Journal of the Operational Research Society

ISSN: 0160-5682 (Print) 1476-9360 (Online) Journal homepage: https://www.tandfonline.com/loi/tjor20

A three-phase algorithm for the cell suppression problem in two-dimensional statistical tables

F D Carvalho & M T Almeida

To cite this article: F D Carvalho & M T Almeida (2008) A three-phase algorithm for the cell suppression problem in two-dimensional statistical tables, Journal of the Operational Research Society, 59:4, 556-562, DOI: 10.1057/palgrave.jors.2602389

To link to this article: https://doi.org/10.1057/palgrave.jors.2602389

Journal of the Operational Research Society (2008) 59, 556–562

www.palgrave-journals.com/jors

A three-phase algorithm for the cell suppression problem in two-dimensional statistical tables

FD Carvalho* and MT Almeida

ISEG, Universidade Técnica de Lisboa, Lisbon, Portugal and CIO – Centro de Investigação Operacional

To obtain full cooperation from respondents, statistical offices must guarantee that confidential data will not be disclosed when their reports are published. For tabular data, cell suppression is one of the preferred techniques to control statistical disclosure. When suppressing only confidential values does not guarantee the desired data protection, it is also necessary to suppress the values in some non-confidential cells. The problem of finding an optimal set of complementary suppressions—the cell suppression problem (CSP)—is NP-hard. We present a three-phase algorithm for the CSP based on a binary relaxation derived from row and column protection conditions. To enforce violated single cell conditions, integer cuts are added to the CSP relaxation. The numerical results obtained in 1410 instances with up to more than 250 000 cells, which were generated to reproduce two classes of real-world data, indicate that the algorithm is quite effective for both classes of instances and that it outperforms state-of-the-art algorithms for one of them.

Journal of the Operational Research Society (2008) **59,** 556–562. doi:10.1057/palgrave.jors.2602389
Published online 28 February 2007

Keywords: integer programming; networks and graphs; heuristics; cell suppression problem

Introduction

Statistical offices face the challenge of disseminating as much data as possible while, at the same time, protecting the right to privacy by guaranteeing confidentiality where appropriate. Ever-growing demands made by public and private decision makers for statistical information come up against increasingly sophisticated computational and statistical technology that is available to intruders, which makes this conflict difficult to resolve. In recent years academics and practitioners have devoted a great deal of effort to developing and implementing efficient statistical methods to control information disclosure. For an overview of these efforts, see Willenborg and de Waal (2001), Doyle *et al* (2001) and Domingo-Ferrer and Torra (2004).

Statistical data are frequently published as two-dimensional arrays, called statistical tables, which are obtained from microdata files by aggregation. In practice, tables are released with row and column subtotals. As a rule, statistical tables contain either frequency counts for sociological aspects, or magnitude data for economic aspects. In frequency count tables, if the value of a cell is very small, some of the respondents may be identified by an intruder that has some information on their other attributes. In magnitude tables, if very few respondents contribute to make up a large share of the cell value, it may be possible for one of them to compute narrow estimates for some of the other individual contributions to the cell (Willenborg and de Waal, 2001). To avoid the disclosure of confidential information in two-dimensional non-negative tables, some cell values may have to be either perturbed or suppressed. When confidential values are suppressed, the additive structure of the table provided by row and column subtotals may still make it possible for an intruder to deduce the missing information or to compute narrow bounds for the information. When this is the case, values in non-confidential cells—called complementary suppressions—must also be omitted to make the table safe for publication. Once the lower and upper protection limits for each confidential value have been defined, the goal is then to guarantee that the narrowest ranges an intruder can compute for confidential figures do not fall within these limits.

Since making complementary suppressions leads to a loss of non-confidential information, the cell suppression problem (CSP) consists of finding a set of complementary suppressions that guarantees protection to all confidential cells while simultaneously minimizing the loss of information. This problem is known to be NP-hard (Kelly *et al*, 1992). Heuristic algorithms for the CSP are presented in Kelly *et al* (1992) and Cox (1995), among others. Carvalho and Almeida (2000) proposed new necessary safety conditions to derive lower bounds on the CSP optimum. Fischetti and Salazar (1999, 2000) developed branch-and-cut algorithms to solve the CSP for two-dimensional tables and for tabular data with linear constraints. Recently, Gonzalez Jr and Cox (2005) presented a desktop software system that implements cell suppression and three other techniques to protect two-dimensional tables. Other versions of the problem are considered in Gusfield (1988),

*Correspondence: FD Carvalho, Instituto Superior de Economia e Gestão, Rua do Quelhas 6, Lisbon 1200-781, Portugal.
E-mail: filipadc@iseg.utl.pt

Kelly *et al* (1992), Carvalho *et al* (1994a,b), Fischetti and Salazar (2003) and Almeida and Carvalho (2005).

In this paper, we present a three-phase algorithm for the CSP, based on an integer relaxation of the problem to which integer cuts are iteratively added. The relaxation used to start the algorithm is an improvement of the lower-bounding model in Carvalho and Almeida (2000). The model is enlarged with a new set of constraints, associated with rows and columns whose subtotal is confidential, and the whole set of constraints is reformulated. Owing to the new constraints, the lower bound is much tighter, especially when large row or column subtotal values are confidential. The reformulation results in an important reduction on memory and run time in phase 1. In phase 2 the generation of a heuristic solution is coupled with a procedure to find single cell protection violations, if any. This is an efficient way of either proving the optimality of the current solution or identifying integer cuts to generate in phase 3. The results obtained in 1410 randomly generated instances with up to more than 250 000 cells, reproducing two classes of real-world tables, indicate that the algorithm is quite effective for both classes and that it outperforms state-of-the-art algorithms for the class having confidential cells with large values.

This paper is organized as follows. The first three sections are devoted to the presentation of the terminology and notation, to the discussion of the safety conditions and to a brief description of the heuristic algorithm to generate near optimal solutions. The next two sections present the three-phase algorithm and its computational performance on two classes of randomly generated instances. The last section contains the conclusions.

Terminology and notation

A two-dimensional statistical table $\mathscr{A} = [a_{ij}]$ is a $(m+1) \times (n+1)$ array of non-negative numbers. The values in the $(m+1)$th row are the column subtotals, and the values in the $(n+1)$th column are the row subtotals. The subtotals will also be called marginal cells. The value $a_{m+1,n+1}$ is the grand total. We assume that $a_{m+1,n+1} > 0$.

The set of all cells of a table will be denoted by $\mathscr{C}$, the set of its rows by R and the set of its columns by C. The set of confidential cells will be denoted by S_1 and its cardinality will be denoted by p. When convenient to simplify the notation, the set S_1 of confidential cells will be considered in the form $S_1 = \{(i_k, j_k) : 1 \leqslant k \leqslant p\}$. The set of confidential cells in row $i \in R$ (respectively column $j \in C$) will be denoted by $S_1^r(i)$ (respectively $S_1^c(j)$). For each confidential cell (i, j), l_{ij} will represent its lower protection level and u_{ij} its upper protection level. The set of non-confidential cells will be denoted by $\mathscr{C} \setminus S_1$. Sets of complementary suppressions will be denoted by S_2.

A cell $(i, j) \in S_1$ is left-protected (respectively right-protected) if, after the suppression of all values in $S_1 \cup S_2$, the tightest range an intruder is able to compute for a_{ij} contains the lower protection limit $a_{ij} - l_{ij}$ (respectively upper protection limit $a_{ij} + u_{ij}$). A cell in S_1 is protected if it is both left-protected and right-protected. To determine the tightest range for a given cell $(i, j) \in S_1$, an intruder has to solve a pair of linear programming problems—a maximization problem for its upper limit and a minimization problem for its lower limit—whose feasible region is defined by the additive structure of the table (Fischetti and Salazar, 1999).

-	-	35	3	158
-	-	40	10	75
10	15	30	5	60
125	45	105	18	293

-	-	35	3	158
-	-	40	-	75
10	-	30	-	60
125	45	105	18	293

Figure 1 S_2' and S_2''

Consider a table with $m = 3$ and $n = 4$, $S_1 = \{(1, 1)\}$, $a_{11} = 100$, $l_{11} = u_{11} = 0.15 \times a_{11}$, and two sets of complementary suppressions, $S_2' = \{(1, 2), (2, 1), (2, 2)\}$ and $S_2'' = \{(1, 2), (2, 1), (2, 2), (2, 4), (3, 2), (3, 4)\}$, depicted in Figure 1.

With S_2' an intruder can deduce that a_{11} is in the range [90, 115], so S_2' does not protect cell (1,1). With S_2'' the narrowest range an intruder can compute for a_{11} includes [85, 115], so S_2'' protects cell (1,1).

A set S_2 of complementary suppressions is feasible if, after suppressing the values in $S_1 \cup S_2$, all confidential cells are protected, that is the table becomes safe for publication. The volume of non-confidential information lost with S_2 is given by $\omega(S_2) = \sum_{(i,j) \in S_2} a_{ij}$. The value $\omega(S_2)$ will be called the cost of S_2.

A two-dimensional table may be represented by a directed bipartite network $\mathscr{N} = (V, \mathscr{A})$, with capacities defined on its arcs (Kelly *et al*, 1992; Fischetti and Salazar, 1999). The node set $V = R \cup C$ is the union of a set R of $m+1$ nodes, representing the table rows, with a set C of $n+1$ nodes, representing the table columns. Each cell (i, j) of the table is represented by two directed arcs: the forward arc (i, j) from row node i to column node j, and the reverse arc (j, i) from column node j to row node i. The capacities on the forward arcs are $c_{ij} = +\infty$ if (i, j) is either an internal cell or the grand total cell, and $c_{ij} = a_{ij}$ if (i, j) is a marginal cell. The capacities on the reverse arcs are $c_{ji} = a_{ij}$ if (i, j) is either an internal cell or the grand total cell, and $c_{ji} = +\infty$ if (i, j) is a marginal cell. Given a set S_2 of complementary suppressions, $\mathscr{N}_{S_1 \cup S_2} = (V, \mathscr{A}_{S_1 \cup S_2})$ is the subnetwork that represents only the suppressed cells. For every $k \in \{1, \ldots, p\}$, $\mathscr{N}^k_{S_1 \cup S_2} = (V, \mathscr{A}^k_{S_1 \cup S_2})$ is the network that results from $\mathscr{N}_{S_1 \cup S_2}$ removing arcs (i_k, j_k) and (j_k, i_k).

Safety conditions

Every feasible set of complementary suppressions must comply with several classes of pattern and volume conditions,

(Kelly *et al*, 1992; Carvalho and Almeida, 2000). Solutions that satisfy pattern and volume constraints but do not protect one or more confidential cells may be eliminated by integer cuts. Pattern and volume conditions are reviewed below and generalized to include a new class. Their mathematical formulation in Carvalho and Almeida (2000) is revised to reduce the dimension of the resulting problem and to strengthen its LP-relaxation. Finally, a flow-based formulation for integer cuts is proposed.

Pattern and volume constraints

Kelly *et al* (1992) proposed an integer model to compute a lower bound on the CSP optimum derived from the observation that if in a row (or column) there is a unique suppression, then its value can always be computed from the published values. This observation sets the ground for two pattern conditions:

(a) if in a row (or column) there is a unique confidential cell, then at least one complementary suppression must be made in that row (or column);
(b) if in a row (or column) there is no confidential cell, then either no or at least two complementary suppressions must be made in that row (or column).

As the rationale for rows and for columns is the same, from now on we will consider only rows. For each row condition there is always a column counterpart.

For rows, the integer model in Kelly *et al* (1992) has two sets of binary variables, x_{ij} and ρ_i, associated with non-confidential cells and with rows having no confidential cells, respectively, defined by

$$\underset{(i,j)\in\mathscr{C}\setminus S_1}{x_{ij}} = \begin{cases} 1 & \text{if } (i,j)\in S_2 \\ 0 & \text{otherwise} \end{cases}$$

$$\underset{i\in\{i:S_1^r(i)=\emptyset\}}{\rho_i} = \begin{cases} 1 & \text{if at least one cell is suppressed in row } i \\ 0 & \text{otherwise} \end{cases}$$

and the following constraints

$$\sum_{(i,j)\in\mathscr{C}\setminus S_1} x_{ij} \geqslant 1, \qquad i\in\{i:|S_1^r(i)|=1\} \tag{1}$$

$$\sum_{(i,j)\in\mathscr{C}} x_{ij} \geqslant 2\rho_i, \qquad i\in\{i:S_1^r(i)=\emptyset\} \tag{2}$$

$$\rho_i \geqslant x_{ij}, \qquad i\in\{i:S_1^r(i)=\emptyset\}, \quad j\in C \tag{3}$$

Constraints (1) guarantee condition (a) and constraints (2)–(3) guarantee condition (b).

The ρ_i variables may be eliminated, substituting (2) and (3) by

$$\sum_{(i,j)\in\mathscr{C}} x_{ij} \geqslant 2x_{is}, \qquad i\in\{i:S_1^r(i)=\emptyset\}, \quad s\in C \tag{4}$$

With the same two sets of variables, Carvalho and Almeida (2000) added volume constraints to Kelly *et al*'s lower-bounding model, based on the following observations:

(c) if in a row there is a unique confidential cell, it is possible to compute a lower bound on the volume of non-confidential information that must be omitted in that row;
(d) for some rows with two or more confidential cells whose marginal is not confidential, it is also possible to guarantee that at least one complementary suppression must be made and to compute a lower bound on the volume of non-confidential information that must be omitted in each such row;
(e) for every row with no confidential cells and a positive marginal cell, if non confidential cells are suppressed, the volume of non-confidential information suppressed in that row must be at least equal to its smallest positive cell value.

The conditions in Carvalho and Almeida (2000) may be generalized to apply to rows with two or more confidential cells whose marginal is confidential. This generalization is decisive for the good overall performance of the three-phase algorithm because it leads to a significant improvement in the optimum of the integer relaxation for many instances. To reduce the problem's dimensions and to strengthen its LP-relaxation, an alternative formulation to the one in Carvalho and Almeida (2000) is proposed next.

Let R_α be the set of rows i with at least one confidential cell, with a non-confidential marginal cell and such that $g_u(i) = \max\{a_{ij}+u_{ij} : (i,j)\in S_1^r(i)\} - \sum_{(i,j)\in S_1} a_{ij} > 0$. A volume of non-confidential information at least equal to $g_u(i)$ must be suppressed in each row $i\in R_\alpha$.

Let R_β be the set of rows i such that $g_l(i) = l_{i,n+1} - \sum_{(i,j)\in S_1\setminus\{(i,n+1)\}} a_{ij} > 0$. A volume of non-confidential information at least equal to $g_l(i)$ must be suppressed in each row $i\in R_\beta$.

Let $R_\emptyset$ be the set of rows described in (e). If complementary suppressions are made in a row $i\in R_\emptyset$, at least one of them must be positive.

These three sets of conditions may be formulated as follows.

$$\sum_{(i,j)\in\mathscr{C}\setminus S_1} \gamma_{ij}x_{ij} \geqslant g_u(i), \qquad i\in R_\alpha \tag{5}$$

$$\sum_{(i,j)\in\mathscr{C}\setminus S_1} \gamma_{ij}x_{ij} \geqslant g_l(i), \qquad i\in R_\beta \tag{6}$$

$$\sum_{(i,j)\neq(i,s)} x_{ij} \geqslant x_{is}, \qquad i\in R_\emptyset, \quad s\in C: a_{is}>0 \tag{7}$$

$$\sum_{(i,j):a_{ij}>0} x_{ij} \geqslant x_{is}, \qquad i\in R_\emptyset, \quad s\in C: a_{is}=0 \tag{8}$$

with the coefficients γ_{ij} in (5) and (6) defined as

$$\gamma_{ij} = \begin{cases} \min\{a_{ij}, g_u(i)\} & \text{if } i \in R_\alpha \\ \min\{a_{ij}, g_l(i)\} & \text{if } i \in R_\beta \end{cases}$$

Note that conditions (5)–(8) dominate conditions (1) and (4), which may therefore be eliminated. Note also that when the integrality constraints are relaxed to $0 \leqslant x_{ij} \leqslant 1$, constraints (5) and (6) still dominate constraints (1), but the dominance would not hold if coefficients a_{ij}, rather than γ_{ij}, were used, as in Carvalho and Almeida (2000). This dominance contributes to the reduction of the run time for the first phase of the algorithm.

If suppressions are made in a row with zero marginal value, this marginal value must be suppressed. With $R^0_\emptyset$ representing rows with zero marginal value, this condition may be formulated as follows:

$$\sum_{(i,j)\neq(i,n+1)} x_{ij} \geqslant x_{i,n+1}, \qquad i \in R^0_\emptyset \tag{9}$$

$$x_{i,n+1} \geqslant x_{ij}, \qquad i \in R^0_\emptyset, \quad j \in C\backslash\{(n+1)\} \tag{10}$$

The counterpart of conditions (5)–(10) for columns will be referred to as (5c)–(10c).

Flow constraints

The network flow representation of the CSP may be used to formulate a necessary and sufficient condition for a confidential cell to be protected with a set S_2 (Kelly *et al*, 1992). Consider the networks $\mathscr{N}^k_{S_1\cup S_2}$ $(1 \leqslant k \leqslant p)$ associated with S_2. An internal cell or the grand total, $(i_k, j_k) \in S_1$, is right-protected if and only if u_k units of flow may be sent from column node j_k to row node i_k, and it is left-protected if and only if l_k units of flow may be sent from row node i_k to column node j_k. A marginal cell, $(i_k, j_k) \in S_1$, is right-protected if and only if u_k units of flow may be sent from row node i_k to column node j_k, and it is left-protected if and only if l_k units of flow may be sent from column node j_k to row node i_k.

If a set S_2 of complementary suppressions that verifies constraints (5)–(10), (5c)–(10c) does not make the table safe for publication, then at least one of the flows above is not feasible in the corresponding network $\mathscr{N}^k_{S_1\cup S_2}$. If s is the source node, t is the sink node and f is the value of such a flow associated with a confidential cell (i_k, j_k), then the following flow constraints eliminate S_2:

$$\sum_v z^k_{uv} - \sum_v z^k_{vu} = \begin{cases} f & \text{if } u = s \\ -f & \text{if } u = t \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

$$0 \leqslant z^k_{uv} \leqslant \xi_{uv} x_{uv}, \qquad 0 \leqslant z^k_{vu} \leqslant \xi_{vu} x_{uv}, \quad (u, v) \in \mathscr{C}\backslash S_1 \tag{12}$$

$$0 \leqslant z^k_{uv} \leqslant \xi_{uv}, \qquad 0 \leqslant z^k_{vu} \leqslant \xi_{vu}, \quad (u, v) \in S_1 \tag{13}$$

where variables z^k represent the flows along the arcs in $\mathscr{A}^k_{S_1\cup S_2}$, $\xi_{uv} = \min\{c_{uv}, f\}$ and $\xi_{vu} = \min\{c_{vu}, f\}$. In general, for small values of f, the sets of st-paths that carry the flow at minimum cost have a simple structure, whereas for large values of f, it is necessary to find a complex pattern of non-disjoint st-paths to minimize the suppression costs. This explains why, in practice, the CSP is harder to solve on tables having confidential cells with large values.

Through the max flow–min cut theorem (Ahuja *et al*, 1993), the flow variables may be projected into the x variable space. However, the model is faster to solve with the flow structure (11)–(13).

Heuristic

To generate near optimal solutions we use the iterative procedure that protects one cell at a time (Kelly *et al*, 1992). As suggested in Carvalho *et al* (1994b), the pair of minimum cost flow problems associated with the protection of each confidential cell is solved heuristically by a sequence of shortest path problems. A similar procedure is discussed in Fischetti and Salazar (1999).

Different solutions may be obtained by modifying the costs used in the path computations to penalize or to favour the inclusion of some cells. In our experiment, assigning zero costs to the cells in the optimal solution of the CSP relaxation led to the best results. As the heuristic considers one confidential cell at a time, the final solution may include redundant complementary suppressions. For each suppression $(i, j) \in S_2$, the heuristic routine is run on $\mathscr{N}_{S_1\cup S_2\backslash\{(i,j)\}}$. Every time a feasible solution is found, the corresponding suppression (i, j) is identified as redundant and discarded. This procedure is known in the literature as a clean up.

Three-phase algorithm

The algorithm starts with the solution of an integer relaxation of the CSP to obtain a set of complementary suppressions. If this set protects all confidential cells, the CSP is solved. Otherwise a near optimal solution is generated and the uncovered protection limits are identified. New constraints are then added to the integer model to enforce those protection limits. The procedure is repeated until all confidential cells are protected.

A description of each phase of the algorithm is given next.

Phase 1

Phase 1 of an iteration q consists of solving an integer relaxation of the CSP, denoted by (P^q).

For $q = 1$ (P^q) is:

$$\begin{aligned} (P) \quad \min \quad & Z = \sum_{(i,j)\in\mathscr{C}\backslash S_1} a_{ij} x_{ij} \\ \text{s.t.} \quad & (5)\text{–}(10), \ (5c)\text{–}(10c) \\ & x_{ij} \in \{0, 1\}, \qquad (i, j) \in \mathscr{C}\backslash S_1. \end{aligned}$$

For $q > 1$, (P^q) is the problem that results from adding to (P) the constraints generated in phase 3 of all preceding iterations.

If the optimal solution of (P^q) is feasible for the CSP, the problem is solved. Otherwise, solving (P^q) provides a lower bound on the CSP optimum value as well as a set of complementary suppressions—say S_2^q—that does not protect all cells in S_1. A feasibility check is performed in phase 2. If S_2^q is feasible, the algorithm stops. Otherwise, new constraints are identified.

Phase 2

In phase 2 of an iteration q, the heuristic algorithm is run with the matrix of modified costs:

$$\alpha'_{ij} = \begin{cases} 0 & \text{if } (i, j) \in S_1 \cup S_2^q \\ a_{ij} & \text{otherwise} \end{cases}$$

Every time an arc with non-zero modified cost is used in a path associated with the lower protection level of a cell (i_k, j_k), this level is included in a list of lower protections. A list of upper protections is built likewise.

If, after executing the heuristic algorithm, both lists are empty, S_2^q is optimal for the CSP, and the algorithm stops. Otherwise, the clean up procedure is called for the heuristic solution. If redundant suppressions are found, their removal generates a new feasible solution for the CSP. If this new solution has a cost equal to the current lower bound value, it is optimal, and the algorithm stops. Otherwise, the lists of lower and upper protections are used in phase 3 to generate new constraints.

Phase 3

In phase 3 of an iteration q, constraints of the form (11)–(13) are generated for the lower and upper protections in the lists built during phase 2 of the current iteration. These new constraints are added to (P^q) to generate problem (P^{q+1}), and the lists are emptied.

Computational experiments

The computational study was carried out on a 866 MHz PC PentiumIII processor with 128 Mb RAM. The integer problems were solved by Cplex 8.0 (ILOG, 2002) with a time limit of 1 h.

To implement the three-phase algorithm, we programmed codes in Pascal with Delphi-32 development environment editor. The shortest path problems needed to generate near optimal solutions were computed with Dijkstra's algorithm (Ahuja *et al*, 1993). In the pre-processing procedure, Dijkstra's algorithm was modified in order to solve maximum capacity path problems.

Data

The computational experiments were performed on two randomly generated datasets, denoted Class I and Class II. To allow meaningful comparisons, we followed generation rules described in Kelly *et al* (1992) and Fischetti and Salazar (1999).

The number of Class I and Class II instances generated was 580 and 830, respectively, with dimensions $m \times n$ ranging from 10×10 up to 500×500. For each combination $m \times n$ the three-phase algorithm was tested on 10 instances. The values used for n were 10, 20, ..., m for $m \leqslant 100$ and 50, 100, ..., m for $m > 100$.

Instances in Class I were randomly generated as the first set of random instances presented in Fischetti and Salazar (1999). Every internal cell has a random integer value in [0, 499], zero valued cells cannot be suppressed, and all cells with values in [1,4] are confidential. The upper and lower protection levels are $u_{ij} = a_{ij}$ and $l_{ij} = a_{ij} - 1$. These rules were suggested by a member of ISTAT, the Italian statistical office, to reproduce real-world statistical tables.

Class II instances were generated following the rules proposed in Kelly *et al* (1992), which were also used for the second set of random instances presented in Fischetti and Salazar (1999). Every internal cell has a random integer value in [0,1000]. Zero valued cells cannot be suppressed, and non-zero internal and marginal cells are confidential with probability of 0.2 and of 0.1, respectively. Upper and lower protection levels for each confidential cell are both set to 15% of the cell value, rounded up to the nearest integer.

Problem pre-processing

A confidential cell may be automatically right-protected and/or left-protected, that is a cell may not require complementary suppressions for its upper and/or lower protection. This happens when an intruder is unable to compute estimates below the right protection limit and/or above the left protection limit, due to the pattern and the values of the other confidential cells. To identify redundant protection limits, it is necessary to execute a safety check. In order to reduce the time spent in problem pre-processing, a modified version of the constructive heuristic was adopted. It was run on the network $\mathcal{N}_{S_1} = (V, \mathcal{A}_{S_1})$, which represents only the confidential cells (all with zero cost), substituting the shortest path routine by a maximum capacity path routine (Ahuja *et al*, 1993) to speed up the computations. Each protection level for which a set of paths is identified whose total capacity is at least equal to the level's value may be ignored, as it is satisfied *a priori*.

If, for a row $i \in R_\alpha$, the right-hand side of the corresponding constraint (5) is greater than the sum of its internal non-confidential cells, then its confidential cells can only be protected by suppressing the marginal value. In this case, the internal values are too small to provide protection to all confidential cells in row i, even if they are all suppressed. The same rationale applies to columns and constraints (5c). For all such rows i and columns j, variables $x_{i,n+1}$ and $x_{m+1,j}$ are set to one.

Results

Tables 1 and 2 show the results obtained with the three-phase algorithm on Class I and Class II data, respectively. Each row

Table 1 Class I

m	*inst*	*opt*	*Popt* (%)	1*gap* (%)	*time* (*s*)
20	20	20	15.0	21.44	22.85
40	40	40	12.5	8.09	132.35
60	60	53	38.3	3.45	39.81
80	80	76	38.8	2.97	37.89
100	100	95	55.0	1.82	32.64
200	40	39	95.0	0.81	26.88
300	60	60	100.0	—	33.17
400	80	80	100.0	—	6.00
500	100	100	100.0	—	6.82

Table 2 Class II

m	*inst*	*opt*	*Popt* (%)	1*gap* (%)	*time* (*s*)
10	10	10	40.0	27.85	0.40
20	20	20	25.0	8.30	3.15
30	30	30	70.0	5.66	0.51
40	40	40	65.0	25.82	2.40
50	50	50	84.0	4.50	1.10
60	60	60	83.3	12.37	35.21
70	70	70	94.3	4.82	15.65
80	80	80	92.5	8.18	23.50
90	90	87	91.1	8.74	154.67
100	100	95	88.0	4.69	219.59
200	40	35	92.5	0.28	617.46
300	60	55	91.7	0.69	365.81
400	80	61	76.3	0.40	882.41
500	100	78	78.0	0.06	873.85

contains the following information: *m*, number of rows; *inst*, number of instances; *opt*, number of proven optimal solutions; *Popt*, percentage of solutions proven optimal at iteration 1; 1*gap*, average gap over the instances not yet solved at the end of iteration 1; *time*, average run time (in seconds).

The gaps were computed as $\mathrm{UB} - \mathrm{LB}/\mathrm{LB} \times 100\%$, where UB and LB stand for the best upper and lower bounds obtained for the CSP optimum.

The values for 1*gap* were computed at the end of iteration 1, with the values of the heuristic solutions generated in phase 2, for UB, and the optima of (*P*), for LB. If the optimum of (*P*) was not yet available when the time limit was hit, its best-known lower bound was used instead.

The pre-processing procedure identified 15.2% of Class I and 2.7% of Class II instances as requiring no complementary suppressions. The average pre-processing times were 52.4 s for each Class I instance and 207.9 s for each Class II instance.

The three-phase algorithm solved to proven optimality 563 Class I instances, that is 97.1% of them. In the remaining 17 instances memory limitations occurred after short run times in phase 1 of either iteration 2 or of iteration 3. For these instances the algorithm produced near optimal solutions, with an average percentage gap below 3.5%. The three-phase algorithm was very successful for the largest instances, failing to solve to proven optimality only 1 of the 280 instances with $m > 100$. The overall average run time was less than 1 min. The algorithm required only one iteration to find 70% of the proven optimal solutions. For Class I instances, the results obtained with the three-phase algorithm were slightly inferior to the results reported in Fischetti and Salazar (1999): on a similar set of 580 instances their branch-and-cut algorithm solved all instances to proven optimality.

In the set of 830 Class II instances, the three-phase algorithm solved to proven optimality 771 instances, that is 92.9% of them. For the remaining 59 instances, the optimal solution of (*P*) had not yet been found when the 1-h time limit was hit. In all these instances, the best solution known for (*P*) when hitting the time limit was identified as feasible for the CSP in the second phase of the iteration. For these instances, the algorithm produced near optimal solutions with an average gap of 0.44%. The key to these results is the tightness of the lower bound given by (*P*); it improved the lower bound in Carvalho and Almeida (2000) for 92% of the instances tested and was the optimum of the CSP for 89% of the instances solved to proven optimality. The short run times of the three-phase algorithm—less than 5 min on average—were partially due to the fact that the LP-relaxation of (*P*) had optimal integer solution for 40% of the instances.

We could not find in the literature computational results of exact methods for Class II instances with $m > 100$. The three-phase algorithm solved to proven optimality 81.8% of the 280 instances tested. For the remaining instances, it produced solutions with an average gap of 0.50%. For $m \leqslant 100$, the results shown in Table 2 may be compared with the results of the branch-and-cut algorithm for instances with the same dimensions, presented in Fischetti and Salazar (1999). The branch-and-cut algorithm solved to proven optimality 91.8% of the instances tried (with no failures in instances with up to $m \times n = 50 \times 50$). For the remaining instances, it generated solutions with an average gap of 0.27%. The corresponding figures for the three-phase algorithm were 98.5% ($m \times n = 80 \times 80$) and 0.03%, respectively.

The three-phase algorithm was also tested to minimize the number of complementary suppressions $\sum_{(i,j)\in\mathscr{C}\setminus S_1} x_{ij}$. This objective is mentioned for the CSP in Willenborg and de Waal (2001) and Cox (1995), among others. In the Class I set, the number of instances solved to proven optimality decreased with the change of objective function. By contrast, in the Class II set, all 830 instances were solved to proven optimality, and the average run time decreased significantly as the 1-h time limit set to the solution of the integer models was never hit.

Conclusions

We have presented an integer cut algorithm for a well-known problem arising in the statistical disclosure control of data published in two-dimensional tables—the CSP. The algorithm is based on a new integer relaxation of the CSP, which is very tight, even for the class of instances known from

experience to be the most difficult to solve in practice. As the separation procedure is performed over integer structures, it is less complex than its counterpart in a branch-and-cut setting. The formulation proposed for the cuts induces a structure that speeds up the solution of the resulting model. The computational experience in 1410 tables that were generated to reproduce two classes of real-world statistical tables indicates that the algorithm is quite effective for both classes. For the class of statistical tables where the state-of-the-art solution methods have proved less successful, the new algorithm outperforms the best exact method in the literature. Statistical tables in that class are particularly relevant in business data.

Acknowledgements—The authors thank two anonymous referees for their comments and suggestions. Thanks are also due to Ann Henshall for her assistance in editing the final version of the paper.

References

Ahuja RK, Magnanti TL and Orlin JB (1993). *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall: Englewood Cliffs, NJ.

Almeida MT and Carvalho FD (2005). Exact disclosure prevention in two-dimensional statistical tables. *Comput Opns Res* **32**: 2919–2936.

Carvalho FD and Almeida MT (2000). Lower-bounding procedures for the 2-dimensional cell suppression problem. *Eur J Opl Res* **123**: 29–41.

Carvalho FD, Dellaert N and Osório M (1994a). Statistical disclosure in two-dimensional tables: General tables. *J Am Statist Assoc* **89**: 1547–1557.

Carvalho FD, Dellaert N and Osório M (1994b). *Statistical disclosure in two-dimensional tables: Positive tables*. Report 9441/a, Econometric Institute, Erasmus University Rotterdam.

Cox LH (1995). Network models for complementary cell suppression. *J Am Statist Assoc* **90**: 1453–1462.

Domingo-Ferrer J and Torra V (eds) (2004). *Privacy in Statistical Databases*. Lecture Notes in Computer Science, Vol. 3050. Springer-Verlag: Berlin, Heidelberg.

Doyle P, Lane J, Theeuwes J and Zayatz L (eds) (2001). *Confidentiality, Disclosure and Data Access: Theory and Practical Applications for Statistical Agencies*. North-Holland: Amsterdam.

Fischetti M and Salazar JJ (1999). Models and algorithms for the 2-dimensional cell suppression problem in statistical disclosure control. *Math Program* **84**: 283–312.

Fischetti M and Salazar JJ (2000). Models and algorithms for optimizing cell suppression in tabular data with linear constraints. *J Am Statist Assoc* **95**: 916–928.

Fischetti M and Salazar JJ (2003). Partial cell suppression: A new methodology for statistical disclosure control. *Statist Comput* **13**: 13–21.

Gonzalez Jr J and Cox LH (2005). Software for tabular data protection. *Statist Med* **24**: 659–669.

Gusfield D (1988). A graph theoretic approach to statistical data security. *SIAM J Comput* **17**: 552–571.

ILOG (2002). *ILOG Cplex 8.0 User's Manual and Reference Manual*. ILOG SA. http://www.ilog.com.

Kelly J, Golden B and Assad A (1992). Cell suppression: Disclosure protection for sensitive tabular data. *Networks* **22**: 397–417.

Willenborg L and de Waal T (2001). *Elements of Statistical Disclosure Control*. Springer-Verlag: New York.

Received April 2006;
accepted December 2006 after one revision