16(5), 2009, pp.670-682.

[24] Kohlmayer F., Prasser F., Eckert C., Kemper A. and Klaus A. K., Flash: Efficient, Stable and Optimal *k*-anonymity, Proceedings of ASE/IEEE International Conference on Privacy, Security, Risk and Trust, Amsterdam, The Netherlands, September 3-5, 2012, pp.708-717.

[25] Cormen T., Leiserson C. and Rivest R., Introduction to Algorithms, Chapter 16 (Greedy Algorithms), MIT Press, 1990, pp. 370-399.

[26] ftp://ftp.ics.uci.edu/pub/machine-learning-databases/adult.

[27] http://kdd.ics.uci.edu/databases/kddcup98/kddcup98.html.

[28] Domingo-Ferrer J. and Torra V., Risk Assessment in Statistical Microdata Protection via Advanced Record Linkage, Statistics and Computing, 13(4), 2003, pp.343-354.

[29] Xu J., Wang W., Pei J., Wang X., Shi B. and Fu A.W., Utility-Based Anonymization Using Local Recoding, Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006, pp.785-790.

[30] Fung B. C. M., Wang K., Fu A. W. and Yu P. S., Introduction to Privacy Preserving Data Publishing: Concepts and Techniques", Chapman & Hall/CRC, 2010.