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

ACHIEVING *k*-ANONYMITY PRIVACY PROTECTION USING GENERALIZATION AND SUPPRESSION[1]

LATANYA SWEENEY

School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
E-mail: latanya@cs.cmu.edu

Received May 2002

Often a data holder, such as a hospital or bank, needs to share person-specific records in such a way that the identities of the individuals who are the subjects of the data cannot be determined. One way to achieve this is to have the released records adhere to *k*-anonymity, which means each released record has at least (*k*-1) other records in the release whose values are indistinct over those fields that appear in external data. So, *k*-anonymity provides privacy protection by guaranteeing that each released record will relate to at least *k* individuals even if the records are directly linked to external information. This paper provides a formal presentation of combining generalization and suppression to achieve *k*-anonymity. Generalization involves replacing (or recoding) a value with a less specific but semantically consistent value. Suppression involves not releasing a value at all. The Preferred Minimal Generalization Algorithm (MinGen), which is a theoretical algorithm presented herein, combines these techniques to provide *k*-anonymity protection with minimal distortion. The real-world algorithms Datafly and µ-Argus are compared to MinGen. Both Datafly and µ-Argus use heuristics to make approximations, and so, they do not always yield optimal results. It is shown that Datafly can over distort data and µ-Argus can additionally fail to provide adequate protection.

Keywords: data anonymity, data privacy, re-identification, data fusion, privacy.

1. Introduction

Today's globally networked society places great demand on the collection and sharing of person-specific data for many new uses [1]. This happens at a time when more and more historically public information is also electronically available. When these data are linked together, they provide an electronic image of a person that is as identifying and personal as a fingerprint even when the information contains no explicit identifiers, such as name and phone number. Other distinctive data, such as birth date and postal code, often combine uniquely [2] and can be linked to publicly available information to re-identify individuals.

[1] This paper significantly amends and expands the earlier paper "Protecting privacy when disclosing information: *k*-anonymity and its enforcement through generalization and suppression" (with Samarati) submitted to IEEE Security and Privacy 1998, and extends parts of my Ph.D. thesis [10].

So in today's technically-empowered data rich environment, how does a data holder, such as a medical institution, public health agency, or financial organization, share person-specific records in such a way that the released information remain practically useful but the identity of the individuals who are the subjects of the data cannot be determined? One way to achieve this is to have the released information adhere to *k*-anonymity [3]. A release of data is said to adhere to *k*-anonymity if each released record has at least (*k*-1) other records also visible in the release whose values are indistinct over a special set of fields called the quasi-identifier [4]. The quasi-identifier contains those fields that are likely to appear in other known data sets. Therefore, *k*-anonymity provides privacy protection by guaranteeing that each record relates to at least *k* individuals even if the released records are directly linked (or matched) to external information.

This paper provides a formal presentation of achieving *k*-anonymity using generalization and suppression. G*eneralization* involves replacing (or recoding) a value with a less specific but semantically consistent value. *Suppression* involves not releasing a value at all. While there are numerous techniques available[2], combining these two offers several advantages.

First, a recipient of the data can be told what was done to the data. This allows results drawn from released data to be properly interpreted. Second, information reported on each person is "truthful" which makes resulting data useful for fraud detection, counter-terrorism surveillance, healthcare outcome assessments and other uses involving traceable person-specific patterns[3]. Third, these techniques can provide results with guarantees of anonymity that are minimally distorted. Any attempt to provide anonymity protection, no matter how minor, involves modifying the data and thereby distorting its contents, so the goal is to distort minimally. Fourth, these techniques can be used with preferences a recipient of the released data may have, thereby providing the most useful data possible. In this way, algorithmic decisions about how to distort the data can have minimal impact on the data's fitness for a particular task.

Finally, the real-world systems Datafly [5] and μ-Argus [6], which are discussed in subsequent sections, use these techniques to achieve *k*-anonymity. Therefore, this work provides a formal basis for comparing them.

2. Background

The ideas of *k*-anonymity and of a quasi-identifier are straightforward. Nevertheless, care must be taken to precisely state what is meant. [3] provides a detailed discussion of *k*-anonymity. A brief summary is provided in this section as background for the upcoming presentations on generalization and suppression.

[2] See Willenborg and De Waal [2] for a list of traditional statistical techniques.

[3] In contrast, other techniques (e.g., additive noise) can destroy the "truthfulness" of the information reported on a person even though they can maintain aggregate statistical properties.

Unless otherwise stated, the term *data* refers to person-specific information that is conceptually organized as a table of rows (or records) and columns (or fields). Each row is termed a *tuple*. Tuples within a table are not necessarily unique. Each column is called an *attribute* and denotes a semantic category of information that is a set of possible values; therefore, an attribute is also a domain. Attributes within a table are unique. So by observing a table, each row is an ordered *n*-tuple of values $<d_1, d_2, \ldots, d_n>$ such that each value d_j is in the domain of the *j*-th column, for $j=1, 2, \ldots, n$ where n is the number of columns. This corresponds to relational database concepts [7].

Let $\mathsf{B}(A_1,\ldots,A_n)$ be a *table* with a finite number of tuples. The finite set of *attributes* of B are $\{A_1,\ldots,A_n\}$. Given a table $\mathsf{B}(A_1,\ldots,A_n)$, $\{A_i,\ldots,A_j\} \subseteq \{A_1,\ldots,A_n\}$, and a tuple $t \in \mathsf{B}$, I use $t[A_i,\ldots,A_j]$ to denote the sequence of the values, $v_i,\ldots,v_j$, of $A_i,\ldots,A_j$ in t. I use $\mathsf{B}[A_i,\ldots,A_j]$ to denote the projection, maintaining duplicate tuples, of attributes $A_i,\ldots A_j$ in B.

Throughout this work each tuple is assumed to be specific to a person and no two tuples pertain to the same person. This assumption simplifies discussion without loss of applicability. Also, this discussion focuses on protecting identity in person-specific data, but is just as applicable to protecting other kinds of information about other kinds of entities (e.g., companies or governments).

Limiting the ability to link (or match) released data to other external information offers privacy protection. Attributes in the private information that could be used for linking with external information are termed the quasi-identifier. Such attributes not only include explicit identifiers such as name, address, and phone number, but also include attributes that in combination can uniquely identify individuals such as birth date, ZIP[4], and gender [8]. A goal of this work is to release person-specific data such that the ability to link to other information using the quasi-identifier is limited.

Definition 1. Quasi-identifier

Given a population U, a person-specific table $\mathsf{T}(A_1,\ldots,A_n)$, $f_c: \mathsf{U} \rightarrow \mathsf{T}$ and $f_g: \mathsf{T} \rightarrow \mathsf{U}'$, where $\mathsf{U} \subseteq \mathsf{U}'$. A quasi-identifier of T, written Q_T, is a set of attributes $\{A_i,\ldots,A_j\} \subseteq \{A_1,\ldots,A_n\}$ where: $\exists p_i \in \mathsf{U}$ such that $f_g(f_c(p_i)[Q_T]) = p_i$.

Definition 2. *k*-anonymity

Let $\mathsf{RT}(A_1,\ldots,A_n)$ be a table and QI_{RT} be the quasi-identifier associated with it. RT is said to satisfy *k*-anonymity if and only if each sequence of values in $\mathsf{RT}[QI_{RT}]$ appears with at least k occurrences in $\mathsf{RT}[QI_{RT}]$.

[4] In the United States, a ZIP code refers to the postal code. Typically 5-digit ZIP codes are used, though 9-digit ZIP codes have been assigned. A 5-digit code is the first 5 digits of the 9-digit code.

Example 1. Table adhering to *k*-anonymity

Figure 1 contains table T, which adheres to *k*-anonymity. The quasi-identifier is QI_{T}= {*Race*, *Birth*, *Gender*, *ZIP*} and *k*=2. Therefore, for each of the tuples contained in T, the values that comprise the quasi-identifier appear at least twice in T. In particular, $t1[\mathsf{QI}_{\mathsf{T}}] = t2[\mathsf{QI}_{\mathsf{T}}]$, $t3[\mathsf{QI}_{\mathsf{T}}] = t4[\mathsf{QI}_{\mathsf{T}}]$, and $t5[\mathsf{QI}_{\mathsf{T}}] = t6[\mathsf{QI}_{\mathsf{T}}] = t7[\mathsf{QI}_{\mathsf{T}}]$.

	Race	Birth	Gender	ZIP	Problem
t1	Black	1965	m	02141	short breath
t2	Black	1965	m	02141	chest pain
t3	Black	1964	f	02138	obesity
t4	Black	1964	f	02138	chest pain
t5	White	1964	m	02138	chest pain
t6	White	1964	m	02138	obesity
t7	White	1964	m	02138	short breath

Figure 1 Example of *k*-anonymity, where *k*=2 and QI={*Race*, *Birth*, *Gender*, *ZIP*}

Theorem 1

Let $\mathsf{RT}(A_1,...,A_n)$ be a table, $QI_{RT}=(A_i,\ldots, A_j)$ be the quasi-identifier associated with RT, $A_i,\ldots,A_j \subseteq A_1,\ldots,A_n$, and RT satisfy *k*-anonymity. Then, each sequence of values in $\mathsf{RT}[A_x]$ appears with at least *k* occurrences in $\mathsf{RT}[QI_{RT}]$ for $x=i,\ldots,j$.

Example 2. *k* occurrences of each value under *k*-anonymity

Table T in Figure 1 adheres to *k*-anonymity. Therefore, each value associated with an attribute of QI in T appears at least *k* times. $|\mathsf{T}[Race = \text{"black"}]| = 4$. $|\mathsf{T}[Race = \text{"white"}]| = 3$. $|\mathsf{T}[Birth = \text{"1964"}]| = 5$. $|\mathsf{T}[Birth = \text{"1965"}]| = 2$. $|\mathsf{T}[Gender = \text{"m"}]| = 5$. $|\mathsf{T}[Gender = \text{"f"}]| = 2$. $|\mathsf{T}[ZIP = \text{"02138"}]| = 5$. And, $|\mathsf{T}[ZIP = \text{"02141"}]| = 2$.

It can be trivially proven that if the released data RT satisfies *k*-anonymity with respect to the quasi-identifier $\mathsf{QI}_{\mathsf{PT}}$, then the combination of the released data RT and the external sources on which $\mathsf{QI}_{\mathsf{PT}}$ was based, cannot link on $\mathsf{QI}_{\mathsf{PT}}$ or a subset of its attributes to match fewer than *k* individuals.

3. Methods

In this section I present formal notions of: (1) generalization incorporating suppression; (2) minimal generalization; and, (3) minimal distortion. The Preferred Minimal Generalization Algorithm (MinGen), which ends this section, combines these notions into a theoretical algorithm that uses generalization and suppression to produce tables that adhere to *k*-anonymity with minimal distortion.

3.1. Generalization including suppression

The idea of generalizing an attribute is a simple concept. A value is replaced by a less specific, more general value that is faithful to the original. In Figure 2 the original ZIP codes {*02138*, *02139*} can be generalized to *0213**, thereby stripping the rightmost digit and semantically indicating a larger geographical area.

In a classical relational database system, domains are used to describe the set of values that attributes assume. For example, there might be a ZIP domain, a *number* domain and a *string* domain. I extend this notion of a domain to make it easier to describe how to generalize the values of an attribute. In the original database, where every value is as specific as possible, every attribute is considered to be in a ground domain. For example, 02139 is in the ground ZIP domain, Z_0. In order to achieve *k*-anonymity I can make ZIP codes less informative. I do this by saying that there is a more general, less specific domain that can be used to describe ZIPs, say Z_1, in which the last digit has been replaced by 0 (or removed altogether). There is also a mapping from Z_0 to Z_1, such as $02139 \rightarrow 0213^*$.

Given an attribute A, I say a *generalization for an attribute* is a function on A. That is, each f: $A \rightarrow B$ is a generalization. I also say that:

$$A_o \xrightarrow{f_0} A_1 \xrightarrow{f_1} \ldots \xrightarrow{f_{n-1}} A_n$$

is a generalization sequence or a functional generalization sequence.

Given an attribute A of a private table PT, I define a **domain generalization hierarchy** DGH_A for A as a set of functions f_h : h=0,…,n-1 such that:

$$A_o \xrightarrow{f_0} A_1 \xrightarrow{f_1} \ldots \xrightarrow{f_{n-1}} A_n$$

A=A_0 and $|A_n| = 1$. DGH_A is over: $\bigcup_{h=0}^{n} A_h$

Clearly, the f_h's impose a linear ordering on the A_h's where the minimal element is the ground domain A_0 and the maximal element is A_n. The singleton requirement on A_n ensures that all values associated with an attribute can eventually be generalized to a single value. In this presentation I assume A_h, h=0,…,n, are disjoint; if an implementation is to the contrary and there are elements in common, then DGH_A is over the disjoint sum of A_h's and definitions change accordingly.

Given a domain generalization hierarchy DGH_A for an attribute A, if $v_i \in A_i$ and $v_j \in A_j$ then I say $v_i \le v_j$ if and only if $i \le j$ and:

$$f_{j-1}(\ldots f_i(v_i)\ldots) = v_j$$

This defines a *partial ordering* ≤ on: $\bigcup_{h=0}^{n} A_h$

Such a relationship implies the existence of a **value generalization hierarchy** VGH_A for attribute A.

I expand my representation of generalization to include suppression by imposing on each value generalization hierarchy a new maximal element, atop the old maximal element. The new maximal element is the attribute's suppressed

value. The height of each value generalization hierarchy is thereby incremented by one. No other changes are necessary to incorporate suppression. Figure 2 and Figure 3 provides examples of domain and value generalization hierarchies expanded to include the suppressed maximal element (*****). In this example, domain Z_0 represents ZIP codes for Cambridge, MA, and E_0 represents race. From now on, all references to generalization include the new maximal element; and, hierarchy refers to domain generalization hierarchies unless otherwise noted.

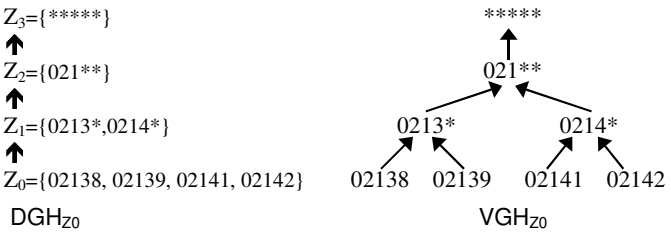

Figure 2 ZIP domain and value generalization hierarchies including suppression

Figure 3 Race domain and value generalization hierarchies including suppression

3.2. Minimal generalization of a table

Given table PT, generalization can be effective in producing a table RT based on PT that adheres to *k*-anonymity because values in RT are substituted with their generalized replacements. The number of distinct values associated with each attribute is non-increasing, so the substitution tends to map values to the same result, thereby possibly decreasing the number of distinct tuples in RT.

A generalization function on tuple t with respect to $A_1,\ldots, A_n$ is a function f_t on $A_1\times\ldots\times A_n$ such that: $f_t(A_1,\ldots,A_n)=(f_{t1}(A_1),\ldots,f_{tn}(A_n))$

where for each i: 1,…,n, f_{ti} is a generalization of the value $t[A_i]$. The function f_t is a set function. I say f_t is generated by the f_{ti}'s.

Given f, $A_1,\ldots,A_n$, a table $T(A_1,\ldots,A_n)$ and a tuple $t\in T$, i.e., $t(a_1,\ldots,a_n)$

$$g(T)=\left\{k\cdot f(t): t\in T \text{ and } \left|f^{-1}(f(t))\right|=k\right\}$$

The function g is a multi-set function and f^{-1} is the inverse function of f. I say that g is the multi-set function generated by f and by the f_i's. Further, I say that *$g(\mathsf{T})$ is a generalization of table* T. This does not mean, however, that the generalization respects the value generalization hierarchy for each attribute in T. To determine whether one table is a generalization with respect to the value generalization hierarchy of each attribute requires analyzing the values themselves.

Let DGH_i be the domain generalization hierarchies for attributes A_{li} where i=1,…,A_n Let $\mathsf{T}_l[A_{l1},\ldots,A_{lAn}]$ and $\mathsf{T}_m[A_{m1},\ldots,A_{mAn}]$ be two tables such that for each i:1,..,n, $A_{li},A_{mi} \in \mathsf{DGH}_i$. Then, I say table T_m is **a generalization of table** T_l, written $\mathsf{T}_l \le \mathsf{T}_m$, if and only if there exists a generalization function g such that $g[\mathsf{T}_l] = \mathsf{T}_m$ and is generated by f_i's where: $\forall t_l \in \mathsf{T}_l$, $a_{li} \le f_i(a_{li}) = a_{mi}$ *and* $f_i : A_{li} \rightarrow A_{mi}$ and each f_i is in the DGH_i of attribute A_{li}. From this point forward, I will use the term *generalization* (as a noun) to denote a generalization of a table.

Race E_0	ZIP Z_0
Black	02138
Black	02139
Black	02141
Black	02142
White	02138
White	02139
White	02141
White	02142

PT

Race E_1	ZIP Z_0
Person	02138
Person	02139
Person	02141
Person	02142
Person	02138
Person	02139
Person	02141
Person	02142

$GT_{[1,0]}$

Race E_1	ZIP Z_1
Person	0213*
Person	0213*
Person	0214*
Person	0214*
Person	0213*
Person	0213*
Person	0214*
Person	0214*

$GT_{[1,1]}$

Race E_0	ZIP Z_2
Black	021**
Black	021**
Black	021**
Black	021**
White	021**
White	021**
White	021**
White	021**

$GT_{[0,2]}$

Race E_0	ZIP Z_1
Black	0213*
Black	0213*
Black	0214*
Black	0214*
White	0213*
White	0213*
White	0214*
White	0214*

$GT_{[0,1]}$

Figure 4 Examples of generalized tables for PT

Definition 3. *k*-anonymity requirement of a generalized table

Let $\mathsf{PT}(A_1,\ldots,A_n)$ be a table, $\mathsf{QI}_{\mathsf{PT}}=\{A_i,\ldots,A_j\}$, be the quasi-identifier associated with PT where $\{A_i,\ldots,A_j\} \subseteq \{A_1,\ldots,A_n\}$, $\mathsf{RT}(A_i,\ldots,A_j)$ be a generalization of PT with respect to $\mathsf{QI}_{\mathsf{PT}}$, $t \in \mathsf{RT}[\mathsf{QI}_{\mathsf{PT}}]$ and k_t be the integer denoted in g for $f(t)$. RT is said to satisfy k-anonymity for k with respect to $\mathsf{QI}_{\mathsf{PT}}$ if $\forall t \in \mathsf{RT}[\mathsf{QI}_{\mathsf{PT}}]$, $k_t \ge k$ and $k \ge 2$.

The k-anonymity requirement guarantees each tuple in the generalized table $\mathsf{RT}[\mathsf{QI}_{\mathsf{PT}}]$ is indistinguishable from at least *k-1* other tuples in table $\mathsf{RT}[\mathsf{QI}_{\mathsf{PT}}]$.

Example 3. *k*-anonymity requirement of a generalized table

Consider PT and its generalized tables in Figure 4 and the hierarchies in Figure 2 and Figure 3. $\mathsf{GT}_{[0,1]}$ and $\mathsf{GT}_{[1,0]}$ satisfy k-anonymity for k = 2; and, $\mathsf{GT}_{[0,2]}$ and $\mathsf{GT}_{[1,1]}$ satisfy k-anonymity for k = 2, 3, 4.

The number of different generalizations of a table T, when generalization is enforced at the attribute level, is equal to the number of different combinations of domains that the attributes in the table can assume. Given domain generalization hierarchies DGH_i for attributes A_i, i:1,…,n; the number of generalizations, enforced at the attribute level, for table $T(A_1,\ldots,A_n)$ is:

$$\prod_{i=1}^{n}(|DGH_i|+1) \qquad \textbf{Equation 1}$$

Clearly, not all such generalizations are equally satisfactory. A generalization whose values result from generalizing each attribute to the highest possible level collapses all tuples in the table to the same list of values. This provides *k*-anonymity using more generalization than needed if a less generalized table exists which satisfies *k*-anonymity. This concept is captured in the following definition.

Definition 4. *k*-minimal generalization

Let $T_l(A_1,\ldots,A_n)$ and $T_m(A_1,\ldots,A_n)$ be two tables such that $T_l[QI_T] \leq T_m[QI_T]$, where $QI_T=\{A_i,\ldots,A_j\}$ is the quasi-identifier associated with the tables and $\{A_i,\ldots,A_j\} \subseteq \{A_1,\ldots,A_n\}$. T_m is said to be a minimal generalization of a table T_l with respect to a *k* anonymity requirement over QI_T if and only if:

1. T_m satisfies the *k*-anonymity requirement with respect to QI_T
2. $\forall T_z$: $T_l \leq T_z$, $T_z \leq T_m$, T_z satisfies the *k*-anonymity requirement with respect to $QI_T \Rightarrow T_z[A_i,\ldots,A_j] = T_m[A_i,\ldots,A_j]$.

Example 4. *k*-minimal generalization

Figure 4 shows generalizations, enforced at the attribute level, of PT over {*Race*, *ZIP*}. Each generalization satisfies *k*-anonymity for k=2. $GT_{[0,1]}$ generalized *ZIP* one level. $GT_{[1,0]}$ generalized *Race* one level. So, $GT_{[1,1]}$, and $GT_{[0,2]}$ did more generalization than necessary. $GT_{[0,2]}$ is a generalization of $GT_{[0,1]}$. $GT_{[1,1]}$ is a generalization of both $GT_{[1,0]}$ and $GT_{[0,1]}$.

So, table T_m, generalization of T_l, is *k*-minimal if it satisfies *k*-anonymity and there does not exist a generalization of T_l satisfying *k*-anonymity of which T_m is a generalization.

3.3. Minimal distortion of a table

When different *k*-minimal generalizations exist, preference criteria can be applied to choose a solution among them. A way of preferring one to another is to select one whose information is most useful. Generalizing a table T results in a table T' that typically has less information than T; and so, T' is considered less useful. In order to capture the information loss, I define an information theoretic metric that

reports the amount of distortion in a generalized table.[5] In a cell of a generalized table, the ratio of the domain of the value found in the cell to the height of the attribute's hierarchy reports the amount of generalization and thereby measures the cell's distortion. *Precision* of a generalized table is then one minus the sum of all cell distortions (normalized by the total number of cells), as defined below.

Definition 5. precision metric *Prec*

Let $\mathsf{PT}(A_1,...,A_{Na})$ be a table, $t_{Pj} \in \mathsf{PT}$, $\mathsf{RT}(A_1,...,A_{Na})$ be a generalization of PT, $t_{Pj} \in \mathsf{PT}$, each $\mathsf{DGH_A}$ be the domain generalization hierarchy for attribute A, and f_i's be generalizations on A. The precision of RT, written $Prec(\mathsf{RT})$, based on generalization and suppression is:

$$Prec(\mathsf{RT}) = 1 - \frac{\sum_{i=1}^{N_A} \sum_{j=1}^{N} \frac{h}{|\mathsf{DGH}_{Ai}|}}{|\mathsf{PT}| \bullet |N_A|} \quad where\ f_1(\ldots f_h(t_{Pj}[A_i])\ldots) = t_{Rj}[A_i]$$

Example 5. precision metric

In the case where $\mathsf{PT} = \mathsf{RT}$, each value is in the ground domain so each $h = 0$; therefore, $Prec(\mathsf{RT}) = 1$. Conversely, in the case where each value in RT is the maximal element of its hierarchy, each $h = |\mathsf{DGH_{Ai}}|$; and so, $Prec(\mathsf{RT}) = 0$.

Example 6. precision metric

Using the hierarchies in Figure 2 and Figure 3, the precision of the generalizations of PT shown in Figure 4 are: $Prec(\mathsf{GT}_{[1,0]}) = 0.75$; $Prec(\mathsf{GT}_{[1,1]}) = 0.58$; $Prec(\mathsf{GT}_{[0,2]}) = 0.67$; and, $Prec(\mathsf{GT}_{[0,1]}) = 0.83$. Each of these satisfy *k*-anonymity for k=2, but $\mathsf{GT}_{[0,1]}$ does so with the least distortion. Notice $\mathsf{GT}_{[1,0]}$ and $\mathsf{GT}_{[0,1]}$ each generalize values up one level, but because $|\mathsf{DGH_{Race}}| = 2$ and $|\mathsf{DGH_{ZIP}}| = 3$, $Prec(\mathsf{GT}_{[0,1]}) > Prec(\mathsf{GT}_{[1,0]})$.

Generalizations based on attributes with taller generalization hierarchies typically maintain precision better than generalizations based on attributes with shorter hierarchies. Further, hierarchies with different heights can provide different *Prec* measures for the same table. So, the construction of generalization hierarchies is part of the preference criteria. *Prec* best measures the quality of the data when the set of hierarchies used contain only values semantically useful. There is no need to arbitrarily increase the heights of hierarchies solely to prefer one attribute to another. Instead, weights can be assigned to attributes in *Prec* to make the preference explicit [9].

[5] While entropy is the classical measure used in information theory to characterize the purity of data [5], a metric based on the semantics of generalization can be more discriminating than the direct comparison of the encoding lengths of the values stored in the table.

As stated earlier, not all *k*-minimal generalizations are equally distorted and preference can be based on the *k*-minimal generalization having the most precision. This concept is captured by the following definition.

Definition 6. *k*-minimal distortion

Let $\mathsf{T}_I(A_1,\ldots,A_n)$ and $\mathsf{T}_m(A_1,\ldots,A_n)$ be two tables such that $\mathsf{T}_I[\mathsf{QI}_T] \leq \mathsf{T}_m[\mathsf{QI}_T]$, where $\mathsf{QI}_T=\{A_i,\ldots,A_j\}$ is the quasi-identifier associated with the tables and $\{A_i,\ldots,A_j\} \subseteq \{A_1,\ldots,A_n\}$ and $\forall x=i,\ldots,j$, DGH_{Ax} are domain generalization hierarchies for QI_T. T_m is said to be a minimal distortion of a table T_I with respect to a *k* anonymity requirement over QI_T if and only if:

1. T_m satisfies the *k*-anonymity requirement with respect to QI_T
2. $\forall \mathsf{T}_z$: $Prec(\mathsf{T}_I) \geq Prec(\mathsf{T}_z)$, $Prec(\mathsf{T}_z) \geq Prec(\mathsf{T}_m)$, T_z satisfies the *k*-anonymity requirement with respect to $\mathsf{QI}_T \Rightarrow \mathsf{T}_z[A_i,\ldots,A_j] = \mathsf{T}_m[A_i,\ldots,A_j]$.

Consider the values reported in Example 6 about the tables in Figure 4. Only $\mathsf{GT}_{[0,1]}$ is a *k*-minimal distortion of PT. A *k*-minimal distortion is specific to a table, a quasi-identifier, a set of domain generalization hierarchies for the attributes of the quasi-identifier, and *Prec* (or a weighted *Prec*).

It is trivial to see that a table that satisfies *k*-anonymity has a unique *k*-minimal distortion, which is itself. It is also easy to see that a generalized table RT that is a *k*-minimal distortion of table PT is also a *k*-minimal generalization of PT, as stated in the following theorem.

Theorem 2

Given tables T_I and T_m such that $\mathsf{T}_I \leq \mathsf{T}_m$ and T_m satisfies *k*-anonymity. T_m is a *k*-minimal distortion of $\mathsf{T}_I \Rightarrow \mathsf{T}_m$ is *k*-minimal generalization of T_I.

3.4. Algorithm for finding a minimal generalization with minimal distortion

The algorithm presented in this section combines these formal definitions into a theoretical model against which real-world systems will be compared.

Figure 5 presents an algorithm, called MinGen, which, given a table $\mathbf{PT}(A_x,\ldots,A_y)$, a quasi-identifier $\mathsf{QI}=\{A_1,\ldots,A_n\}$, where $\{A_1,\ldots,A_n\} \subseteq \{A_x,\ldots,A_y\}$, a *k*-anonymity constraint, and domain generalization hierarchies DGH_{Ai}, produces a table **MGT** which is a *k*-minimal distortion of **PT**[*QI*]. It assumes that $k < |\mathbf{PT}|$, which is a necessary condition for the existence of a *k*-minimal generalization.

The steps of the MinGen algorithm are straightforward. [step 1] Determine if the original table, PT, itself satisfies the *k*-anonymity requirement; and if so, it is the *k*-minimal distortion. In all other cases execute step 2. [step 2.1] Store the set of all possible generalizations of PT over QI into *allgens*. [step 2.2] Store those generalizations from *allgens* that satisfy the *k*-anonymity requirement into

protected. [step 2.3] Store the *k*-minimal distortions (based on *Prec*) from *protected* into *MGT*. It is guaranteed that $|MGT| \geq 1$. [step 2.4] Finally, the function *preferred()* returns a single *k*-minimal distortion from *MGT* based on user-defined specifications[6].

Input: Private Table **PT**; quasi-identifier QI = (A_1, …, A_n),
k constraint; domain generalization hierarchies
DGH_{Ai}, where *i*=1,…,*n*, and *preferred()* specifications.
Output: MGT, a minimal distortion of PT[QI] with respect to *k*
chosen according to the preference specifications
Assumes: |PT |≥ *k*
Method:

1. **if PT**[QI] satisfies *k*-anonymity requirement with respect to *k* **then do**
   1.1. MGT ← { **PT** } // **PT** is the solution
2. **else do**
   2.1. *allgen* ← {T_i : T_i is a generalization of **PT** over QI}
   2.2. *protected* ← {T_i : $T_i \in$ *allgen* ∧ T_i satisfies *k*-anonymity of *k*}
   2.3. MGT ← {T_i : $T_i \in$ *protected* ∧ there does not exist $T_z \in$ *protected* such that *Prec*(T_z) > *Prec*(T_i) }
   2.4. MGT ← **preferred**(MGT) // select the preferred solution
3. **return** MGT

Figure 5 Preferred Minimal Generalization (MinGen) Algorithm

Example 7. MinGen produces *k*-minimal distortions

Let the hierarchies in Figure 2, Figure 3 and Figure 6 for the quasi-identifier QI={*Race*, *BirthDate*, *Gender*, *ZIP*}, the table PT in Figure 7, and a *k*-anonymity constraint of *k*=2 be provided to MinGen. After step 2.3 of MinGen, *MGT*= {GT}, where GT is shown in Figure 7. GT is a *k*-minimal distortion of PT over QI and a *k*-minimal generalization of PT over QI.

It can be proved that a generalization of a table T over a quasi-identifier QI, that satisfies a given *k*-anonymity requirement, and has the least amount of distortion of all possible generalizations of T over QI, is a *k*-minimal distortion of T over QI with respect to *Prec*. From Theorem 2, the solution is also a *k*-minimal generalization of T over QI.

With respect to complexity, MinGen makes no claim to be efficient. If generalization is enforced at the attribute level, the number of possible

[6] The *preferred()* function returns only one table as a solution. The single solution requirement is a necessary condition because the chosen solution becomes part of the join of external information against which subsequent linking must be protected.

generalizations, |*allgens*|, is expressed in Equation 1. If generalization is enforced at the cell level, $|allgens| = \prod_{i=1}^{n} \left(|DGH_{Ai}| + 1\right)^{|PT|}$. **Equation 2**

Clearly, an exhaustive search of all possible generalizations is impractical even on modest sized tables. So, how do real-world systems find solutions in real-time?

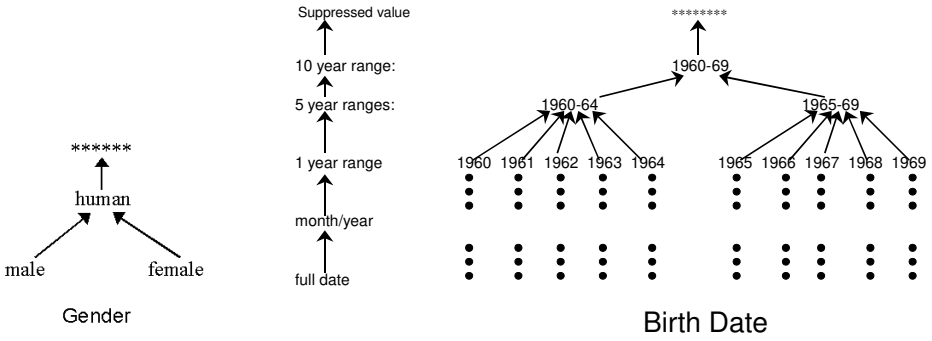

Figure 6 Value generalization hierarchies for {*Gender*, *BirthDate*} with suppression

Race	BirthDate	Gender	ZIP	Problem
black	9/20/1965	male	02141	short of breath
black	2/14/1965	male	02141	chest pain
black	10/23/1965	female	02138	painful eye
black	8/24/1965	female	02138	wheezing
black	11/7/1964	female	02138	obesity
black	12/1/1964	female	02138	chest pain
white	10/23/1964	male	02138	short of breath
white	3/15/1965	female	02139	hypertension
white	8/13/1964	male	02139	obesity
white	5/5/1964	male	02139	fever
white	2/13/1967	male	02138	vomiting
white	3/21/1967	male	02138	back pain

PT *Prec*=1.00

Race	BirthDate	Gender	ZIP	Problem
black	1965	male	02141	short of breath
black	1965	male	02141	chest pain
person	1965	female	0213*	painful eye
person	1965	female	0213*	wheezing
black	1964	female	02138	obesity
black	1964	female	02138	chest pain
white	1960-69	male	02138	short of breath
person	1965	female	0213*	hypertension
white	1964	male	02139	obesity
white	1964	male	02139	fever
white	1960-69	male	02138	vomiting
white	1960-69	male	02138	back pain

GT *Prec*=0.90

Figure 7 *k*-minimal distortion for PT where *k*=2

4. Real-world results

Here are two real-world systems that seek to provide *k*-anonymity protection using generalization and suppression. They are: (1) my Datafly[7] Systems [5]; and, (2) Statistics Netherlands' μ-Argus System [6]. This section shows that Datafly can over distort the data and that μ-Argus can fail to provide adequate protection.

[7] The systems Datafly and Datafly II refer to two kindred algorithms. The differences between them do not substantially alter the findings reported herein, so in this writing, the term Datafly refers to a simplified abstraction of these algorithms known as the core Datafly algorithm [9].

4.1. The Datafly System

Here is a summary of the setting in which the core Datafly algorithm operates. The data holder (1) declares specific attributes and tuples in the original private table (PT) as being eligible for release. The data holder also (2) groups a subset of attributes of PT into one or more quasi-identifiers (QI_i) and assigns (3) a weight from 0 to 1 to each attribute within each QI_i that specifies the likelihood the attribute will be used for linking; a 0 value means not likely and a value of 1 means highly probable. The data holder (4) specifies a minimum anonymity level that computes to a value for *k*. Finally, (5) a weight from 0 to 1 is assigned to each attribute within each QI_i to state a preference of which attributes to distort; a 0 value means the recipient of the data would prefer the values not to be changed and a value of 1 means maximum distortion could be tolerated. For convenience in this discussion, I remove many of the finer features. I consider a single quasi-identifier, where all attributes of the quasi-identifier have equal preference and an equal likelihood for linking so the weights can be considered as not being present.

Figure 8 presents the core Datafly algorithm, which, given a table **PT**(A_x,…,A_y), a quasi-identifier QI={A_1,…,A_n}, where {A_1,…,A_n} ⊆ {A_x,…,A_y}, a *k*-anonymity constraint, and domain generalization hierarchies DGH_{Ai}, produces a table **MGT** which is a generalization of **PT**[*QI*] that satisfies *k*-anonymity. It assumes that *k* < |**PT**|, which is a necessary condition for satisfying *k*-anonymity.

Input: Private Table **PT**; quasi-identifier QI = (A_1, …, A_n),
k constraint; hierarchies DGH_{Ai}, where *i*=1,…,*n*.
Output: MGT, a generalization of PT[QI] with respect to *k*
Assumes: |PT |≥ *k*
Method:

1. freq ← a frequency list contains distinct sequences of values of PT[QI], along with the number of occurrences of each sequence.
2. **while there exists** sequences in freq occurring less than *k* times that account for more than *k* tuples **do**
   2.1. **let** A_j be attribute in freq having the most number of distinct values
   2.2. freq ← generalize the values of A_j in freq
3. freq ← suppress sequences in freq occurring less than *k* times.
4. freq ← enforce *k* requirement on suppressed tuples in freq.
5. **Return** MGT ← construct table from freq

Figure 8 Core Datafly Algorithm

The core Datafly algorithm has few steps. Step 1 constructs freq, which is a frequency list containing distinct sequences of values from PT[QI], along with the number of occurrences of each sequence. Each sequence in freq represents one or more tuples in a table. Step 2.1 uses a heuristic to guide generalization. The attribute having the most number of distinct values in freq is generalized.

Generalization continues until there remains *k* or fewer tuples having distinct sequences in freq. Step 3 suppresses any sequences of freq occurring less than *k* times. Complimentary suppression is performed in step 4 so that the number of suppressed tuples satisfies the *k* requirement. Finally, step 5 produces a table MGT, based on freq such that the values stored as a sequence in freq appear as tuple(s) in MGT replicated in accordance to the stored frequency.

Let the hierarchies in Figure 2, Figure 3 and Figure 6 for the quasi-identifier QI={*Race*, *BirthDate*, *Gender*, *ZIP*}, the table PT in Figure 7, and a *k*-anonymity constraint of *k*=2 be provided to Datafly. Figure 9A shows the contents of freq after step 1. *Birthdate* has the most number of distinct values (12) so its values are generalized. Figure 9B shows the contents of freq after step 2. Tuples t7 and *t8* will be suppressed. Table MGT in Figure 10 is the final result. MGT[QI] satisfies *k*-anonymity for *k*=2 with *Prec*(MGT[QI]) = 0.75. However, from Example 7 and Figure 7, GT is a *k*-minimal distortion for PT with *Prec*(GT[QI])=0.90. So, Datafly distorted the results more than needed.

Race	BirthDate	Gender	ZIP	#occurs	
black	9/20/65	male	02141	1	t1
black	2/14/65	male	02141	1	t2
black	10/23/65	female	02138	1	t3
black	8/24/65	female	02138	1	t4
black	11/7/64	female	02138	1	t5
black	12/1/64	female	02138	1	t6
white	10/23/64	male	02138	1	t7
white	3/15/65	female	02139	1	t8
white	8/13/64	male	02139	1	t9
white	5/5/64	male	02139	1	t10
white	2/13/67	male	02138	1	t11
white	3/21/67	male	02138	1	t12
2	12	2	3		

A

Race	BirthDate	Gender	ZIP	#occurs	
black	1965	male	02141	2	t1,t2
black	1965	female	02138	2	t3, t4
black	1964	female	02138	2	t5, t6
white	1964	male	02138	1	t7
white	1965	female	02139	1	t8
white	1964	male	02139	2	t9, t10
white	1967	male	02138	2	t11, t12
2	3	2	3		

B

Figure 9 Intermediate stages of the core Datafly algorithm

Race	BirthDate	Gender	ZIP	Problem
black	1965	male	02141	short of breath
black	1965	male	02141	chest pain
black	1965	female	02138	painful eye
black	1965	female	02138	wheezing
black	1964	female	02138	obesity
black	1964	female	02138	chest pain
white	1964	male	02139	obesity
white	1964	male	02139	fever
white	1967	male	02138	vomiting
white	1967	male	02138	back pain

Figure 10 Table MGT resulting from Datafly, *k*=2, QI={*Race*, *Birthdate*, *Gender*, *ZIP*}

The core Datafly algorithm does not necessarily provide *k*-minimal generalizations or *k*-minimal distortions, even though it can be proved that its solutions always

satisfy *k*-anonymity. One of the problems is that Datafly makes crude decisions – generalizing all values associated with an attribute and suppressing all values within a tuple. MinGen makes decisions at the cell-level and by doing so, can provide results with more precision. Another problem is the heuristic that selects the attribute with the greater number of distinct values as the one to generalize. This may be computationally efficient, but can be shown to perform unnecessary generalization. In summary, Datafly produces generalizations that satisfy *k*-anonymity, but such generalizations may not be *k*-minimal distortions.

4.2. The μ-Argus System

In μ-Argus, the data holder provides a value for *k* and specifies which attributes are sensitive by assigning a value between 0 and 3 to each attribute. These correspond to "not identifying," "most identifying," "more identifying," and "identifying," respectively. μ-Argus then identifies rare and therefore unsafe combinations by testing 2- and 3-combinations of attributes. Unsafe combinations are eliminated by generalizing attributes within the combination and by cell suppression. Rather than removing entire tuples, μ-Argus suppresses values at the cell-level. The resulting data typically contain all the tuples and attributes of the original data, though values may be missing in some cell locations.

Figure 11 presents the μ-Argus algorithm. Given a table $\mathbf{PT}(A_x,\ldots,A_y)$, a quasi-identifier $QI=\{A_1,\ldots,A_n\}$, where $\{A_1,\ldots,A_n\} \subseteq \{A_x,\ldots,A_y\}$, disjoint subsets of QI known as *Identifying*, *More*, and *Most* where $QI = Identifying \cup More \cup Most$, a *k*-anonymity constraint, and domain generalization hierarchies DGH_{Ai}, μ-Argus produces a table **MT** which is a generalization of **PT**[QI].

The basic steps of the μ-Argus algorithm are provided in Figure 11. This algorithm results from my reverse engineering an implementation [9]. Shortcomings of the actual μ-Argus implementation were found. So, results from both are reported. In general, the constructed μ-Argus algorithm generates solutions that are better protected than those released by the actual program.

The program begins in step 1 by constructing freq, which is a frequency list containing distinct sequences of values from PT[QI], along with the number of occurrences of each sequence. In step 2, the values of each attribute are automatically generalized until each value associated with an attribute in the quasi-identifier QI appears at least *k* times. This is a necessary condition for *k*-anonymity (see Theorem 1). In step 3, the program automatically tests combinations of attributes to identify those combinations of attributes whose assigned values in combination do not appear at least *k* times; these combinations are stored in *outliers*. Afterwards, the data holder, in steps 4 and 5, decides whether to generalize an attribute in QI that has values in *outliers* and if so, selects the attribute to generalize. Finally, in step 6, μ-Argus automatically suppresses a

value from each combination in *outliers*. Precedence is given to the value occurring most often in order to reduce the total number of suppressions.

Input: Private Table **PT**; quasi-identifier QI = (A_1, …, A_n),
disjoint subsets of QI known as *Identifying*, *More*, and
Most where QI = *Identifying* ∪ *More* ∪ *Most*, *k* constraint;
domain generalization hierarchies DGH_{Ai}, where i=1,…,n.
Output: MT containing a generalization of PT[QI]
Assumes: |PT| ≥ *k*
Method:

1. freq ← a frequency list containing distinct sequences of values of PT[QI], along with the number of occurrences of each sequence.
2. Generalize each A_i∈ QI in freq until its assigned values satisfy *k*.
3. Test 2- and 3- combinations of *Identifying*, *More* and *Most* and **let** *outliers* store those cell combinations not having *k* occurrences.
4. Data holder decides whether to generalize an A_j∈ QI based on *outliers* and if so, identifies the A_j to generalize. freq contains the generalized result.
5. **Repeat** steps 3 and 4 until the data holder no longer elects to generalize.
6. Automatically suppress a value having a combination in *outliers*, where precedence is given to the value occurring in the most number of combinations of *outliers*.

Figure 11 µ-Argus algorithm

One shortcoming of the actual µ-Argus implementation appears in step 3 of Figure 11. The actual program does not test all 2- and 3- combinations; this may be a programming error. Figure 12 reports which combinations µ-Argus tests. Six combinations (not listed) are not tested at all. It is easy to have tables in which values appear in combinations not examined by µ-Argus.

Combinations Always Tested
Identifying × *More* × *Most*, *Identifying* × *Most* × *Most*, *Most* × *Most* × *Most*, *Identifying* × *More*, *Identifying* × *Most*, *More* × *Most*, *Most* × *Most*
Combinations Tested only if |*Identifying*| > 1
More × *More* × *Most*, *Most* × *Most* × *More*, *More* × *More*

Figure 12 Combinations of *More*, *Most*, *Identifying* tested by µ-Argus

Let the hierarchies in Figure 2, Figure 3 and Figure 6 for the quasi-identifier QI={*Race*, *BirthDate*, *Gender*, *ZIP*} where *Most* = {*BirthDate*}, *More* = {*Gender*, *ZIP*} and *Identifying* = {*Race*}, the table PT in Figure 7, and a *k*-anonymity constraint of *k*=2 be provided to µ-Argus. In Figure 13, V shows the result of testing *Most* × *More* at step, and freq is updated to show {*BirthDate*, *ZIP*} for *t8* did not satisfy *k*. Figure 14 shows freq before step 6; the values to be suppressed are underlined. Table MT in Figure 15 is the final result from the µ-Argus

algorithm provided in Figure 11. The *k*-anonymity requirement is not enforced on suppressed values, making it vulnerable to linking as well as to an inference attack using summary data[8]. For example, knowing the total number of men and women allows the suppressed values for *Gender* to be inferred.

Birth	ZIP	*occurs*	*sid*	*outliers*
1965	02141	2	{t1,t2}	{}
1965	02138	2	{t3,t4}	{}
1964	02138	3	{t5,t6,t7}	{}
1965	02139	1	{t8}	{}
1964	02139	2	{t9,t10}	{}
1967	02138	2	{t11,t12}	{}

V

Race	Birth	Sex	ZIP	*occurs*	*sid*	*outliers*
black	1965	male	02141	2	{t1,t2}	{}
black	1965	female	02138	2	{t3,t4}	{}
black	1964	female	02138	2	{t5,t6}	{}
white	1964	male	02138	1	{t7}	{}
white	1965	female	02139	1	{t8}	**{{birth,zip}}**
white	1964	male	02139	2	{t9,t10}	{}
white	1967	male	02138	2	{t11,t12}	{}

freq

Figure 13 *Most* × *More* combination test and resulting freq

Race	Birth	Sex	ZIP	*occurs*	*sid*	*outliers*
black	1965	male	02141	2	{t1,t2}	{}
black	1965	female	02138	2	{t3,t4}	{}
black	1964	female	02138	2	{t5,t6}	{}
white	1964	male	02138	1	{t7}	{{**birth**,sex,zip}, {race,**birth**,zip)}
white	1965	female	02139	1	{t8}	{{**birth**,zip}, {**sex**,zip}, {**birth**,**sex**,zip}, {race,**birth**,**sex**}, {race,**birth**,zip}, {race,**sex**}, {race,**birth**)}
white	1964	male	02139	2	{t9,t10}	{}
white	1967	male	02138	2	{t11,t12}	{}

Figure 14 freq before suppression

id	Race	BirthDate	Gender	ZIP
t1	black	1965	male	02141
t2	black	1965	male	02141
t3	black	1965	female	02138
t4	black	1965	female	02138
t5	black	1964	female	02138
t6	black	1964	female	02138
t7	white		male	02138
t8	white			02139
t9	white	1964	male	02139
t10	white	1964	male	02139
t11	white	1967	male	02138
t12	white	1967	male	02138

MT

id	Race	BirthDate	Gender	ZIP
t1	black	1965	male	02141
t2	black	1965	male	02141
t3	black	1965	female	02138
t4	black	1965	female	02138
t5	black	1964	female	02138
t6	black	1964	female	02138
t7	white	1964	male	02138
t8	white		female	02139
t9	white	1964	male	02139
t10	white	1964	male	02139
t11	white	1967	male	02138
t12	white	1967	male	02138

MT actual

Figure 15 Results from the μ-Argus algorithm and from the program

The actual μ-Argus program provides MTactual shown in Figure 15. The tuple identified as *t7* is ["white", "1964", "male", "02138"], which is unique over MTactual[QI]. Therefore, MTactual does not satisfy the requirement for *k*=2.

[8] See [3] for a detailed discussion of attacks on *k*-anonymity.

A shortcoming of μ-Argus stems from not examining all combinations of the attributes in the quasi-identifier. Only 2- and 3- combinations are examined. There may exist 4-combinations or larger that are unique. Directly extending μ-Argus to compute on all combinations loses its computational efficiency. So, generalizations from μ-Argus may not always satisfy *k*-anonymity, even though all generalizations from Datafly do satisfy *k*-anonymity.

Both algorithms may provide generalizations that are not *k*-minimal distortions because they both enforce generalization at the attribute level. This renders crude *Prec* measures. There may exist values in the table that when generalized at the cell level, satisfy *k* without modifying all values in the attribute. In summary, more work is needed to correct these heuristic-based approaches.

Acknowledgments

Vicenc Torra and Josep Domingo provided the opportunity to write this paper. Jon Doyle gave editorial suggestions on a very early draft. Pierangela Samarati recommended I engage the material formally and started me in that direction in 1998. Finally, I thank the corporate and government members of the Laboratory for International Data Privacy at Carnegie Mellon University for the opportunity to work on real-world data anonymity problems.

References

1 L. Sweeney, Information Explosion. *Confidentiality, Disclosure, and Data Access: Theory and Practical Applications for Statistical Agencies*, L. Zayatz, P. Doyle, J. Theeuwes and J. Lane (eds), Urban Institute, Washington, DC, 2001.

2 L. Sweeney, *Uniqueness of Simple Demographics in the U.S. Population*, LIDAP-WP4. Carnegie Mellon University, Laboratory for International Data Privacy, Pittsburgh, PA: 2000. Forthcoming book entitled, *The Identifiability of Data*.

3 L. Sweeney. *k*-Anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10 (7), 2002.

4 T. Dalenius. Finding a needle in a haystack – or identifying anonymous census record. *Journal of Official Statistics,* 2(3):329-336, 1986.

5 L. Sweeney. Guaranteeing anonymity when sharing medical data, the Datafly system. *Proceedings, Journal of the American Medical Informatics Association*. Washington, DC: Hanley & Belfus, Inc., 1997.

6 A. Hundepool and L. Willenborg. μ- and τ-argus: software for statistical disclosure control. *Third International Seminar on Statistical Confidentiality.* Bled: 1996.

7 J. Ullman. *Principles of Database and Knowledge Base Systems*. Computer Science Press, Rockville, MD. 1988.

8 L. Sweeney, *Uniqueness of Simple Demographics in the U.S. Population*, LIDAP-WP4. Carnegie Mellon University, Laboratory for International Data Privacy, Pittsburgh, PA: 2000. Forthcoming book entitled, *The Identifiability of Data*.

9 L. Sweeney, *Computational Disclosure Control: A primer on data privacy protection*. Ph.D. Thesis, Massachusetts Institute of Technology, 2001.