Table Ambiguation By Minimal Cell Suppression: Problem Characterization And Effect On Functional Dependencies

DSG Technical Report 2001

Staal A. Vinterbo [a,b,*], Lucila Ohno-Machado [a,b,c]

[a] *Decision Systems Group, Brigham and Women's Hospital, 75 Francis Street, Boston, MA 02115, U.S.A.*

[b] *Harvard Medical School, Boston, MA, USA*

[c] *Division of Health Sciences and Technology, Harvard Medical School/MIT, Boston, MA, USA*

Abstract

Ambiguous relational data are of interest in the field of disclosure control as they hinder the induction of functional dependencies. We define a data table to be ambiguous if for any row i there exists another row j such that they agree upon all positions, called cells, that have not been designated a special value *suppressed*. We show that the problem of ambiguating a data table by a minimal number of cell suppressions is equivalent to the minimal satisfiability problem for a certain Boolean function. We establish some properties of this function, including that it is a Min Horn Deletion-complete NP optimization problem, and present lower and upper bounds on the size of a minimal solution. We analyze a recently published method for table ambiguation based on cell suppression and compare this to two simple heuristics.

Keywords: cell suppression; minimal satisfiability; anonymity; disclosure avoidance; disclosure control; functional dependency

1 Introduction

The concern about protection of privacy is shared by a multitude of authors in the medical literature [3,22,20,8,7,6,5]. The protection of privacy rests, among

[*] Corresponding author.
Email address: `staal@dsg.harvard.edu` (Staal A. Vinterbo).

others, upon aspects of health care institutional policy such as patient data access limitations, administrative security (such as computer system integrity and access limitation mechanisms), and maintenance of patient and provider confidentiality in disseminated data. This exposition is concerned with the latter.

Focus in the past has been on hindering inferences using known functional dependencies in data. An example is the protection against linking disclosed combinations of non-explicit identifiers with explicit identifiers found elsewhere in public (or at least available to the attacking party) databases. Most medical data that are released to the research community are “de-identified” (i.e., explicit identifiers such as names and social security numbers have been deleted). Sweeney [19], however, reports that birth date and 5 digit zip code alone uniquely identifies 69% of the population in Cambridge, Massachusetts, showing that de-identification alone is not enough. Approaches to preventing inferences come in different flavors like, generalization [21,9], cell suppression [4,15], randomized transformations [13], and column suppression [18].

Our approach is complementary to the previous work listed in that it protects against the proof of existence of functional dependencies. This is relevant for the protection of both providers of health care and patients, as it decreases the number of logical conclusions that can be drawn from the data.

In this exposition, we define a data table to be ambiguous if for any row i there exists another row j such that they agree upon all positions, called cells, that have not been designated a special value *suppressed*. We characterize the problem of finding a minimal number of cell suppressions to achieve ambiguity in terms of a minimal propositional satisfiability problem, and show that ambiguation, as defined, disallows the inference of a certain family of functional dependencies. Bounds on the size of the minimal set of suppressions are proposed. We also analyze a previously published approach that ambiguates tables [15], and compare it to two simple heuristics on example data.

2 Methods

2.1 Problem Characterization

Let U be a finite set of size n of objects of interest, and define an *attribute* a on U to be a function $a : U \rightarrow V_a$ from U into an attribute value set V_a. A data table over $U = \{x_i\}$ and a set of attributes $A = \{a_j\}$ can be described as a matrix $T = \{c_{ij}\}$ where $c_{ij} = a_j(x_i)$ or $c_{ij} = \perp$. The latter indicates that the *cell* c_{ij} contains a value $\perp$ that (by definition) is indistinguishable from any element in V_{a_j}. The changing of $c_{ij} = a_j(x_i)$ to $c_{ij} =\perp$ is called *suppression* of the cell c_{ij}. Akin to Skowron et. al. [17], we define a *discernibility matrix* $M = \{m_{ij}\}$ on T, where m_{ij} contains the indices of columns in T where rows i and j are distinguishable. Formally:

$$m_{ij} = \{k | c_{ik} \not\doteq c_{jk}\},$$

where $a \doteq b \Leftrightarrow a = b \vee a =\perp \vee b =\perp$, i.e., $a \doteq b$ if and only if they are indistinguishable or indiscernible. Using M we define row i and j to be indistinguishable if and only if $m_{ij} = \emptyset$. Further, we can count the number of rows that are indistinguishable from a given row i. Let

$$\alpha_i = |\{j | m_{ij} = \emptyset\}|,$$

be that count. Note that $\alpha_i > 0$ as row i always is indistinguishable from itself. We are now able to give a precise problem formulation.

Problem 1 *Given a data table T, find a minimal set of cell suppressions such that $\alpha_i > 1$ for any row i in T.*

Example 1 *Consider the example data table in Table 1. The corresponding 4×4 discernibility matrix M is:*

a	b	c
1	1	2
3	1	1
3	2	2
1	1	2

Table 1
Example data table T, with attributes a, b, and c and four data rows.

$$M = \begin{bmatrix} \{\} & \{0, 2\} & \{0, 1\} & \{\} \\ \{0, 2\} & \{\} & \{1, 2\} & \{0, 2\} \\ \{0, 1\} & \{1, 2\} & \{\} & \{0, 1\} \\ \{\} & \{0, 2\} & \{0, 1\} & \{\} \end{bmatrix}.$$

Looking at M, we see that $\alpha_0 = \alpha_3 = 2$ because rows 0 and 3 in T are identical, while $\alpha_1 = \alpha_2 = 1$ as rows 1 and 2 are discernible from all other rows. The index set of rows that are distinguishable is then $\{1, 2\}$. Consider row 1 in

Table 1. The entry in M corresponding to row 1 and row 0 contains column numbers 0 and 2. Row 1 can thus be made indistinguishable from row 0 by suppressing the cells corresponding to these columns in row 0, i.e., suppressing cells $(0,0)$ and $(0,2)$. This results in row 0 looking like $\perp \quad 1 \quad \perp$. *Another possibility is to suppress these columns in row 1, i.e., suppressing the cells $(1,0)$ and $(1,2)$, making row 1 looking like* $\perp \quad 1 \quad \perp$. *Two other possibilities are to suppress either $(0,0)$ and $(1,2)$, resulting in rows 0 and 1 looking like* $\perp \quad 1 \quad 2$ *and* $3 \quad 1 \quad \perp$, *respectively, or $(1,0)$ and $(0,2)$, resulting in rows 0 and 1 looking like* $1 \quad 1 \quad \perp$ *and* $\perp \quad 1 \quad 1$, *respectively. Note that the listed possibilities are exactly the minimal sets of cell suppressions that will render rows 1 and 0 indistinguishable, and that they correspond to the possible combinations of rows 0 and 1, and columns 0 and 2.*

We will now exploit the observation made in Example 1 to present the solutions to Problem 1 in terms of satisfiability of a certain Boolean function [1].

To do this we need some technical definitions. Let $b_{(i,j)}$ be a Boolean variable associated with cell c_{ij}, i.e, there exists an invertible function var from $\mathbb{Z} \times \mathbb{Z}$ to a set of Boolean variables such that $b_{(i,j)} = \text{var}(i,j)$. Also, let $b_{(i,j)}$ reflect the status of cell c_{ij} being suppressed.

As we noted above, in order to make two distinguishable cells indistinguishable, we only need to suppress one of them, giving us a choice. Let S_{ij} represent the set of all minimal combinations of suppressions such that rows i and j can made indistinguishable. If we define S_{ij} as

$$S_{ij} = \times_{k \in m_{ij}} \{(i,k),(j,k)\},$$

then any "point" or tuple in S_{ij} represents one minimal way of achieving the indiscernibility of rows i and j.

Example 2 *For rows* 0 *and* 1 *in Table 1,*

$$\begin{aligned} S_{10} &= \{(0,0),(1,0)\} \times \{(0,2),(1,2)\} \\ &= \{((0,0),(0,2)),((0,0),(1,2)),((1,0),(0,2)),((1,0),(1,2))\} \end{aligned}$$

To convert a point in S_{ij}, which formally is not a set, to a set, we use the function L defined as

$$L(s_1, s_2, \ldots, s_m) = \{s_1, s_2, \ldots, s_m\}.$$

[1] For definitions and theory concerning Boolean functions and their application to reasoning, see Brown [1]

The function L applied to a tuple s returns the set of the tuple's elements.

Example 3 *For* $s = ((0,0),(0,2))$, $L(s) = \{(0,0),(0,2)\}$.

We can now formally express our possibilities given as points in S_{ij} as a Boolean formula. For a given point $s \in S_{ij}$, we let $t(s)$, defined as

$$t(s) = \bigwedge_{l \in L(s)} b_l,$$

be the term that assumes the value 1 when the cells corresponding to s are suppressed.

Example 4 *For* $s = ((0,0),(0,2))$, $t(s) = b_{(0,0)} \wedge b_{(0,2)}$.

The disjunction f_{ij} of all of the terms corresponding to the points in S_{ij} is defined as:

$$f_{ij} = \bigvee_{s \in S_{ij}} t(s).$$

We have that the Boolean equation $f_{ij} = 1$ holds if and only if $t(s) = 1$ for at least one $s \in S_{ij}$. As each $s \in S_{ij}$ corresponds to a set of cells whose suppression leads to the indiscernibility of rows i and j, and the corresponding variables in f_{ij} reflect the state of suppression of the corresponding cells, we now have that for a non-empty S_{ij}, $f_{ij} = 1$ if and only if rows i and j are indiscernible.

Let $I = \{i | \alpha_i = 1\}$ be the set of indices of rows in T that are distinguishable from all other rows in T. Given a data table T, with associated discernibility matrix M, let F_M be the Boolean formula

$$F_M = \bigwedge_{i \in I} \bigvee_{j \in J_i} f_{ij},$$

where $J_i = \{j | 0 \leq j < n \wedge m_{ij} \neq \emptyset\}$. By the above construction, we can formulate:

Theorem 1 *Let f be the Boolean function realized by F_M. Let S be the set of Boolean variables occurring in F_M, and associate with each $b_{(i,j)} \in S$ the suppression of c_{ij}. Let $\mathcal{S}$ denote the collection of sets $S' \subseteq S$ such that $f = 1$ when all variables in S' are assigned 1. The elements of $\mathcal{S}$ of minimal cardinality correspond exactly to the solutions of Problem 1 via the associated suppressions.*

Example 5
Again consider the example data table in Table 1. For this table and the corresponding matrix M, we have that

$$\begin{aligned}
&S_{10} = \{(0,0),(1,0)\} \times \{(0,2),(1,2)\}\\
&S_{12} = \{(1,1),(2,1)\} \times \{(1,2),(2,2)\}\\
&S_{13} = \{(1,0),(3,0)\} \times \{(1,2),(3,2)\}\\
&S_{20} = \{(0,0),(2,0)\} \times \{(0,1),(2,1)\}\\
&S_{21} = \{(1,1),(2,1)\} \times \{(1,2),(2,2)\}\\
&S_{23} = \{(2,0),(3,0)\} \times \{(2,1),(3,1)\}\\
&f_{10} = (b_{(0,0)} \wedge b_{(0,2)}) \vee (b_{(0,0)} \wedge b_{(1,2)}) \vee (b_{(1,0)} \wedge b_{(0,2)}) \vee (b_{(1,0)} \wedge b_{(1,2)})\\
&f_{12} = (b_{(1,1)} \wedge b_{(1,2)}) \vee (b_{(1,1)} \wedge b_{(2,2)}) \vee (b_{(2,1)} \wedge b_{(1,2)}) \vee (b_{(2,1)} \wedge b_{(2,2)})\\
&f_{13} = (b_{(1,0)} \wedge b_{(1,2)}) \vee (b_{(1,0)} \wedge b_{(3,2)}) \vee (b_{(3,0)} \wedge b_{(1,2)}) \vee (b_{(3,0)} \wedge b_{(3,2)})\\
&f_{20} = (b_{(0,0)} \wedge b_{(0,1)}) \vee (b_{(0,0)} \wedge b_{(2,1)}) \vee (b_{(2,0)} \wedge b_{(0,1)}) \vee (b_{(2,0)} \wedge b_{(2,1)})\\
&f_{21} = (b_{(1,1)} \wedge b_{(1,2)}) \vee (b_{(1,1)} \wedge b_{(2,2)}) \vee (b_{(2,1)} \wedge b_{(1,2)}) \vee (b_{(2,1)} \wedge b_{(2,2)})\\
&f_{23} = (b_{(2,0)} \wedge b_{(2,1)}) \vee (b_{(2,0)} \wedge b_{(3,1)}) \vee (b_{(3,0)} \wedge b_{(2,1)}) \vee (b_{(3,0)} \wedge b_{(3,1)})\\
&F_M = (f_{10} \vee f_{12} \vee f_{13}) \wedge (f_{20} \vee f_{21} \vee f_{23}).
\end{aligned}$$

As previously noted, the listed possibilities in example 1 comprise S_{10}. In general, the set S_{ij} represents all minimal sets of cell suppressions that will make row i and j indistinguishable. Associating a Boolean variable $b_{(i,j)}$ with cell (i,j), and assigning the value 1 to $b_{(i,j)}$ if (i,j) is suppressed, we see that $f_{ij} = 1$ if and only if row i is indistinguishable from row j. Similarly the disjunction of all f_{ik} for fixed i, e.g., $(f_{10} \vee f_{12} \vee f_{13})$ for $i = 1$ in the example, is 1 if and only if row i is indistinguishable from at least one other row. Continuing in this manner, we see that $F_M = 1$ if and only if row i is indistinguishable from at least one other row for all $i \in I$.

Given two Boolean functions f and g, we say that g is included in f, written $g \leq f$, in case the identity $gf' = 0$, where f' denotes the complement of f, is satisfied. An *implicant* of f is a term h such that $h \leq f$. A *prime implicant* is an implicant that ceases to be an implicant if any literal in it is removed. Let $PI(f)$ denote the set of all prime implicants of the Boolean function f. As F_M contains no negated literals, a consequence of Theorem 1 is that Problem 1 can be stated as the problem of finding a minimal cardinality prime implicant of the Boolean function represented by F_M.

Following an approach taken by Shaefer [16], the function F_M can be characterized further. Let $R \subseteq \{0,1\}^{k_i}$ be a non-empty logical relation of arity k_i. We view R as a truth table for a that relation, or alternatively, the truth table of a k_i-ary logical connective.

Example 6 *Consider binary conjunction and disjunction. The corresponding* *logical relations are* $R_c = \{(1,1)\}$ *and* $R_d = \{(0,1),(1,0),(1,1)\}$*, respectively. Given a pair (2-tuple) of variables,* (a,b)*, and symbols* R'_c *and* R'_d *representing the relations* R_c *and* R_d *respectively,* $R'_c(a,b) = a \wedge b$*, and* $R'_d(a,b) = a \vee b$*.*

Given a set $S = \{R_1, R_2, \ldots, R_m\}$, then a CNF($S$) formula is a conjunction of expressions of the form $R'_i(x_{i1}, x_{i2}, \ldots, x_{ik_i})$ where each R'_i is a symbol representing the relation R_i, and each x_{ij} is a Boolean variable. Further a CNF(S) formula can be classified according to the properties of the relations $R_i \in S$. Shaefer identified six such properties. They, together with identifying criteria [16,2] are given below. Let $R \subseteq \{0,1\}^l$, then R is

(1) *1-valid* if and only if $(1,1,...,1) \in R$,
(2) *0-valid* if and only if $(0,0,...,0) \in R$,
(3) *bijunctive* if and only if for all $t_i, t_j, t_k \in R$ then $(t_i \vee t_j) \wedge (t_j \vee t_k) \wedge (t_i \vee t_k) \in R$, i.e., R is closed under 3-ary majority,
(4) *horn* (or weakly positive) if and only if for all $t_i, t_j \in R$ then $t_i \wedge t_j \in R$,
(5) *dual horn* (or weakly negative) if and only if for all $t_i, t_j \in R$ then $t_i \vee t_j \in R$, and
(6) *affine* if and only if for all $t_i, t_j, t_k \in R$ then $t_i \oplus t_j \oplus t_k \in R$, where $\oplus$ is the exclusive disjunction connective (addition modulo 2).

Further we define S having property p if and only if all R_i in S have property p. Also, S is said to be *Shaefer* if and only if it has at least one of the following four properties: bijunctive, horn, dual horn, and affine. If S does not have one of these properties, S is said to be *non-Shaefer.*

As F_M is a CNF(S) formula for $S = \{R_1, R_2, \ldots, R_m\}$, where R_i is the relation corresponding to the truth value assignments of the conjunct $f_i = \bigvee_{j \in J_i} f_{ij}$ of F_M, we can use the above machinery to characterize our problem further. The form of f_i is a disjunction of conjunctions of positive literals (variables). As an immediate consequence, the corresponding relation R_i is always 1-valid, dual horn, and *never* 0-valid. As F_M is dual horn, F_M is Shaefer. Consider $f_1 = (f_{10} \vee f_{12} \vee f_{13})$ of F_M from Example 1. The three truth assignments t_1, t_2, and t_3 shown in Table 1 are in the relation R_1 corresponding to f_1.

	$b_{(0,0)}$	$b_{(0,2)}$	$b_{(1,0)}$	$b_{(1,1)}$	$b_{(1,2)}$	$b_{(2,0)}$	$b_{(2,1)}$	$b_{(2,2)}$	$b_{(3,0)}$	$b_{(3,2)}$
t_1	1	1								
t_2				1	1					
t_3							1	1		

Table 2
Truth assignments in R_1. Blank fields indicate values of 0.

We now check the listed criteria for the remaining properties in turn. As $t_1 \wedge t_2 = (0,0,\ldots,0)$, and R_1 never is 0-valid, we conclude that R_1 is not horn.

Further, as R_1 is dual horn,

$$
\begin{aligned}
&t_4 = t_1 \vee t_2 \in R_1\\
&t_5 = t_1 \vee t_3 \in R_1\\
&t_6 = t_2 \vee t_3 \in R_1.
\end{aligned}
$$

As both

$$
\begin{aligned}
t_4 \wedge t_5 \wedge t_6 = (0, 0, \ldots, 0) \notin R_1\\
t_4 \oplus t_5 \oplus t_6 = (0, 0, \ldots, 0) \notin R_1
\end{aligned}
$$

R_1 is neither bijunctive, nor affine. This allows us to formulate the following theorem:

Theorem 2 *The formula F_M is a* CNF(S) *formula for S such that S is 1-valid and dual horn, but neither 0-valid, bijunctive, horn nor affine in general.*

Theorem 2 lets us immediately conclude that F_M is indeed satisfiable [16] (which is not surprising as suppressing every cell in the table is a trivial solution to our problem), and more importantly, that given a set of cell suppressions A that is a solution to our problem, the problem of determining whether there exists a proper subset B of A such that B still is a solution, is solvable in polynomial time. This is a consequence of Theorem 2 and Theorem 3.2 in Kirousis et al. [12]. Also, the set of satisfying truth assignments form a join semi-lattice.

Unfortunately, our problem is hard. A result given in Theorem 10 in Khanna et. al [11] lets us formulate the following theorem.

Theorem 3 *Problem 1 is a* Min Horn Deletion*-complete NP optimization problem.*

Proof: By the combination of Theorem 1 and Theorem 2, Problem 1 is equivalent finding a minimal satisfying truth assignment of a 1-valid and dual horn, but not 0-valid, horn, affine, nor bijunctive CNF(S) formula. By Theorem 10 in Khanna et. al [11] this problem is an Min Horn Deletion-complete NP optimization problem. □

This means that Problem 1 is not solvable to optimality in polynomial time, and we are forced to apply approximate algorithms as heuristics. We can however determine an upper bound for size of the minimal truth assignment. This bound is n if T is an $m \times n$ table, i.e., T is a table over n attributes, and comes from the observation that suppressing an entire row in the table will make all other rows indistinguishable from this row. A lower bound can be established by inspecting M. The lower bound is $\min_{m_{ij} \in M \wedge m_{ij} \neq \emptyset} |m_{ij}|$ as this represents the absolute minimum of cell suppressions needed to increase α_i for any i.

2.2 Ambiguous Tables and Functional Dependencies

Let T be a table over the finite set of attributes A and the finite set of objects U, as defined above. Given two sets of attributes $X \subseteq A$ and $Y \subseteq A$, we say that there is a *functional dependency* between X and Y in T if and only if whenever two rows agree in the values corresponding to the attributes (columns) in X, they agree in the values corresponding to the attributes in Y. Such a functional dependency is often denoted as $X \rightarrow Y$. Given the knowledge of $X \rightarrow Y$ in T, we can, given the values corresponding to X in a given row, infer the values in the same row corresponding to Y. The field of knowledge discovery in databases (e.g., the AAAI International conferences on Knowledge Discovery and Data Mining) is concerned with finding such functional dependencies. Predictive modeling is an application of discovered functional dependencies. Let $T[X]$ denote, for a given subset X of A, the table resulting in removal of all columns corresponding to the attributes in $A - X$ ($-$ denoting set difference), i.e., $T[X]$ denotes the projection of T onto X. Let, similarly, $r[X]$ denote, for a given row r, the list of values in r for the columns corresponding to attributes X. We further define $r_i[X] = r_j[X]$ if and only if $r_i[\{a_k\}] = r_j[\{a_k\}]$ for all $a_k \in X$. Similarly, we define $r_i[X] \doteq r_j[X]$ if and only if $r_i[\{a_k\}] \doteq r_j[\{a_k\}]$ for all $a_k \in X$. Using these definitions, we can formulate a necessary (and sufficient) requirement for a functional dependency between X and Y in T as follows: for all pairs of rows r_i and r_j in T,

$$r_i[X] = r_j[X] \Rightarrow r_i[Y] = r_j[Y].$$

The counter-positive equivalent of this is

$$r_i[Y] \neq r_j[Y] \Rightarrow r_i[X] \neq r_j[X].$$

From this we see that a counterexample of a functional dependency $X \rightarrow Y$ is the existence of rows r_a and r_b in T such that

$$r_a[Y] \neq r_b[Y] \wedge r_a[X] = r_b[X].$$

As we have that

$$a = b \Rightarrow a \doteq b \not\Rightarrow a \neq b,$$

we cannot guarantee that

$$r_a[Y] \neq r_b[Y] \wedge r_a[X] \doteq r_b[X]$$

is not a counterexample of $X \rightarrow Y$. This disallows proof of the existence of $X \rightarrow Y$ in T. Recalling that a table $T[X]$ is ambiguous if and only if there exist for each rows r_i another row r_j such that $r_i[X] \doteq r_j[X]$, we state the following.

Theorem 4 *Ambiguating $T[X]$ resulting in T' disallows proof of existence of the following family of functional dependencies:*

$$X' \rightarrow S'$$

where $X' \subseteq X$, $S' \subseteq S$, and

$$S = \{s_i | \exists r_j, r_i \in T'(r_i[\{s_i\}] \neq r_j[\{s_i\}] \wedge r_i[X] \doteq r_j[X])\}.$$

Proof: Assume, without loss of generality, that $s_i \in S'$. We have that $r_i[X] \doteq r_j[X] \Rightarrow r_i[X'] \doteq r_j[X']$, and by assumption we know that there exist r_i and r_j, and that they agree on X' but differ in at least s_i, which means that they also differ in S' which contains s_i. This in turn provides us with the potential counterexample $r_i[S'] \neq r_j[S'] \wedge r_i[X'] \doteq r_j[X']$ of $X' \rightarrow S'$, disallowing the proof of this dependency. Q.E.D.

Note that the S is not restricted to the attributes found in T. In particular S can be extended to contain the infinite family $\{s_i | s_i(x_i) \neq s_i(x_j) = 0\}$.

Also worth noting is that the ambiguation of rows i and j by suppressing values of attributes corresponding to the entries in m_{ij} from Section 2.1 disallows proof of any functional dependency $X \rightarrow Y$ where $X \subseteq A$ and $Y \subseteq \{a \in A | r_i[\{a\}] \neq r_j[\{a\}]\}$. This observation can be used to target dependencies $X \rightarrow Y$ for a particular Y by adjusting F_M in a way that forces the agreement in X of at least two rows that disagree in Y. This approach can be used to provide potential counterexamples for any non-constant functional dependency in the table.

2.3 Greedy Heuristics

Two simple heuristics are presented to cope with the minimal suppression problem.

2.3.1 Heuristic A

This heuristic is based on the observation leading to the upper bound on the size of a minimal cell suppression. It simply deletes all entries in the first row. This can be done in linear time in the number of columns in the table.

2.3.2 Heuristic B

Heuristic B is a greedy heuristic for the minimal satisfiability problem of the function F_M. It is based on selecting the variable that occurs the most often in F_M, substituting 1 for every occurrence in F_M, and evaluating as large parts of F_M as possible. This process is repeated until F_M has been reduced to 1, and the list of variables selected during the process is output as the solution. As F_M is satisfied by the assignment of 1 to all variables, the algorithm is guaranteed to return a satisfying set. A straightforward implementation can do so in quadratic time in the length of F_M.

2.4 A review of a previously published method

Øhrn et al. [15] present the following method for ambiguating tables using cell suppression. Associate with each integer i a Boolean variable b_i. Let, as in Section 2.1, T be the $m \times n$ table with the associated discernibility matrix M. Unlike in Section 2.1 however, define

$$F_i = \bigwedge_{j \in \{j' | m_{ij'} \neq \emptyset\}} \bigvee_{k \in m_{ij}} b_k,$$

and let f_i be the Boolean function realized by F_i. The prime implicants of f_i correspond to the minimal set of columns needed to distinguish row i from all other rows [17]. Øhrn et al. observe that suppressing the columns in row i corresponding to a set of variables in F_i such that each prime implicant contributes to this set with at least one variable, will render row i indistinguishable from at least one other row. This observation is key for the two algorithm variations they propose for the cell suppression problem. The problem of finding a minimal set of variables such that each prime implicant contributes with at least one is equivalent to the minimal hitting set problem described in Section 2.1. As this is an NP-hard combinatorial optimization problem, Øhrn et al. apply a greedy approximation algorithm [10] for this selection. The row corresponding to i is then ambiguated by suppressing the cells in the columns corresponding to the selected variables. Øhrn et al. also propose two different measures of anonymity of the table as a whole. Both are based on the per row number of rows that are indistinguishable from it, i.e., based on the value α_i as defined in Section 2.1. The first is based on the minimum over the entire table, the other on the mean. Given one of these measures the two proposed algorithm alternatives iterate over the rows until the measure of anonymity has reached a preselected threshold. The first alternative, C, checks this measures after each row, and modifying M according to the suppressions made, the second, D, updates M and checks for sufficient anonymity after each whole iteration over T.

3 Examples

We implemented heuristics A and B, and the method published by Øhrn et al., designated heuristic C. For heuristic C, the overall anonymity measure for the table was chosen to be the minimum row anonymity over all rows. As examples of heuristic behavior, these were applied to four data tables:

- The data table T_1 in Table 3,
- the data table T_2 found in Table 3, and
- the de-identified data table T_3 found as Table 2 in Øhrn et al., reproduced in Table 3, and
- a data table T_4 of 300 rows and 4 columns constructed from a data set from the University of New Mexico Trauma Center and used among others in a machine learning context [14].

The number of suppressions made by each heuristic on each table is given in Table 3.

a	b	c
1	1	2
3_C	1	$1_{B,C}$
3	$2_{B,C}$	2
1	1	2

Table 3
Example data table T_1, with attributes a, b, and c having four data rows. Subscripts denote suppressions done by the respective heuristics.

B	P	A	C	F
N	N	Y	N	N$_{B,C}$
N	N	Y	N	Y
N	N	Y	Y$_{B,C}$	Y
Y$_{C}$	N	N$_{C}$	N	Y
Y	Y$_{B,C}$	N	N	N$_{B,C}$

Table 4
Example data table T_2, with attributes B, P, A, C and F having five data rows. Subscripts denote suppressions done by the respective heuristics.

Birth year	Gender	Ethnicity	Zip
1964	M	Caucasian	02116$_{B,C,*}$
1964	F	Caucasian	02138
1970	M	Black	02144
1968	F	Asian	02166
1969	F	Black$_{B,C,*}$	02156$_{B,C,*}$
1970	M	Black	02144
1964	F	Caucasian	02138
1969$_{*}$	F	Asian	02116$_{*}$
1968	F	Asian	02166
1964	M	Caucasian	02166$_{*}$

Table 5
The data table T_3 presented in Øhrn et al. with the SSN column removed. Subscripts denote suppressions done by the respective heuristics or suppression reported in the paper by Øhrn et al. by an asterisk.

	A	B	C
T_1	3	2	3
T_2	5	4	6
T_3	4	3	3
T_4	4	112	159

Table 6
The number of suppressions made by each heuristic on the different example data tables.

4 Discussion

Looking at the example results summary in Table 3, we can see that even though no heuristic is best in all instances, heuristic C is never the best alone in our example runs. One weakness of heuristic C, also pointed out by Øhrn et al., is the indeterminacy in choice of the order in which to ambiguate the rows. This is apparent also from that our implementation and the one of Øhrn et al. report different results on T_3. The heuristic C considers each row in isolation with respect to what must be done to ambiguate all other non-ambiguous rows. To see this, consider T_2 and let the order of rows in which heuristic C ambiguates them be 1,2 (0, and 3 are already ambiguous). The algorithm computes the the prime implicants (b_2) and $(b_0 \wedge b_1)$ for row 1. The possible ways of invalidating these is either suppressing columns in $\{2, 0\}$ or $\{2, 1\}$ in row 1. If $\{2, 0\}$ is chosen, column 1 has to be suppressed in row 3, while if we suppress $\{2, 1\}$, this also makes row 3 ambiguous. This second property is shared by the alternative to heuristic C, which we will call D, given by Øhrn et al. This alternative heuristic D views the suppressions in a row in isolation not only from what still has to be suppressed, but also from the already ambiguated rows, making it insensitive to order, but at the cost of computing a solution that will never be better than the solution given by the worst ordering for heuristic C.

Consider F_i as defined in Section 2.4, reviewing the approach of Øhrn et al. The problem of finding a minimal set of variables such that each prime implicant contributes with at least one variable can be stated as the equivalent problem of finding a minimal 0-assignment such that $f_i = 0$. A minimal 0-assignment must at least result in one of the conjuncts in F_i evaluating to 0. Hence a minimal 0-assignment can be obtained by selecting the shortest conjunct in F_i directly, saving first a combinatorial problem of generating $PI(f_i)$ and then the application of a greedy algorithm that only approximates the minimal 0-assignment. Making the above substitution results in a improvement, both in computational complexity and solution quality.

The poor results of heuristics B and C on T_4 seem to suggest that the applicability of both to problems with many rows and few columns needs to be investigated. On the other hand, if the number of columns exceed the number of rows, such as might be the case in genomic data, the performance, in terms of minimal suppressions, of heuristic A should be investigated.

A table that has been ambiguated by minimal cell suppression can be vulnerable to disambiguation attacks if it is known that it has been ambiguated by minimal cell suppression. The reason for this is that minimal cell suppression allows minimal redundant suppressions. This vulnerability can be seen from the following example. Consider the data table containing one column representing attribute a, and two rows, representing objects x_0 and x_1, containing values 0 and 1, respectively. There are two ways of ambiguating this table by minimal suppressions, namely by suppressing either row 0 or row 1. Without loss of generality, assume we suppress row 0. Seeing the suppressed result and knowing that ambiguation by minimal cell suppression was done, we can immediately infer that $a(x_1) \neq a(x_0)$. If we additionally know that the image of $\{x_0, x_1\}$ under a is $\{0, 1\}$ (modeling, e.g, gender), we can infer that $a(x_1) = 1$. By this argument, we see that additional analysis is warranted if ambiguation is applied for hindering induction of functional dependencies if it is known that minimal cell suppression has been applied.

The table ambiguation problem can be extended to the problem of the table k-ambiguation problem, where k denotes the number of distinct rows that a given row should be indistinguishable from. The ambiguation problem discussed here is then the 2-ambiguation problem. The extension of the construction of the formula F_M for this case can be done in a straightforward manner, and the characterization can be done analogously. For the induction of functional dependencies, k-ambiguity means that we can at least find k potential counterexamples for each of the functional dependencies in the family given Theorem 4.

Although functional dependencies are destroyed in the data table by introducing ambiguity, the application of a priori known functional dependency is not foiled. Consider Table 3 in Section 3. Assume that this table describes patients tested for HIV, and that we augment this table with a column describing the test results. Also assume that neither none or all patients tested positive. By Theorem 4, an ambiguated table contains possible counterexamples of the functional dependency between the combination of birth year, gender, ethnicity, zip code, and HIV test results. This, however is not enough if this functional dependency is known, because we still can look up the HIV test results for 8 of the patients in the table assuming the application of heuristics B or C. This becomes serious if we assume the existence of a table representing a functional dependency between the combination of birth year, gender, ethnicity, zip code, and social security number, allowing the explicit identification of the 8 patients.

5 Summary

We have shown that ambiguation of a table provides a potential counterexample for each of an infinite family of functional dependencies. We have formally defined the table ambiguation problem in terms of discernibility using a discernibility matrix. We have shown that the minimal cell suppression problem as defined is equivalent to the problem of finding a minimal satisfying truth assignment of a particular Boolean function and established upper and lower bounds of the size of such an assignment. Further, we show that this function is an instance of a 1-valid, dual horn CNF(S) formula that in general is neither 0-valid, horn, bijunctive, nor affine. This let us prove that the problem is a Min Horn Deletion-complete NP optimization problem. The approach of Øhrn et al. [15] was analyzed and an improvement of the approach in both computational complexity and solution size was presented. The original approach was also compared in four example data tables to two heuristics suggested by our analysis of the problem.

6 Future Work

This exposition is limited to the technical problem of ambiguation of data tables, and the description of the family of functional dependencies that are destroyed in the process. Even though we have a set of potential counterexamples of a functional dependency, we could still produce statistically strong hypotheses about this functional dependency. Future work will be on investigating k-ambiguity for obfuscating functional dependencies. Of interest are those dependencies that allow reconstruction of values that have been suppressed (as shown above in the discussion on the 2×1 table over the attribute a). From the discussion, we also know that the approach to ambiguation described here is unsuitable for the protection against application of known functional dependencies. We will investigate the application of the framework presented here to this problem.

Acknowledgements

This work was supported in part by grant R01 LM 06538 from the National Library of Medicine and the Virginia and John Taplin Award from Harvard/MIT.

References

[1] F. M. Brown. *Boolean Reasoning.* Kluwer Academic Publishers, 1990.

[2] R. Dechter and J. Pearl. Structure identification in relational data. *Artificial Intelligence*, 48:237–270, 1992.

[3] R. S. Dick, E. B. Steen, and D. E. Detmer. *The Computer Based Patient Record: An Essential Technology for Health Care, Revised Edition*. Institute of Medicine, 1997.

[4] M. Fischetti and J. J. Salazar. Models and algorithms for the 2-dimensional cell suppression problem in statistical disclosure control. *Mathematical Programming*, (84):283–312, 1999.

[5] F. H. France and P. N. Gaunt. The need for security–a clinical view. *Int J Biomed Comput*, 35 Suppl:189–94, 1994.

[6] J. K. Gevers. Issues in the accessibility and confidentiality of patient records. *Soc Sci Med*, 17(16):1181–90, 1983.

[7] L. O. Gostin. A proposed national policy on health care workers living with hiv/aids and other blood-borne pathogens. *Jama*, 284(15):1965–70, 2000.

[8] J. Hodge, J. G., L. O. Gostin, and P. D. Jacobson. Legal issues concerning electronic health information: privacy, quality, and liability. *Jama*, 282(15):1466–71, 1999.

[9] A. J. Hundepool and L. C. R. J. Willenborg. Mu- and tau-argus: Software for statistical disclosure control. In *Third International Seminar on Statistical Confidentiality at Bled*, Bled, 1996.

[10] D. S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[11] S. Khanna, M. Sudan, and L. Trevisan. Constraint satisfaction: The approximability of minimization problems. *Electronic Colloquium on Computational Complexity (ECCC)*, 3(064), 1996.

[12] L. M. Kirousis and P. G. Kolaitis. The complexity of minimal satisfiability problems. *Electronic Colloquium on Computational Complexity*, 82, 2000.

[13] P. Kooiman, L. Willenborg, and J. Gouweleeuw. Pram: a method for disclosure limitation of microdata. Rsm-80330, Statistics Netherland, 1997.

[14] L. Ohno-Machado and S. Vinterbo. Effects of case removal in prognostic models for medicine. *Methods of Information in Medicine*, 40(1):32–8, 2001.

[15] A. Øhrn and L. Ohno-Machado. Using boolean reasoning to anonymize databases. *Artif Intell Med*, 15(3):235–54, 1999.

[16] T. J. Shaefer. The complexity of satisfiability problems. In *Proceedings of the 10th ACM Symposium on Theory of Computing*, pages 216–226, 1978.

[17] A. Skowron and C. Rauszer. The discernibility matrices and functions in information systems. volume 11 of *Series D: System Theory, Knowledge Engineering and Problem Solving*, chapter III-2, pages 331–362. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

[18] T.-A. Su and G. Ozsoyoglu. Controlling fd and mvd inferences in multilevel relational database systems. *IEEE Transactions on Knowledge and Data Engineering*, 3(4):474–485, 1991.

[19] L. Sweeney. Weaving technology and policy together to maintain confidentiality. *J Law Med Ethics*, 25(2-3):98–110, 82, 1997.

[20] L. Sweeney. Privacy and medical-records research [letter; comment]. *N Engl J Med*, 338(15):1077; discussion 1077–8, 1998.

[21] L. Sweeney and P. Samarati. Generalizing data to provide anonymity when disclosing information. *Proceedings of the ACM Principles of Database Systems*, 1998.

[22] B. Woodward. The computer-based patient record and confidentiality. *New England Journal of Medicine*, 333(21), 1995.