Statistical Disclosure Control with General Cell Suppressions*

Tsan-sheng Hsu[†]

Institute of Information Science
Academia Sinica, Nankang 11529
Taipei, Taiwan, ROC
tshsu@iis.sinica.edu.tw

Ming-Yang Kao[‡]

Department of Computer Science
Northwestern University
Evanston, IL 60201, USA
kao@cs.northwestern.edu

Abstract

This paper studies statistical database problems for two-dimensional tables whose regular cells, row sums, column sums and table sums may be suppressed. Using graph-theoretical techniques, we give optimal or efficient algorithms for the query system problem, the adversary problem and the minimum complementary suppression problem. The three problems are considered for the case of protecting a single cell or a sum of cells against exact or interval disclosures in a positive or general table.

Previously, graph-theoretical techniques are known for the three problems when the row, column and table sums are not suppressed, and when the data are being protected against exact disclosures. This paper provides two generalized graph-theoretical techniques, which unify previous results, to solve the three statistical database problems without the above constraints.

Keywords: statistical databases, cell suppression, exact disclosure, interval disclosure, graph theory, complementary suppression.

1 Introduction

In many applications [AW89, CÖ82, Cox81, Cox80, Cox95, dCDdSO94, DS83, Den83, IJ94, KGA92], it is a common practice to organize data in a two-dimensional table. In addition to

*An extended abstract of part of an earlier version of this paper appears in [HK97].

[†]Supported in part by NSC Grant 88-2213-E-001-026.

[‡]Supported in part by NSF Grant CCR-9531028.

index	1	2	3	4	5	6	sum
1	2	4	7	3	3	2	21
2	4	3	9	4	2	4	26
3	1	8	6	5	7	3	30
4	8	9	7	6	9	5	44
5	4	4	5	9	8	2	32
sum	19	28	34	27	29	16	153

Table 1: A complete table. The row sums are recorded in the last column, the column sums in the last row, and the table sum at the bottom rightmost cell. The upper and lower bounds of cells are omitted here.

regular cells, a table often includes the marginal cells, i.e., the row, column and table sums. For security reasons, some cells in a published table may be suppressed. However, if the values of enough cells are reported, the exact or bounds of value of a suppressed cell may be determined from the published data. For example, if a cell is the only suppressed cell in a given row, then its value can be deduced by subtracting the total value of the un-suppressed cells in that row from the row sum. Therefore, to achieve desired security, we may need to suppress additional cells so that the values of the original suppressed cells are protected.

This example addresses the problem of whether leaked information can be found in individual cells. More generally, a piece of sensitive information can be defined as a function f of suppressed cells. If the value or bounds of f can be determined by the values and bounds of un-suppressed cells, then the information f is leaked. Larger classes of functions correspond to tighter security requirements. For example, linear combinations of suppressed cells contain more information than sums of suppressed cells. Thus, protecting the former class of functions provides a higher level of data security than protecting the latter. In this paper, we focus on when f is a sum of individual cells [Cox80], though our method can be extended for f to be an arbitrary analytic function[1].

Example I A complete table is illustrated in Table 1 and its published version in Table 2. Let $T(i,j)$ be the cell at the intersection of the ith row and the jth column. The function $T(2,3)+T(2,6)$, a sum combination of suppressed cells in Table 2, can be uniquely evaluated in Table 2 because $T(2,2)$ always equals 3, $T(2,4)$ always equals 4 and $T(2,2)+T(2,3)+T(2,4)+T(2,6)$ always equals 20. Hence the information represented by $T(2,3)+T(2,6)$ is

[1]We say that f is an *analytic function* of (T,X,Y) if it is a power series of the cells z_i such that the convergence radius is ∞ [Kao97b]. For example, $\sin(z_1 \cdot z_2 - 6.5 \cdot z_3) + z_4$ is an analytic function.

index	1	2	3	4	5	6	sum
1		4	7	3	3		21
2	4				2		26
3		8	6	5	7		30
4	8	9		6		5	
5	4	4	5	9		2	32
sum	19	28	34	27		16	153

Table 2: The published version of Table 1. Each blank cell is a suppressed cell. The upper and lower bounds of the cells are omitted here.

not protected.

We investigate the following three statistical database problems.

• *The query system problem.* This detection problem is of concern to the table maker, who has the complete version of a table and the set of cells to be suppressed.
Input: A security requirement, a table with the values and bounds of all cells, and the set of all suppressed cells in the published version of the table.
Output: We wish to decide if any nontrivial information as defined by the given security requirement are protected in the published version of the table.

• *The adversary problem.* This detection problem is of concern to an adversary who only has the published version of a table and desires certain nontrivial information.
Input: A function on values of suppressed cells which has an undisclosed value, and the published version of a table where the values of some cells are suppressed.
Output: We wish to find out certain nontrivial information of the given function.

• *The minimum complementary suppression problem.* This protection problem is also of concern to the table maker.
Input: A security requirement, a table with the values and bounds of all cells, and the set of all sensitive cells in the published version of the table.
Output: We wish to report a set of the smallest number of additional cells, which are called *complementary suppressed cells*, whose suppression protects all the nontrivial information as defined by the security requirement in the published version of the table.

These three problems have been extensively studied for the case where row, column and table sums are not allowed to be suppressed [Gus87, Gus88, HK96, Kao95, Kao97a, Kao97b, KG93]. Some work explored the case where row, column and table sums may be suppressed [AW89, MM96, MM97, MMR91]. It is further argued [MMR91] that it may be best to suppress only additional row and column sums to solve the minimum complementary suppression problem because such data are derived data and thus the table maker is more comfortable with suppressing them.

Using graph-theoretical techniques, this paper studies these three database security problems. Our work subsumes most of the best previous results by obtaining optimal or more efficient algorithms for general problems. Detailed comparisons are given throughout the paper and a summary is given in Section 2.2. Our graph-theoretical techniques are of interest in their own right and may be useful for studying other database security problems.

The rest of this paper is organized as follows. Section 2 presents definitions, summary of results, a useful data structure and general properties. Section 3 discusses the query system problem. Section 4 tackles the adversary problem. Section 5 solves the minimum complementary suppression problem. Section 6 considers additional security requirements.

2 Preliminaries

2.1 Definitions

Let T denote a two-dimensional table with n rows and m columns. The rows and columns are indexed from 1 to n and from 1 to m, respectively. Let $T(i,j)$ denote the value of the cell at the intersection of row i and column j. Such cells are called the *regular cells.* Let *row sum* i refer to the total value of regular cells in the ith row, i.e., $\sum_{k=1}^{m} T(i,k)$; similarly, *column sum* j is the total value of regular cells in the jth column. The *table sum* is the total value of all regular cells in T. The row, column and table sums are the *marginal cells.* For this paper, we assume each cell value is an integer. It can be easily extended to the case when each cell value is a floating-point number with a fixed precision.

A table with suppressions, denoted by (T,X,Y), consists of T as well as a set X of regular cells and a set Y of marginal cells whose values are suppressed in the published version of T. As part of (T,X,Y), for each cell a lower bound and an upper bound are also given to indicate the possible range of the value that cell may have. A cell is *unbounded* if its lower and upper bounds are $-\infty$ and $+\infty$, respectively. A cell that is not unbounded is *bounded.* A cell is *positive* if its lower bound is at least 0. A table with a bounded cell is called a *bounded table.* It is called *positive* if all the cells are positive. If all cells have no bounds (or equivalently with the lower bound $-\infty$ and the upper bound $+\infty$), then it is called *general.*

Let $T_{X,Y}$ denote the published version of (T,X,Y), where the information of whether it is bounded, general or positive, and the bounds of all suppressed cells are also given in addition to the values of all the un-suppressed cells.

A *legal assignment* for (T, X, Y) is an assignment of values to all the cells in T such that (1) the assigned value for an un-suppressed cell equals its value in T, (2) the assigned value for a suppressed cell is within its specified lower and upper bounds, (3) the total assigned value of regular cells in a row (respectively, column) equals its row (respectively, column) sum, and (4) the total assigned value of marginal cells in the table equals the table sum. Note that the cell values in T form a legal assignment for (T, X, Y). A *legal value* for a suppressed cell in (T, X, Y) is the value of this cell in a legal assignment. Let f be a function on suppressed cells. A *legal value* for f is the value of f by assigning the legal values of the suppressed cells in f in some legal assignment.

Let T_f be the value of f in T. We say that f is *protected in* (T, X, Y) *against exact disclosure* if there is a legal assignment for (T, X, Y) such that the legal value of f in this assignment is not T_f. That is, the information represented by f cannot be uniquely determined using published information. Let low_f and $high_f$ be two positive numbers. Given an open interval $(T_f - low_f, T_f + high_f)$, we also say that f is *protected in* (T, X, Y) *against interval disclosure in* $(T_f - low_f, T_f + high_f)$, if there is a legal assignment such that the legal value of f in this assignment is not in the open interval $(T_f - low_f, T_f + high_f)$. Note that the exact disclosure protection is a special case of the interval disclosure. It is usual the case that the interval is specified in a percentage over T_f [Cox95], e.g., an interval of 60% means $low_f = high_f = T_f * 60\%$.

The following fact can be easily verified.

Fact 1

1. *If a function is protected against interval disclosure, then it is also protected against exact disclosure.*

2. *If a function is not protected against exact disclosure, then it is also not protected against any interval disclosure.*

A security requirement is specified by the the set or type of functions as well as whether it is exact or interval disclosure that we desire to protect. This paper focuses on functions that are *sum combinations*, i.e., in the form of $\sum_{i=1}^{k} d_i \cdot z_i$ of (T, X, Y) where for all i, d_i is 0 or 1, and z_i is a cell of (T, X, Y). This paper also studies functions that are *single cells*, i.e., in the form of z where z is a cell of (T, X, Y).

Example II A complete table T is illustrated in Table 1 and its published version in Table 2. Let $T(i, j)$ be the cell at the intersection of the ith row and the jth column. Assume that T is a positive table. The function $f = T(3, 6)$ is protected against exact disclosure in Table 2 as its value can be 1, 2, 3 or 4. However f is not protected against interval disclosure in the interval $(0, 6)$.

2.2 Summary of Results

For the query system problem, Gusfield [Gus88] gives an algorithm for checking whether a single cell in a positive table is protected against interval disclosure by computing a max flow in a flow network which runs in time $O((n + m)^3)$. In his setting, no suppressed marginal cell is allowed. In this paper, we show an algorithm to check the same protection for a sum combination in a positive table with suppressed marginal cells by computing a max flow in a simpler bipartite flow network[2], which takes time $O(\alpha \cdot |X| + \alpha^2 \cdot \sqrt{|X|})$ where $\alpha = \min\{n, m\}$ and X is the set of suppressed cells.

Duarte de Carvalho et al. [dCDdSO94] first shows that in a general table, a single cell is protected against exact disclosure if and only if it is protected against any interval disclosure. They give an $O((n + m)^2)$-time algorithm to check whether a single cell is protected in a general table. Kao [Kao96] solves the same problem in $O(n + m + |X|)$ time, where X is the set of suppressed cells. Malvestuto, Moscarini, and Rafanelli [MMR91] and Malvestuto and Moscarini [MM97] give algorithms to check whether a sum combination in a positive or general table is protected against exact disclosure in time that is linear in the input size. In this paper, we show a linear-time algorithm to check whether a sum combination in a bounded table is protected against exact disclosure. A summary of results on the query system problem is shown in Table 3.

Note that for the adversary problem, the values of the suppressed cells are not part of the input. However, they are inputs for solving the query system problem. There are two versions of the adversary problem. Given $T_{X,Y}$ and a sum combination S of suppressed cells, the *adversary problem of evaluating invariants* is to test whether S is protected against exact disclosure and to compute its exact value if it is not so. The *adversary problem of finding bounds* is to find tight lower and upper bounds of S. It is trivial that the adversary problem of evaluating invariants is a special case of the adversary problem of finding bounds. We are unaware of any result for solving the adversary problem of finding bounds. We show a

[2] The definition of a bipartite flow network is given in Section 3.1.

	interval disclosure	exact disclosure
previous	in a positive table, without suppressed marginal cells, a single cell, max flow time [Gus88];	in a positive or general table, a sum combination, linear time [MM97, MMR91];
ours	in a positive table, a sum combination, bipartite max flow time;	in a bounded table, a sum combination, linear time;

Table 3: A summary of results on the query system problem.

	finding bounds	evaluating invariants
previous	unknown;	in a bounded table without suppressed marginal cells, a set of row/column combinations, linear time [Kao97a];
ours	in a positive table, a sum combination, bipartite max flow time;	in a bounded table, a set of row/column combinations, linear time;

Table 4: A summary of results on the adversary problem.

polynomial-time algorithm to solve this problem for a sum combination in a positive table by finding a max flow in a bipartite flow network. For the adversary problem of evaluating invariants, Kao [Kao97a] shows a linear-time algorithm when S is a sum of cells in the same row or column of a bounded table without suppressed marginal cells. We extend his result by allowing suppressed marginal cells. A summary of results on the adversary problem is shown in Table 4.

For the minimum complementary suppression problem, results are known only for protecting single suppressed cells against exact disclosure. Malvestuto and Moscarini [MM97] and Malvestuto, Moscarini, and Rafanelli [MMR91] consider the case when the additional complementary cells are all marginal cells. Their algorithms run in linear time on a positive table for protecting all suppressed cells against exact disclosure. We show a linear-time algorithm that protects a selected subset of suppressed cells in a general table. Our algorithm also solves the problem of deciding whether it is possible to protect the given suppressed cells by only adding complementary marginal cells.

Gusfield [Gus87] solves the minimum complementary suppression problem when the table is *strictly positive*, i.e., all cell values are greater than 0, and the additional complemen-

	suppress only additional marginal cells	may suppress any additional cells
previous	in a positive table, protect all suppressed cells, linear time [MM97];	(1) in a strictly positive table, without suppressed marginal cells, protect all suppressed cells, linear time [Gus87]; (2) in a general table, protect only one cell, min-cost flow time [Cox95];
ours	in a general table, protect selected suppressed cells, deal with a special case, linear time;	in a general table, protect selected suppressed cells, deal with a special case, linear time;

Table 5: A summary of results on the minimum complementary suppression problem.

tary cells are all regular cells. He gives a linear-time algorithm to protect all suppressed cells when there is no suppressed marginal cell. Cox [Cox80] gives an algorithm to protect only one suppressed cell in a general table without suppressed marginal cells. His algorithm needs to find a minimum-cost flow in a constructed flow network, which takes $O((|X|\cdot \log(n+m))\cdot(|X|+(n+m)\cdot \log(n+m)))$ time [AMO93], where X is the set of originally suppressed cells. Duarte de Carvalho et al. [dCDdSO94] assigns positive weights to all potential complementary cells. They show a heuristic algorithm based on branch-and-bound to find a set of complementary cells in a general table with the minimum sum of weights. This paper shows a linear-time algorithm to find a minimum set of complementary cells to protect a subset of suppressed cells in a general table that may has suppressed marginal cells. We also show an algorithm to deal with the special case that the table sum cannot be a complementary cell. A summary of results on the complementary suppression problem is shown in Table 5.

2.3 Extended Tables

Previously, graph-theoretical techniques [Gus87, Gus88, HK96, Kao95, Kao97a, Kao97b, KG93] are developed for solving security problems on a statistical table without suppressed marginal cells. Here we provide the following transformation from a table with suppressed marginal cells to an equivalent table without suppressed marginal cells. Through the transformation and additional graph-theoretical properties, we can apply previous techniques to

index	1	2	3	4	5	6	7	sum
1	2	4	7	3	3	2	−21	0
2	4	3	9	4	2	4	−26	0
3	1	8	6	5	7	3	−30	0
4	8	9	7	6	9	5	−44	0
5	4	4	5	9	8	2	−32	0
6	−19	−28	−34	−27	−29	−16	153	0
sum	0	0	0	0	0	0	0	0

Table 6: The extended table of Table 1.

index	1	2	3	4	5	6	7	sum
1		4	7	3	3		−21	0
2	4				2		−26	0
3		8	6	5	7		−30	0
4	8	9		6		5		0
5	4	4	5	9		2	−32	0
6	−19	−28	−34	−27		−16	153	0
sum	0	0	0	0	0	0	0	0

Table 7: The extended table with suppressions of Table 2.

solve database security problems where the input table has suppressed marginal cells.

The *extended table* T' for T consists of $n+1$ rows and $m+1$ columns, where

$$T'(i,j) = \begin{cases} T(i,j) & 1 \leq i \leq n \;\&\; 1 \leq j \leq m; \\ -\sum_{k=1}^{n} T(k,j) & i = n+1 \;\&\; 1 \leq j \leq m; \\ -\sum_{k=1}^{m} T(i,k) & 1 \leq i \leq n \;\&\; j = m+1; \\ T(n+1, m+1) & i = n+1 \;\&\; j = m+1. \end{cases}$$

The row, column and table sums in T' are always 0 regardless the cell values in T.

Example III Table 6 gives the extended table of Table 1. Table 7 is the extended table with suppressions of Table 2.

Let $\min T'$ be the minimum value of all cells in an extended table T'. Given a table T and its extended table T', the *positive extended table* T'' is

$$T''(i,j) = T'(i,j) - \min T' \quad 1 \leq i \leq n+1 \;\&\; 1 \leq j \leq m+1.$$

index	1	2	3	4	5	6	7	sum
1	46	48	51	47	47	46	23	308
2	48	47	53	48	46	46	18	308
3	45	52	48	49	51	47	14	308
4	52	53	51	50	53	49	0	308
5	48	48	49	53	52	46	12	308
6	25	16	10	14	15	28	197	308
sum	264	264	264	264	264	264	264	1848

Table 8: The positive extended table of Table 1.

index	1	2	3	4	5	6	7	sum
1		48	51	47	47		23	308
2	48				46		18	308
3		52	48	49	51		14	308
4	52	53		50		49		308
5	48	48	49	53		46	12	308
6	25	16	10	14		28	197	308
sum	264	264	264	264	264	264	264	1848

Table 9: The positive extended table with suppressions of Table 2.

The row, column and table sums in T' are then computed. It is a fact that the value of each cell in T'' is positive.

Let $\xi(Y)$ denote the set of cells in T' that correspond to the suppressed marginal cells in T. For (T, X, Y) (respectively, $T_{X,Y}$), the corresponding *extended table with suppressions* (respectively, *extended published table*) is $(T', X \cup \xi(Y), \emptyset)$ (respectively, $T'_{X\cup\xi(Y),\emptyset}$). The bounds for each suppressed cell in T' are the same as those of its corresponding cell z in T if z is a regular cell or the table sum. If z is a row or column sum in T, then the lower (respectively, upper) bound of z in T' equals the upper (respectively, lower) bound multiplied by -1. Note that since z is a cell in T, z cannot be corresponding to a row, column or table sum in T'. We similarly define a *positive extended table with suppressions.*

Let $T'_{X\cup\xi(Y),\emptyset}$ be an extended published table with finite lower bounds on all suppressed cells. Let $\min T'_{X\cup\xi(Y),\emptyset}$ be the minimum value of the values of all un-suppressed cells and the lower bounds of all suppressed cells. The *positive extended published table* is

$$T''_{X\cup\xi(Y),\emptyset}(i,j) = T'_{X\cup\xi(Y),\emptyset}(i,j) - \min T'_{X\cup\xi(Y),\emptyset} \quad \text{for all un-suppressed cells } (i,j).$$

The bounds for each suppressed cell in $T''_{X\cup\xi(Y),\emptyset}$ are similarly defined as the ones for positive extended tables.

Example IV Table 8 gives the positive extended table of Table 1. Note that the extended table of Table 1 is shown in Table 6 whose minimum cell value is -44. Table 9 illustrates the positive extended table with suppressions of Table 2.

A function $f'(z'_1, \ldots, z'_k)$ over suppressed cells in $(T', X \cup \xi(Y), \emptyset)$ is *equivalent* to a function $f(z_1, \ldots, z_k)$ over the suppressed cells in (T, X, Y) if (1) $z_i = z'_i$ where z_i is a regular cell or the table sum in T and (2) $z_i = -z'_i$ where z_i is a row or column sum in T. A function $f''(z''_1, \ldots, z''_k)$ over regular suppressed cells in $(T'', X \cup \xi(Y), \emptyset)$ is *equivalent* to $f'(z'_1, \ldots, z'_k)$ over regular suppressed cells in $(T', X \cup \xi(Y), \emptyset)$ if $z''_i = z'_i - \min T'$. The function f'' is also *equivalent* to f.

Example V Let T be the published table in Table 2. Let T' be the extended table of T as shown in Table 7. Then $\min T' = -44$. Let $T(i, \text{sum})$ and $T(\text{sum}, j)$ denote the ith column sum and the jth row sum of T, respectively. Let $T(\text{sum}, \text{sum})$ denote the table sum of T. The function $f = T(1,1) + T(2,4) + T(4, \text{sum})$ is a sum combination over suppressed cells in T. The function $f' = T'(1,1) + T'(2,4) - T'(4,7)$ is equivalent to f. The function $f'' = T''(1,1) + T''(2,4) - T''(4,7) + \min T'$ is equivalent to both f' and f.

Intuitively, (T, X, Y), $(T', X \cup \xi(Y), \emptyset)$ and $(T'', X \cup \xi(Y), \emptyset)$ contain essentially the same information, which is characterized formally in the next lemma.

Lemma 2

1. *Assume that T is a general table. A function over suppressed cells of (T, X, Y) is protected in (T, X, Y) if and only if its equivalent function is protected in $(T', X \cup \xi(Y), \emptyset)$.*

2. *Assume that T is a positive table. A function over suppressed cells of (T, X, Y) is protected in (T, X, Y) if and only if its equivalent function is protected in $(T'', X \cup \xi(Y), \emptyset)$.*

Proof. Straightforward. □

2.4 Suppressed Graphs

A *mixed* graph is one where each edge (u, v) may be directed from u to v, directed from v to u, or undirected. The *suppressed graph* $G_{T,X}$ of a table T with regular suppressed cells X is a bipartite mixed graph with the following two sets A and B of vertices and the following set E of edges. The vertices in $A = \{r_1, \ldots, r_n\}$ are called the *row* vertices, where r_i corresponds to row i; the vertices in $B = \{c_1, \ldots, c_m\}$ are the *column* vertices, where c_j corresponds to column j. For each $T(i, j) \in X$, there is an edge $e = (r_i, c_j) \in E$. If the value $T(i, j)$ equals its lower bound, then e points from r_i to c_j, denoted by $< r_i, c_j >$. We say that $< r_i, c_j >$ is an *incoming* edge for the vertex c_j and an *outgoing* edge for the vertex r_i. If the value $T(i, j)$ equals its upper bound, then e points from c_i to r_j, denoted by $< c_j, r_i >$. Otherwise, e is undirected and is denoted by (r_i, c_j) or (c_j, r_i). Note that if T is general, then $G_{T,X}$ is undirected. We define the *undirected version* of $G_{T,X}$, denoted by $u(G)_{T,X}$, by replacing each directed edge of $G_{T,X}$ with an undirected edge with the same endpoints. For convenience, an edge is identified with its corresponding cell, and a vertex with its corresponding row or column.

Example VI Figure 1 shows the suppressed graph of Table 2. We assume that all the suppressed marginal cells are unbounded and that the lower and upper bounds of every suppressed regular cell are 1 and 9, respectively. Thus Figure 1 shows a mixed graph.

Figure 2 shows the suppressed graph of Table 2 if there is no bound on the cells. Since the table is general, the suppressed graph is undirected.

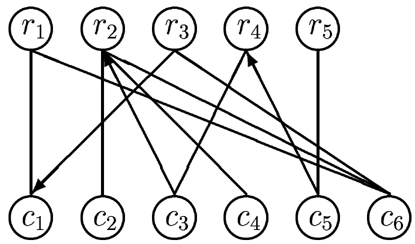

Figure 1: The suppressed graph of Table 2. We assume that all the suppressed marginal cells are unbounded and that the lower and upper bounds of every suppressed regular cell are 1 and 9, respectively.

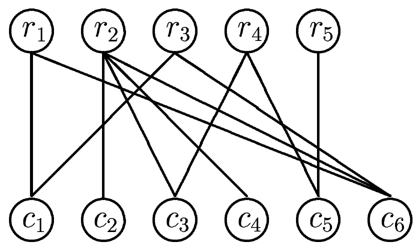

Figure 2: The suppressed graph of Table 2 if there is no bounds on the cells. Since the table is general, the suppressed graph is undirected.

Throughout the paper, (T, X, Y) is an input to the query system problem and the minimum complementary suppression problem, while $T_{X,Y}$ is an input to the adversary problem. However, since $G_{T,X}$, $G_{T',X\cup\xi(Y)}$ and $G_{T'',X\cup\xi(Y)}$ usually have smaller representations than (T, X, Y) and $T_{X,Y}$, we always convert the tables into their suppressed graphs, where appropriate, to obtain sharper complexity bounds.

2.5 Properties

In a mixed graph, a *cycle* of length k is a sequence of vertices $v_0, v_1, \ldots, v_{k-1}$ such that for each i, either $(v_i, v_{i+1 \bmod k})$ or $<v_i, v_{i+1 \bmod k}>$ is an edge of the graph, which is called the *ith edge of the cycle*. A *simple cycle* is one such that $v_i \neq v_j$ unless $i = j$. It is well-known that each cycle in a bipartite graph has an even number of edges.

Lemma 3 *[Gus87] Given a general table $(T, X, \emptyset)$, if an edge of $u(G)_{T,X}$ is not contained in any cycle, then it is not protected against exact disclosure.*

Remark: Lemma 3 is true regardless the lower and upper bounds of each cell in T.

Given $G_{T,X}$, its undirected version $u(G)_{T,X}$, an edge e of $u(G)_{T,X}$ and a cycle C containing e, an edge $e' \in C$ is the *same parity* with e if e and e' are both either odd numbered or even numbered in C. Let e^* be an edge of $u(G)_{T,X}$ and let $C(e^*)$ be a cycle of $u(G)_{T,X}$ containing e^*. Let $C_1(e^*)$ (respectively, $C_2(e^*)$) be the set of edges of $C(e^*)$ that are the same (respectively, different) parity with e^*. Let U_{e^*} (respectively, L_{e^*}) be the upper (respectively, lower) bound of the cell corresponding to e^*. Let T_{e^*} be the value of e^* in T.

- $C(e^*)_{inc} = \min\{\min_{e \in C_1(e^*)}\{T_{e^*} - L_{e^*}\}, \min_{e \in C_2(e^*)}\{U_{e^*} - T_{e^*}\}\}$ and $e^*(C)_{max} = T_{e^*} + C(e^*)_{inc}$.
- $C(e^*)_{dec} = \min\{\min_{e \in C_1(e^*)}\{U_{e^*} - T_{e^*}\}, \min_{e \in C_2(e^*)}\{T_{e^*} - L_{e^*}\}\}$ and $e^*(C)_{min} = T_{e^*} - C(e^*)_{dec}$.

Lemma 4

1. *There is a legal assignment for $(T, X, \emptyset)$ such that the value of e^* in the assignment is $e^*(C)_{max}$.*
2. *There is a legal assignment for $(T, X, \emptyset)$ such that the value of e^* in the assignment is $e^*(C)_{min}$.*
3. *For each value w in the close interval $[e^*(C)_{min}, e^*(C)_{max}]$, w is a legal value for the cell corresponding to e^*.*

Proof. Note that $e^* \in C_2(e^*)$. Let A be the legal assignment for $(T, X, \emptyset)$ whose assigned values for each suppressed cell equals its that of its corresponding cell in T. Let w be a positive integer. Let A' be the legal assignment of $(T, X, \emptyset)$ by applying the following modifications to A:

- The value of a cell corresponding to an edge in $C_1(e^*)$ is decreased by w.
- The value of a cell corresponding to an edge in $C_2(e^*)$ is increased by w.
- The value of a cell not corresponding to an edge in $C(e^*)$ is unchanged.

It is trivial to see that if the value of each cell in A' is within the bounds of this cell, then A' is also a legal assignment for $(T, X, \emptyset)$. It is also trivial to see that if $-C(e^*)_{dec} \le w \le C(e^*)_{inc}$, then the value of each cell in A' is within the bounds of this cell. Hence this lemma follows. □

Remark: For the extremal case in Lemma 4(3), the close interval can be chosen as $[-\infty, +\infty]$.

Lemma 5 *Let A be a legal assignment for $(T, X, \emptyset)$. Given a suppressed cell $e \in X$, let A_e be the value of e in A. Let T_e be the value of e in T. For each value w in the close interval $[\min\{A_e, T_e\}, \max\{A_e, T_e\}]$, there is an legal assignment for $(T, X, \emptyset)$ whose assigned value for e is w'.*

Proof. In order for A to be a legal assignment, if $A_e > T_e$ (respectively, $A_e < T_e$), then there is a a cell $e_1 \in X$ such that e and e_1 are in the same row or column and $A_{e_1} < T_{e_1}$ (respectively, $A_{e_1} > T_{e_1}$). This lemma is trivially true when $A_e = T_e$. Assume without loss of generality, $A_e > T_e$. Hence there is a cell $e_1 \in X$ such that $A_{e_1} < T_{e_2}$. Following the same reasoning, there is a cell $e_2 \in X$ such that such that e_1 and e_2 are in the same row or column and $A_{e_2} > T_{e_2}$. Thus we can find $e_1, e_2, \ldots, e_k = e$ all satisfying the above. This corresponds to a cycle in the suppressed graph. Hence k is even. Let $\delta = \min_{1 \le i \le k/2}\{T_{e_{2 \cdot i - 1}} - A_{e_{2 \cdot i-1}}, A_{e_{2 \cdot i}} - T_{e_{2 \cdot i}}\}$. Note that $\delta > 0$. We perform the following modifications to the original legal assignment to obtain a new assignment:

- for each i such that $1 \le i \le k/2$, replace the assigned value for $e_{2 \cdot i - 1}$ by $A_{e_{2 \cdot i+1}} + \delta$;
- for each i such that $1 \le i \le k/2$, replace the assigned value for $e_{2 \cdot i}$ by $A_{e_{2 \cdot i}} - \delta$ in A.

It is clear the resulting set of assigned values still forms a legal assignment. Thus for each value w' in $[A_e - \delta, A_e]$, there is a legal assignment such that the value of e in this assignment is w'. By applying the same argument, we can find $\delta_1 > 0$ such that for each value w'' in $[A_e - \delta - \delta_1, A_e - \delta]$, there is a legal assignment such that the value of e in this assignment is w''. By repeatedly applying this argument, this lemma follows. □

Theorem 6 *Let e be an edge of $u(G)_{T,X}$. There exists a close interval $[e_{min}, e_{max}]$ such that a value w is a legal value of e if and only if w is in $[e_{min}, e_{max}]$.*

Proof. The *only if* part follows from Lemma 5. Let $\mathcal{C}$ be the set of cycles containing the edge e. Let $e_{max} = \max_{C \in \mathcal{C}} e(C)_{max}$ and let $e_{min} = \min_{C \in \mathcal{C}} e(C)_{min}$. Hence the *if* part follows from Lemma 4. □

3 The Query System Problem

Recall that given (T, X, Y) and a set S of sum combinations of (T, X, Y), the *query system problem* is to find the sum combinations in S that are not protected against exact or interval disclosure. We fist give a result for checking a sum combination of suppressed cells in a positive table against arbitrary interval disclosure. We then give a faster algorithm for checking a set of sum combinations of suppressed cells in a bounded table against exact disclosure.

3.1 Protected Against Interval Disclosure

This section assumes positive tables. The *row residual* for a row whose row sum is not suppressed equals the the total value of the suppressed regular cells in that row. We similarly define *column residuals*. Note that all the row and column residuals are defined in the positive extended $T''_{X\cup\xi(Y),\emptyset}$. Let $\pi_{T,X}$ be the sum of the row residuals in $(T'', X \cup \xi(Y), \emptyset)$. The *flow graph of* $u(G)_{T'',X\cup\xi(Y)}$ *with lower bounds*, denoted as $F_{T'',X\cup\xi(Y)}$, is a directed graph defined as follows:

- The vertex set is the vertex set of $u(G)_{T'',X\cup\xi(Y)}$ plus two additional vertices s and t.
- The edge set is the union of $\{<s, r>|\ r$ is a row vertex$\}$, $\{<c, t>|\ c$ is a column vertex$\}$ and $\{<r, c>|\ (r, c)$ is an edge of $u(G)_{T'',X\cup\xi(Y)}\}$.

Note that s has no incoming edges and t has no outgoing edges. Note that $u(G)_{T'',X\cup\xi(Y)}$ is *bipartite network* [AMO93] as each of the edges, excluding those starts s or ends t, has an endpoint being row vertex and has the other endpoint being a column vertex. In a flow graph, a *lower bound* and a *capacity* are assigned to each of its edges as follows:

- Given an edge in the form of $<s, r_i>$, let its lower bound and capacity both be the row residual of row r_i.
- Given an edge in the form of $<c_j, t>$, let its lower bound and capacity both be the column residual of column c_j.
- Given an edge in the form of $<r_i, c_j>$, let its lower bound (respectively, capacity) be the lower (respectively, upper) bound of its corresponding cell in T.

Example VII The flow graph for the positive extended table with suppression in Table 9 is shown in Figure 3. The capacities of the edges $<s, r_1>, <s, r_2>, \ldots, <s, r_6>$ are 92, 194, 92, 104, 52 and 15, respectively. Note that these values equal the row residuals of $r_1, \ldots,$ r_6, respectively. The capacities of the edges $<c_1, t>, <c_2, t>, \ldots, <c_7, t>$ are 91, 47, 104,

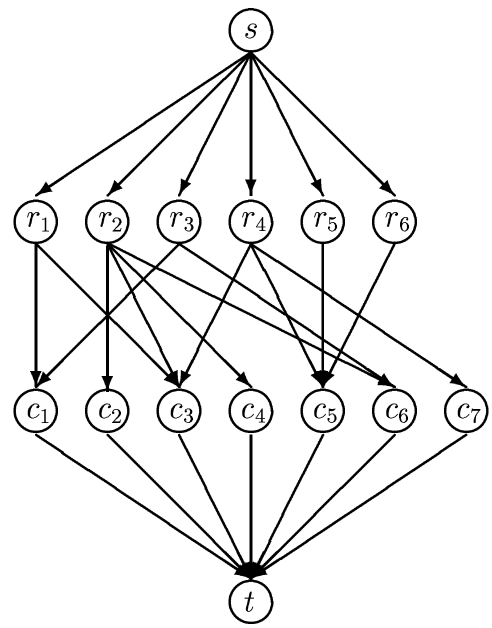

Figure 3: The flow graph of Table 9.

48, 120, 139 and 0, respectively. Note that these values equal the column residuals of c_1, …, c_7, respectively. The lower bounds of these edges equal their capacities. For the rest of the edges, their lower bound and capacity are the same with the lower and upper bounds of their corresponding cells in Table 9.

A *flow* in the flow graph with lower bounds is an assignment of non-negative weights to the edges such that

- For each edge, the weight is at most its capacity and at least its lower bound.
- For each vertex that is neither s nor t, the sum of the weights of its incoming edges equals that of its outgoing edges.
- The sum of the weights of the outgoing edges of s equals that of its incoming edges of t. This value is called the *value* of this flow.

A *max flow* is a flow with the maximum value among all flows.

Lemma 7 *A max flow of $F_{T'',X\cup\xi(Y)}$ whose value to be equal to $\pi_{T'',X\cup\xi(Y)}$ corresponds to a legal assignment of $(T'', X \cup \xi(Y), \emptyset)$ and vice versa.*

Proof. We assign the value of a cell equals the assigned weight of the corresponding edge in the found max flow. It corresponds to a legal assignment of $(T'', X \cup \xi(Y), \emptyset)$ since each assigned value is within the lower and upper bound of each suppressed cell. Further, the constraints imposed on the flow forces the values in a row (respectively, column) to add up to be equal to its corresponding row (respectively, column) sum. Using a similar argument, it is trivially to see a legal assignment corresponds to a max flow with the value $\pi_{T'', X \cup \xi(Y)}$.
□

Given a sum combination S, $F_{T'', X \cup \xi(Y)}$ and a value k, the *k,S-flow graph* $F_{T'', X \cup \xi(Y)}(k, S)$ is a flow graph obtained by revising $F_{T'', X \cup \xi(Y)}$ as follows:

- We add two new vertices r and c.
- We add two new edges $<s, r>$ and $<c, t>$ whose lower bound and capacities are both k.
- For each edge $<r_i, c_j> \in S$, we remove the edge $<r_i, c_j>$ and then add the two edges $<r_i, c>$ and $<r, c_j>$. The capacity and lower bound of $<r_i, c>$ equal the upper and lower bounds of $<r_i, c_j>$, respectively. The capacity and lower bound of $<r, c_j>$ also equal the upper and lower bounds of $<r_i, c_j>$, respectively. For any fixed i, if there are multiple edges of the form $<r_i, c>$ or $<r, c_i>$ added, then we keep only one copy whose lower bound (respectively, capacity) equals the sum of the lower bounds (respectively, capacities) of all of the copies.

Example VIII Let $(T', X', \emptyset)$ be Table 9. Let $k = 101$ and $S = T'(2,3) + T'(2,4)$. Figure 4 shows the k, S-flow graph of Table 9. The lower bound and the capacity of the edges $<s, r>$ and $< c, t >$ are both 0 and k, respectively. The lower bound (respectively, capacity) of the edge $<r_2, c>$ equals the sum of the lower bounds (respectively, capacities) of the edges $<r_2, c_3>$ and $<r_2, c_4>$ in Figure 3. The lower bounds (respectively, capacities) of the edges $<r, c_3>$ and $<r, c_4>$ equal the lower bounds (respectively, capacities) of the edges $<r_2, c_3>$ and $<r_2, c_4>$, respectively.

Lemma 8 *There exists a max flow of $F_{T'', X \cup \xi(Y)}(k, S)$ whose flow value is $k + \pi_{T,X}$ if and only if there is a legal assignment of (T, X, Y) such that the legal value of S in the assignment is k.*

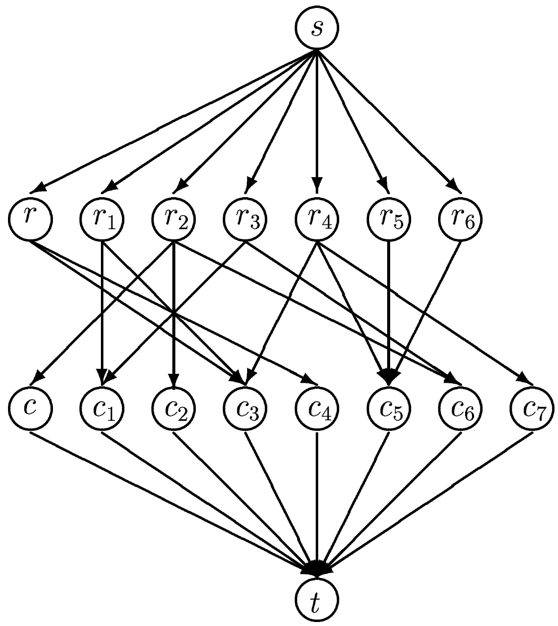

Figure 4: Let $(T', X \cup \xi(Y), \emptyset)$ be Table 9. Let $k = 101$ and $S = T'(2,3) + T'(2,4)$. Illustrating the k, S-flow graph for $(T', X \cup \xi(Y), \emptyset)$.

Proof. We first use the positive extended table technique if $Y \neq \emptyset$. Hence we can assume $Y = \emptyset$ in the discussion. First we see that a max flow f' in $F_{T'', X \cup \xi(Y)}(k, S)$ with the flow value $k + \pi_{T,X}$ corresponds to a max flow f in $F_{T'', X \cup \xi(Y)}$ with the flow value $\pi_{T,X}$, and vice versa. Hence this lemma follows from Lemma 7. □

Theorem 9 *Given an open interval (w_1, w_2) and a sum combination S of suppressed cells in a positive table (T, X, Y), we can check whether it is protected against interval disclosure in the open interval (w_1, w_2) in $O(\alpha \cdot (|X| + |Y|) + \alpha^2 \cdot \sqrt{|X| + |Y|})$ time, where $\alpha = \min\{n, m\}$.*

Proof. Let T_S be the value of S in T. Note that $T_S \in (w_1, w_2)$. By Theorem 6, if there is a legal assignment whose legal value for S is w, then for each value w' in $[\min\{w, T_S\}, \max\{w, T_S\}]$, there is a legal assignment whose legal value for S is w'.

Hence we first use Lemma 8 to check whether there is a legal assignment such the legal value of S in it equals w_1. If there is a legal assignment A such that the legal value of S in A is w_1 or w_2, then A can be found using Lemma 8. If A exists, then S is protected against interval disclosure in the open interval (w_1, w_2). Note that by our definition, the legal value of S in T is in the open interval (w_1, w_2). If there is no such A, then by Theorem 6 S is not protected against interval disclosure in (w_1, w_2).

The time to find a max flow with lower bounds in a bipartite network with $|E|$ edges and $|V_1| + |V_2|$ vertices is $O(\min\{|V_1|, |V_2|\}\cdot|E| + \min\{|V_1|, |V_2|\}^2\cdot\sqrt{|E|})$ time [AMO93], where V_1 and V_2 are the two disjoint sets of vertices in the network. Since the constructed $F_{T'',X\cup\xi(Y)}(k, S)$ has $|X| + |Y|$ edges and $|n| + |m| + 4$ vertices, this theorem follows. □

3.2 Protected Against Exact Disclosure

We now show a fast algorithm to decide whether a set of sum combinations in a bounded table is protected against exact disclosure.

There is a linear-time algorithm to test whether a sum combination of suppressed cells in a positive table is protected against exact disclosure [MM97, MMR91]. There is also a linear-time algorithm to test whether a linear combination[3] (and therefor a sum combination) of cells in a general table is protected against exact disclosure when marginal cells are not allowed to be suppressed [KG93].

Using extended tables with suppressions, we have the following theorem.

Theorem 10 *Given a sum combination S and $G_{(T',X\cup\xi(Y)),\emptyset)}$, the query system problem of testing where the sum combination S is protected against exact disclosure can be solved in optimal $O(n + m + |X|)$ time no matter (T, X, Y) is a positive or general table.*

Proof. Using the notion of an extended table with suppressions, we can obtain an equivalent table without suppressed marginal cells. Hence we can apply the techniques in [KG93] to solve the query system problem when the table maker wants to know whether sensitive information defined by linear combinations of suppressed cells is protected. □

Example IX Table 7 gives the extended table of Table 2. Figure 5 gives its suppressed graph where all the suppressed marginal cells are unbounded, and the lower and upper bounds of every suppressed regular cell are 1 and 9, respectively. Since there is no suppressed marginal cell, we can apply previous graph-theoretical techniques in [KG93] to solve the query system problem.

Remark: Note that it is possible to speed up the testing whether several linear combinations are protected against exact disclosure to be linear in the total number of terms in the given set of linear combinations and $n+m+|X|$ using advanced data structures and graph-theoretical techniques as the ones used in [KG93]. Details are omitted.

[3] We say that a function is a *linear combination* of (T, X, Y) if it is $\sum_{i=1}^{k} d_i\cdot z_i$ for some constants d_i, where z_i is a suppressed cell of (T, X, Y).

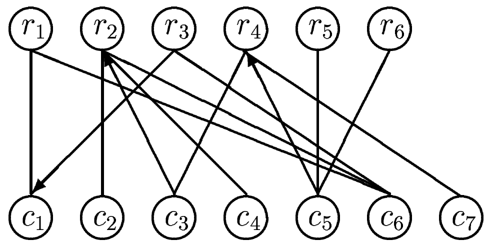

Figure 5: The suppressed graph of Table 7. We assume that all the suppressed marginal cells are unbounded and that the lower and upper bounds of every suppressed regular cell are 1 and 9, respectively.

4 The Adversary Problem

This section addresses the adversary problem. Hence the input is the publish version of the suppressed table $T_{X,Y}$. Given a sum combination, we first show an algorithm to solve the adversary problem of finding bounds in a positive table. We then show a faster algorithm, in a positive or general table, to solve the adversary problem of evaluating invariants.

4.1 Finding Bounds

We assume suppressed tables in this section are positive. If there are suppressed marginal cells, i.e., $Y \neq \emptyset$, then we can apply the positive extended table technique to convert $T_{X,Y}$ to an equivalent positive table $T''_{X \cup \xi(Y),\emptyset}$. Hence we can assume without loss of generality that $Y = \emptyset$.

From $T_{X,\emptyset}$, we can construct its flow graph $F_{T,X}$. By Lemma 7, a max flow in $F_{T,X}$ corresponds to a legal assignment for $T_{X,\emptyset}$. Let S be a given sum combination.

Lemma 11 *The upper bound for the legal value of S in $T_{X,\emptyset}$ can be obtained by finding a max flow in $F_{T,X}(+\infty, S)$.*

Proof. Note that the capacities of the edges $<s,r>$ and $<c,t>$ are both $+\infty$ in $F_{T,X}(+\infty, S)$. We find a max flow in $F_{T,X}(+\infty, S)$. The lower bounds and capacities of the edges ensure any flow in $F_{T,X}(+\infty, S)$ corresponds to a flow in $F_{T,X}$, and vice versa. By Lemma 7, a flow in $F_{T,X}(+\infty, S)$ corresponds to a legal assignment for $T_{X,\emptyset}$, and vice versa. Let T_S be the legal value for S in T. Note that the flow value in $F_{T,X}(+\infty, S)$ is at least $T_S + \pi_{T,X}$. Further, any flow with the value $q + \pi_{T,X}$ corresponds to a legal assignment such that the legal value of S in this assignment is q, and vice versa. Hence a max flow in $F_{T,X}(+\infty, S)$ finds the largest possible legal value for S □

Lemma 12 *The lower bound for the legal value of S in $T_{X,\emptyset}$ can be obtained by finding a max flow in $F_{T,X}(0,S)$.*

Proof. Note that the capacities of the edges $<s,r>$ and $<c,t>$ are both 0 in $F_{T,X}(0,S)$. A flow f in $F_{T,X}$ with the flow value $\pi_{T,X}$ corresponds to a flow in $F_{T,X}(0,S)$ with the flow value $\pi_{T,X} - q$ where q is the legal value of S in the legal assignment corresponding to f, and vice versa. Let w be the value of a max flow in $F_{T,X}(0,S)$. Thus the lower bound for the legal value of S is $\pi_{T,X} - w$. □

Theorem 13 *The lower and upper bounds of a sum combination in a positive table (T,X,Y) can be found in $O(\alpha\cdot(|X|+|Y|) + \alpha^2\cdot\sqrt{|X|+|Y|})$ time, where $\alpha = \min\{n,m\}$.*

Proof. If $Y \neq \emptyset$, then we first apply the positive extended table technique. Hence we assume without loss of generality $Y = \emptyset$. The theorem follows from Lemmas 11 and 12. □

4.2 Evaluating Invariants

This section assumes that the suppressed table is general. This section focuses on the case of evaluating row and column invariants.

A *cut* in a graph is a minimal subset of edges whose removal separates a connected component into two new connected components. Let $G^* = u(G)_{(T',X\cup\xi(Y),\emptyset)}$. Let W be a cut in G^*. Let K_1 and K_2 be the two connected components of G^* created by removing W. Let U_i be the set of endpoints of W in K_i. W is called a *bipartite cut* if U_1 consists of only row vertices and U_2 consists of only column vertices. Note that a cut edge forms a bipartite cut. Let $\mathcal{R}(W)$ (respectively, $\mathcal{C}(W)$) be the set of endpoints of the edges in W that are row (respectively, column) vertices. W is a *γ-cut* if $|\mathcal{R}(W)| = 1$ or $|\mathcal{C}(W)| = 1$. Note that a γ-cut is a bipartite cut.

Lemma 14 *Let W be a bipartite cut in $G_{(T,X,\emptyset)}$. Let K_1 be the connected component in $G_{(T,X,\emptyset)} - W$ that contains $\mathcal{C}(W)$. If W, K_1, and all the row and column residuals for vertices in K_1 are given, then the total value of the cells in W can be computed in $O(|W| + |K_1|)$ time.*

Proof. We partition K_1 into a set R of row vertices and a set C of column vertices. Let $U(C)$ be the set of edges that are incident to the column vertices in C. We define similarly

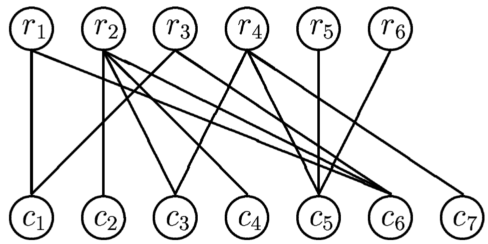

Figure 6: The suppressed graph of Table 7 when the table is general. Since the table is general, the suppressed graph is undirected.

$U(R)$ for R. Then $U(C) \cup W = U(R)$. Hence $\sum_{r \in W} value(r) = \sum_{v \in C} column\_residual(v) - \sum_{u \in R} row\_residual(u)$. □

Remark that a version of this lemma without the time complexity analysis is also in [Kao97a, Mal93].

Example X Let G be the graph in Figure 6. Then $\{(r_3, c_1), (r_1, c_6)\}$ is a cut in G but is not a bipartite cut. The set $\{(r_1, c_6), (r_3, c_6)\}$ is a bipartite cut but is not a row or column cut. The set $\{(r_1, c_1), (r_3, c_1)\}$ is a γ-cut, and $\{(r_2, c_6)\}$ is also a γ-cut. The row residuals for rows $r_1, \ldots, r_6$ are 4, 20, 4, -28, 8, and -29, respectively. The column residuals for columns $c_1, \ldots, c_7$ are 3, 3, 16, 4, -12, 9, and -44, respectively.

The cell corresponds to the γ-cut $\{(r_2, c_6)\}$ in Table 2 is $T(2,6)$. Note that $T(2,6) = (T(1,1) + T(3,1) + T(1,6) + T(2,6) + T(3,6)) - (T(1,1) + T(1,6) + T(3,1) + T(3,6)) = column\_residual(c_1) + column\_residual(c_6) - (row\_residual(r_1) + row\_residual(r_3)) = 4$.

For brevity, we say that a sum combination of (T, X, Y) *corresponds to* the set of edges in $G_{(T, X \cup \xi(Y), \emptyset)}$ that appear as the nonzero terms in that combination, and vice versa.

Theorem 15 *A sum combination is protected in a general table against any interval disclosure if and only if it is not the disjoint union of bipartite cuts.*

Proof. By Lemmas 4 and 14 and the fact that a general table has no lower or upper bounds for each cell. □

Theorem 15 is well-known when the sum combination has only one term, i.e., a single cell, and the security requirement is against exact disclosure [Gus87].

Theorem 16 *Assume that (T, X, Y) is general. Let S be a set of row and column invariants of (T, X, Y). Given S and $G_{(T', X \cup \xi(Y), \emptyset)}$, the adversary problem of evaluating invariants can be solved in $O(n + m + |X| + ||S||)$ time, where $||S||$ is the number of nonzero terms in S.*

Proof. It is shown in [Kao97a, Mal93] that given $G_{(T,X,\emptyset)}$, every invariant in a set S of row and column invariants of $(T, X, \emptyset)$ can be expressed as a linear combination of sum invariants corresponding to γ-cuts in $G_{(T,X,\emptyset)}$ in $O(n+m+|X|+||S||)$ time, where $||S||$ is the number of nonzero terms in S. Using this fact and Lemma 14, the theorem holds. ☐

5 Minimum Complementary Suppression Problems

Throughout this section, we assume that (T, X, Y) is general and $S \subseteq X \cup Y$. This section studies the following two versions of the minimum cell suppression problem for general tables.

- *The minimum marginal cell suppression problem.* Given S and (T, X, Y), find a set Y' of marginal complementary suppressed cells with the minimum cardinality such that every cell in S is protected in $(T, X, Y \cup Y')$.
- *The minimum general cell suppression problem.* Given S and (T, X, Y), find a set $X' \cup Y'$ of complementary suppressed cells with the minimum cardinality such that every cell in S is protected in $(T, X \cup X', Y \cup Y')$, where X' is a set of regular cells and Y' is a set of marginal cells.

The minimum general cell protection problem has been studied for the case where $Y = Y' = \emptyset$ [Gus87, Gus88]. The minimum marginal cell suppression problem has been studied for the case where Y only consists of the table sum [MMR91] and is extended in [MM97, MM96] to a table with a lower bound 0 for all cells.

Note that a solution to the minimum marginal cell suppression problem depends on whether the table sum is allowed to be suppressed. If the table sum is not allowed to be suppressed, then it may be the case that no matter how many additional marginal cells are suppressed, some suppressed cells are still unprotected, as in the case where the table has only one suppressed cell.

In [Gus87, Gus88], it is proved that a suppressed cell is protected in a general table $(T, X, \emptyset)$ if and only if it is a cut edge in $G_{(T,X,\emptyset)}$. Hence we transform our problem into the following graph-theoretical problem. The *smallest augmentation problem* [Esw73] is that of adding a set of edges with the minimum cardinality to a graph such that each added edge satisfies a given constraint and the resulting graph satisfies a given connectivity requirement. Depending on the edge constraint and the connectivity requirement, we have different versions of the smallest augmentation problem [Hsu93, Fra94].

Input: A graph $G = (V, E)$, $\Gamma \subseteq E$ and $\Lambda \subseteq \{(w_1, w_2) \mid w_1, w_2 \in V\}$.
Output: A smallest subset of edges $\mathcal{D} \subseteq \Lambda$ such that no edge in Γ is a cut edge in $G \cup \mathcal{D}$.

Let $2\text{eaug}(G, \Gamma, \Lambda)$ denote this augmentation problem. A *solution* to $2\text{eaug}(G, \Gamma, \Lambda)$ refers to a correct output as specified. We say that an instance of this problem is *solved* if either we determine that it has no solution or we find one of its solutions. Intuitively, we want to add edges to the suppressed graph such that the resulting graph contains no cut edge. If we are solving the minimum marginal cell suppression problem, then the added edges should correspond to marginal cells. For the minimum general cell suppression problem, the added edges may correspond to any un-suppressed cells.

Recall that $r_1, \ldots, r_n$ (respectively, $c_1, \ldots, c_m$) are the row (respectively, column) vertices in $G_{(T,X,Y)}$. Let $G^* = G_{(T',X\cup\xi(Y),\emptyset)}$. G^* has one extra row vertex r_{n+1} and one extra column vertex c_{m+1}. Let $\Phi = \{(c_i, r_{n+1}), (c_{m+1}, r_j) \mid 1 \leq i \leq m \text{ and } 1 \leq j \leq n\}$. By Lemma 2, the minimum marginal cell suppression problem is equivalent to $2\text{eaug}(G^*, S, \Phi)$. Similarly, let $\Phi' = \{(c_i, r_j)) \mid 1 \leq i \leq m+1 \text{ and } 1 \leq j \leq n+1\}$. The minimum general cell suppression problem is equivalent to $2\text{eaug}(G^*, S, \Phi')$.

5.1 The Minimum Marginal Cell Suppression Problem

To solve $2\text{eaug}(G^*, S, \Phi)$ and thus the minimum marginal cell suppression problem, we first simplify the structure of G^* via contractions. A *2-edge-block* in G^* is a maximal induced subgraph of G^* in which every pair of two distinct edges are contained in some cycle. Note that a cut edge is not in any 2-edge-block. The *2-edge-block forest* F [Har69] for G^* is the forest constructed from G^* by contracting each 2-edge-block of G^* into a single super vertex. For convenience, we identify the vertices in F with their corresponding 2-edge-blocks in G^*, and the edges in F with the cut edges in G^*.

Two 2-edge-blocks are *adjacent* if their corresponding vertices in F are adjacent. A 2-edge-block is a *leaf-block* if it is adjacent to exactly one 2-edge-block, i.e., the degree of its corresponding vertex in F is 1. A 2-edge-block is *hybrid* if it contains both row and column vertices. A 2-edge-block is *row-only* (respectively, *column-only*) if it contains only row (respectively, column) vertices. Note that a row-only or column-only 2-edge-block consists of a single vertex because G^* is bipartite. Hence a row-only or column-only 2-edge-block is also called a *singular* 2-edge-block.

Let b_r and b_c be the vertices in F corresponding to the 2-edge-blocks that contain the

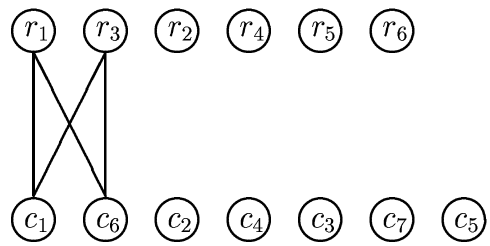

Figure 7: The 2-edge-blocks of Figure 2.

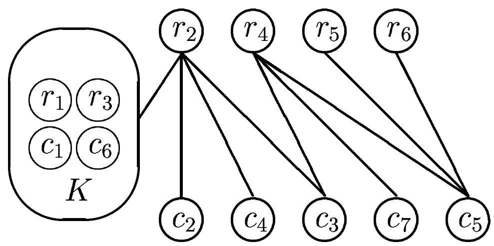

Figure 8: The 2-edge-block forest of Figure 2.

vertices r_{n+1} and c_{m+1}, respectively. The intuition here is to add edges to F such that one of the endpoints of each added edge is a degree 1 vertex in F and the other endpoint is either b_r or b_c. The added edges are added in two phases. First, we add those edges that must be added, i.e., one of whose endpoints is a degree 1 vertex in F that is row-only or column-only and b_r. For a degree-1 vertex p in F that is hybrid, we can either add (p, b_r) or (p, b_c). For each added edge (b', b''), we make sure that b' and b'' are not both row-only or both column-only. Hence for (b', b'') we can find a corresponding bipartite edge (u', u'') in G^* whose addition has the same effect of adding (b', b''), where u' and u'' are vertices in b' and b'', respectively.

Example XI The 2-edge-blocks of Figure 2 are shown in Figure 7. Eight of the nine 2-edge-blocks consists of a single vertex. The only 2-edge-block with more than one vertex is the one consists of vertices r_1, r_3, c_1 and c_6 and is hybrid. All the other 2-edge-blocks are singular, e.g., r_2 is row-only and c_2 is column-only.

The 2-edge-block forest of Figure 2 is shown in Figure 8. In the figure, the 2-edge-block containing the vertices r_1, r_3, c_1 and c_6 are contracted into a super vertex K. Degree-1 vertices in this 2-edge-block forest are S, c_2, c_4, c_7, r_5 and r_6. In Figure 8, b_r is r_6 and b_c is c_7.

For a column-only (respectively, row-only) degree-1 2-edge-block w in F, let legal(w)

index	1	2	3	4	5	6	sum
1		4	7	3	3		21
2	4				2		26
3		8	6	5	7		30
4	8	9		6		5	
5	4	4	5	9		2	
sum			34			16	152

Table 10: A minimum marginal cell protection solution for Table 2.

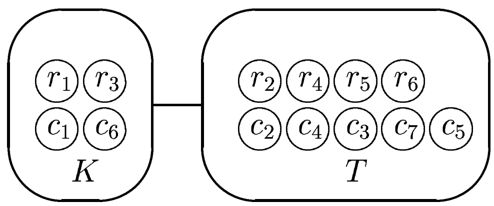

Figure 9: The 2-edge-block forest of Figure 8 after adding the set of edges $\{(c_2, r_6), (c_4, r_6), (r_5, c_7)\}$.

denote b_r (respectively, b_c).

Theorem 17 *Given S and $G_{(T', X \cup \xi(Y), \emptyset)}$, the minimum marginal cell suppression problem can be solved in $O(n + m + |X|)$ time.*

Proof. We solve the minimum marginal cell suppression problem with the following steps.

1. Let Q be the set of 2-edge-blocks in $\mathcal{P}(F, \eta(S))$ that are both singular and of degree-1. Let $\mathcal{D}_0 = \{(q, \text{legal}(q)) \mid q \in Q\}$. Add $\mathcal{D}_0$ to F to eliminate all singular 2-edge-blocks in F that are of degree-1. Let F' be the 2-edge-block forest of $F \cup \mathcal{D}_0$.

2. Let Q_1 be the set of degree-1 2-edge-blocks in F'. Let $\mathcal{D}_1$ be the set of edges $\{(w, b_r) \mid w \in Q_1 \setminus \{b_r\}\}$, where $\setminus$ denotes the set difference operator. Add $\mathcal{D}_1$ to F' to eliminate all degree-1 2-edge-blocks in F'.

3. We output $\mathcal{D}_0 \cup \mathcal{D}_1$ as a solution to the given instance of the minimum marginal cell suppression problem.

□

Example XII The first set of edges added to the F in Figure 8 is $\mathcal{D}_0 = \{(c_2, r_6), (c_4, r_6), (r_5, c_7)\}$. After adding $\mathcal{D}_0$, vertices c_2, c_3, c_4, c_5, c_7, r_2, r_4, r_5 and r_6 are in the 2-edge-block represented by the super vertex T. The 2-edge-block forest F' of $F \cup \mathcal{D}_0$ is shown in Figure 9.

Hence $\mathcal{D}_1 = \{(K, T)\}$. A set of edges corresponds to $\mathcal{D}_0 \cup \mathcal{D}_1$ in the graph shown in Figure 2 is thus $\{(c_2, r_6), (c_4, r_6), (r_5, c_7), (c_1, r_6)\}$. This is a solution for the minimum marginal cell protection problem. Table 10 gives the table obtained from Table 7 after suppressing $\{(c_2, r_6), (c_4, r_6), (r_5, c_7), (c_1, r_6)\}$.

Remark 1: If we are required to protect only a selected subset of original suppressed cells, we can prune vertices in F such that all degree-1 vertices in the pruned forest are incident to a cut edge in S before we apply the algorithm in Theorem 17.

Remark 2: The minimum marginal cell suppression problem may have no solution, i.e., it is not feasible to protect some sensitive suppressed cells by only suppressing additional marginal cells. We have proved that the minimum marginal cell suppression problem has no solution if and only if there is a 2-edge-block that is singular, of degree-1 in F, and it is adjacent to the vertex b_r or b_c. Intuitively, if the column (respectively, row) sum is the only cell that is suppressed in the column (respectively, row), then the minimum marginal cell suppression problem has no solution.

Theorem 17 subsumes the result in [MMR91] which only deals with the case where no marginal cell needs protection.

5.2 The Minimum General Cell Suppression Problem

Before we proceed, we define a *cut vertex* in a graph to be a vertex whose removal disconnects two vertices that are originally connected. In [HK96], a linear-time algorithm is given to find a smallest set of complementary suppressed cells whose addition makes the resulting graph biconnected. The algorithm uses a data structure similar to the 2-edge-block forest F used here. We note that by changing every cut edge $e = (u, v)$ in F to a cut vertex c_e and two additional edges (c_e, u) and (c_e, v), we obtain a data structure that is equivalent to the one used in [HK96]. By using this new data structure in the algorithm in [HK96], we can solve the minimum general cell suppression problem. We also note that the number of complementary suppressed cells so obtained is minimum.

This result assumes that there is no suppressed marginal cell. Using our extended table, we can create an equivalent table without suppressed marginal cells. Hence the following theorem holds.

index	1	2	3	4	5	6	sum
1			7	3	3		21
2	4				2		26
3		8	6	5	7		30
4	8	9		6		5	
5	4	4	5			2	32
sum	19	28	34	27		16	

Table 11: A solution for the minimum general cell suppression problem of Table 2.

Theorem 18 *Given S and $G_{(T',X\cup\xi(Y),\emptyset)}$, the minimum general cell protection problem can be solved in $O(n+m+|X|)$ time.*

Example XIII The set $\{(r_6,c_7),(r_1,c_2),(r_5,c_4)\}$ is an example of a set of edges added for Figure 2. The protected suppressed table is shown in Table 11 for the minimum general cell suppression problem. Note that it takes one fewer complementary suppressed cell than the solution found for the minimum marginal cell protection problem as shown in Example XII.

There is a subtle complication with the algorithm in Theorem 18. The algorithm may suppress the table sum of T. If the table sum is not allowed to be suppressed, then the following lemma can be used to enforce this constraint for the case when $S \subseteq X$.

Theorem 19 *Assume that $S \subseteq X$ and that $m \geq 2$ or $n \geq 2$. Let H be a set of cells in (T,X,Y) satisfying the output specification of the minimum general cell suppression problem and $|H| > 1$. Then, there exists some set H' with $|H'| = |H|$ such that H' also satisfies the output specification of the problem but does not contain the table sum of T.*

Proof. We assume without loss of generality that H contains the table sum of T. Our goal is to construct a desired H' from H.

Recall the definition of a 2-edge-block forest and related concepts from §5.1. Also recall that $\setminus$ is the set difference operator. Note that the table sum of T corresponds to the edge (r_{n+1},c_{m+1}) in G. Let $G' = G \cup H \setminus \{(r_{n+1},c_{m+1})\}$. By the minimality of H, the 2-edge-block forest of G' contains a path with the two endpoints b_c and b_r, where b_c contains c_{m+1} and b_r contains r_{n+1}. There are two cases.

Case 1: b_c and b_r are not both singular. We assume without loss of generality that b_c is non-singular. Let $H' = H \cup \{(r_{n+1},c_t)\} \setminus \{(r_{n+1},c_{m+1})\}$, where c_t is a column vertex in $b_c - \{c_{m+1}\}$. It is easily verified that H' is as desired.

index	1	2	3	4	5	6	sum
1			7	3	3		21
2	4				2		26
3		8	6	5	7		30
4	8	9		6		5	
5	4	4	5	9		2	
sum	19	28	34			16	152

Table 12: A solution for the minimum general cell suppression problem for Table 2 without suppressing the table sum.

Case 2: b_c and b_r are both singular. Since $|H| > 1$, let $e = (r^*, c^*)$ be an edge in H such that $e \neq (r_{n+1}, c_{m+1})$. Since b_c and b_r are both singular, $c^* \neq c_{m+1}$ and $r^* \neq r_{n+1}$. Let $H' = H \cup \{(r_{n+1}, c^*), (r^*, c_{m+1})\} \setminus \{(r^*, c^*), (r_{n+1}, c_{m+1})\}$. It can be verified that $G \cup H'$ is 2-edge-connected. □

Example XIV The set $\{(r_6, c_4), (r_1, c_2), (r_5, c_7)\}$ is an example of a set of edges added for Figure 2 without suppressing the table sum. This corresponds to *Case 2* in the proof of Theorem 19. In the proof, $e = (r_5, c_4)$. The corresponding protected table is shown in Table 12.

Note that in Theorem 19, if $H = \{(c_{m+1}, r_{n+1})\}$, then it is trivial to prove that it takes at least two edges to solve the minimum general cell suppression problem without suppressing the table sum. It is also trivial to find these two edges.

6 Extensions

We further consider the following security requirements:

1. **Row and Column Invariants.** A linear invariant f of (T, X, Y) is *semi-positive* if each regular cell has a nonnegative coefficient in f and each marginal cell has a non-positive coefficient. Given a row or column Z in (T, X, Y), let $\beta(Z) = 0$ if the marginal cell in Z is suppressed; otherwise, let $\beta(Z)$ be the suppressed marginal cell of Z. Let $I(Z)$ be the set of suppressed regular cells in Z. Let $\overline{Z} = \sum_{x \in I(Z)} x - \beta(Z)$.

   We define the following three security requirements for protecting linear combinations in (T, X, Y); see [Kao96] for motivations of these requirements.

- For *level 1* security of (T, X, Y), every suppressed cell in the table is protected and for each row or column Z in the table, every linear invariant of suppressed cells in Z is a multiple of $\overline{Z}$.
- For *level 2* security, given a constant positive integer k, every suppressed cell is protected and for each set of k rows $Z_1, \ldots, Z_k$ or k columns, but not the mixed case, every linear invariant of suppressed cells in $Z_1, \ldots, Z_k$ is a linear combination of $\overline{Z}_1, \ldots, \overline{Z}_k$.
- For *level 3* security, every suppressed cell is protected and every semi-positive invariant is a positive combination of $\overline{R}_1, \ldots, \overline{R}_n$, $\overline{C}_1, \ldots, \overline{C}_m$, where R_i and C_j range over the rows and columns in the table, respectively.

Given (T, X, Y), the *query system problem of testing level i security* is that of checking whether level i security of (T, X, Y) is satisfied. Kao [Kao95, Kao96] and Hsu and Kao [HK96] studied this query system problem for tables without suppressed marginal cells.

2. **Analytic Invariants.** Let $L \subseteq X \cup Y$. (T, X, Y) is *totally protected on* L if every analytic function whose arguments are cells in L is not an invariant. Given L and (T, X, Y), the *query system problem of testing total protection on* L is that of checking whether (T, X, Y) is totally protected on L. Given L and (T, X, Y), the *minimum general cell protection problem for achieving total protection on* L is that of finding a set $X' \cup Y'$ of the smallest number of cells such that $(T, X \cup X', Y \cup Y')$ is totally protected on L. Kao [Kao97b] studied this query system problem and its corresponding minimum general cell protection problem for tables without suppressed marginal cells.

We can solve all these additional problems while permitting suppressed marginal cells with the same running times as in [HK96, Kao95, Kao96, Kao97b] using techniques similar to those used in this paper. We omit the details here.

Acknowledgment

We thank Meral Özsoyoğlu and anonymous referees for helpful comments on earlier versions of this paper.

References

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applcations.* Prentice-Hall, Englewood Cliffs, New Jersey, 1993.

[AW89] N. R. Adam and J. C. Wortmann. Security-control methods for statistical database: A comparative study. *ACM Computing Surveys*, 21:515–556, 1989.

[CÖ82] F. Y. Chin and G. Özsoyoğlu. Auditing and inference control in statistical databases. *IEEE Transactions Software Engineering*, 8:574–582, 1982.

[Cox80] L. H. Cox. Suppression methodology and statistical disclosure control. *Journal of the American Statistical Association*, 75:377–385, 1980.

[Cox81] L. H. Cox. Linear sensitivity measures in statistical disclosure control. *Journal of Statistical Planning and Inference*, pages 153–164, 1981.

[Cox95] L. H. Cox. Network models for complementary cell suppression. *Journal of the American Statistical Association*, 432:1453–1462, 1995.

[dCDdSO94] F. Duarte de Carvalho, N. P. Dellaert, and M. de Sanches Osório. Statistical disclosure in two-dimensional tables: General tables. *Journal of the American Statistical Association*, 428:1547–1557, 1994.

[Den83] D. E. R. Denning. *Cryptography and Data Security.* Addison-Wesley, 1983.

[DS83] D. E. Denning and J. Schlörer. Inference controls for statistical databases. *IEEE Computer*, pages 69–82, July 1983.

[Esw73] K. P. Eswaran. Representation of graphs and minimally augmented Eulerian graphs with applications in data base management. Technical Report RJ 1305, IBM, Yorktown Heights, N.Y., 1973.

[Fra94] A. Frank. Connectivity augmentation problems in network design. In J. R. Birge and K. G. Murty, editors, *Mathematical Programming: State of the Art 1994*, pages 34–63. The University of Michigan, 1994.

[Gus87] D. Gusfield. Optimal mixed graph augmentation. *SIAM Journal on Computing*, 16:599–612, 1987.

[Gus88] D. Gusfield. A graph theoretic approach to statistical data security. *SIAM Journal on Computing*, 17:552–571, 1988.

[Har69] F. Harary. *Graph Theory*. Addison-Wesley, Reading, Massachusetts, 1969.

[HK96] T.-s. Hsu and M. Y. Kao. Optimal augmentation for bipartite componentwise biconnectivity in linear time. In T. Asano, Y. Igarashi, H. Nagamochi, S. Miyano, and S. Suri, editors, *Lecture Notes in Computer Science 1178: Proceedings of the 7th International Symposium on Algorithms and Computation*, pages 213–222. Springer-Verlag, New York, NY, 1996.

[HK97] T.-s. Hsu and M. Y. Kao. Security problems for statistical databases with general cell suppressions. In *Proceedings of the 9th International Conference on Scientific and Statistical Database Management*, pages 155–164, 1997.

[Hsu93] T.-s. Hsu. *Graph Augmentation and Related Problems: Theory and Practice*. PhD thesis, University of Texas at Austin, 1993.

[IJ94] R. W. Irving and M. R. Jerrum. Three-dimensional statistical data security problems. *SIAM Journal on Computing*, 23(1):170–184, 1994.

[Kao95] M. Y. Kao. Linear-time optimal augmentation for componentwise bipartite-completeness of graphs. *Information Processing Letters*, pages 59–63, 1995.

[Kao96] M. Y. Kao. Data security equals graph connectivity. *SIAM Journal on Discrete Mathematics*, 9:87–100, 1996.

[Kao97a] M. Y. Kao. Efficient detection and protection of information in cross tabulated tables II: Minimal linear invariants. *Journal of Combinatorial Optimization*, 1:187–202, 1997.

[Kao97b] M. Y. Kao. Total protection of analytic-invariant information in cross-tabulated tables. *SIAM Journal on Computing*, 26:231–242, 1997.

[KG93] M. Y. Kao and D. Gusfield. Efficient detection and protection of information in cross tabulated tables I: Linear invariant test. *SIAM Journal on Discrete Mathematics*, 6:460–476, 1993.

[KGA92] J. P. Kelly, B. L. Golden, and A. A. Assad. Cell suppression: Disclosure protection for sensitive tabular data. *Networks*, 22:397–417, 1992.

[Mal93] F. M. Malvestuto. A universal-scheme approach to statistical databases containing homogeneous summary tables. *ACM Transactions on Database Systems*, pages 678–708, 1993.

[MM96] F. M. Malvestuto and M. Moscarini. Censoring statistical tables to protect sensitive information: Easy and hard problems. In *Proceedings of the 8th International Conference on Scientific and Statistical Database management*, pages 12–21, 1996.

[MM97] F. M. Malvestuto and M. Moscarini. Suppressing marginal totals from a two-dimensional table to protect sensitive information. *Statistics and Computing*, 7:101–114, 1997.

[MMR91] F.M. Malvestuto, M. Moscarini, and M. Rafanelli. Suppressing marginal cells to protect sensitive information in a two-dimensional statistical table. In *Proceedings of ACM Symposium on Principles of Database Systems*, pages 252–258, 1991.