Computational Issues Connected with the Protection of Sensitive Statistics by Auditing Sum-Queries

F. M. Malvestuto
Dipartimento di Scienze dell’Informazione
Università LA SAPIENZA di Roma
mal@dsi.uniroma1.it

M. Moscarini
Dipartimento di Scienze dell’Informazione
Università LA SAPIENZA di Roma
moscarini@dsi.uniroma1.it

Abstract

An implementation of the auditing strategy is presented to avoid both exact and approximate disclosure. The key data structure is a query map, which is a graphical summary of answered queries. Since the size of a query map may be exponential in the number of answered queries, a query-restriction criterion is introduced to make every query map a graph. An auditing procedure on such a graph is presented and the computational issues connected with its implementation are discussed. All the computational tasks can be carried out efficiently but one, which is a provably intractable problem.

1. Introduction

Query auditing [1, 2] is an effective strategy for guarding the confidentiality of individual data in a statistical database. Its existing implementations (e.g., the Audit Expert [2]) meet the minimal security requirement, which consists in avoiding exact disclosure of confidential attributes in individual records. This security requirement is not satisfactory because it does not take into account accurate estimates (“approximate disclosure”) [1, 14], and statistical publications usually ensure a higher security level, which consists in avoiding approximate disclosure too. Approximate disclosure occurs when, given a file \( \Omega \), a confidential numeric attribute \( \text{CONF} \) and a (value) query \( q \) of the type \( \text{SUM} (\text{CONF}) \) specified by a logical formula \( f \), the value \( t \) of \( q \), that is, \( t = \sum_{r \in R} r[\text{CONF}] \) where \( R \) is the set of records in \( \Omega \) qualified by \( f \), allows the value of \( \text{CONF} \) for some record in \( R \) to be estimated with an error less than a prefixed tolerance value. Queries such as \( q \) are said to be sensitive and their values are to be protected either by leaving them unanswered or by delivering suitable value intervals. Several criteria of sensitivity exist [4, 14] but we only assume the existence of a test which takes as input a set of records \( R \) and determines whether the statistic \( \pi_{\text{CONF}}(R) \) (without removing duplicates) is or is not sensitive. Approximate disclosure may also occur when the current query is not sensitive but its value, combined with the values of previously answered queries, allows a sensitive query to be implicitly answered.

Example 1. Consider the following fictitious table EMP containing records of people working in a hypothetical research institute. Each employee is described by three attributes: \( \text{E#} \) (the employee’s code), \( \text{DEP} \) (the department where the employee works) and \( \text{SALARY} \) (the employee’s salary).

<table>
<thead>
<tr>
<th>E#</th>
<th>DEPARTMENT</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>e1</td>
<td>Management</td>
<td>85</td>
</tr>
<tr>
<td>e2</td>
<td>Management</td>
<td>3</td>
</tr>
<tr>
<td>e3</td>
<td>Management</td>
<td>2</td>
</tr>
<tr>
<td>e4</td>
<td>Administration</td>
<td>4</td>
</tr>
<tr>
<td>e5</td>
<td>Administration</td>
<td>3</td>
</tr>
<tr>
<td>e6</td>
<td>Administration</td>
<td>3</td>
</tr>
<tr>
<td>e7</td>
<td>Services</td>
<td>4</td>
</tr>
<tr>
<td>e8</td>
<td>Services</td>
<td>3</td>
</tr>
<tr>
<td>e9</td>
<td>Services</td>
<td>3</td>
</tr>
<tr>
<td>e10</td>
<td>Marketing</td>
<td>50</td>
</tr>
<tr>
<td>e11</td>
<td>Marketing</td>
<td>30</td>
</tr>
<tr>
<td>e12</td>
<td>Marketing</td>
<td>20</td>
</tr>
</tbody>
</table>

Consider the following three SQL-like queries:

\[
q_1: \text{select } \text{SUM} (\text{SALARY}) \\
\text{from EMP} \\
\text{where } \text{DEP} = \text{Administration}
\]

\[
q_2: \text{select } \text{SUM} (\text{SALARY}) \\
\text{from EMP} \\
\text{where } \text{DEP} = \text{Services}
\]

\[
q_3: \text{select } \text{SUM} (\text{SALARY}) \\
\text{from EMP} \\
\text{where } \text{NOT DEP} = \text{Marketing}
\]

Let \( R_i \) be the query-set of \( R_i, i = 1, 2, 3 \). The statistics \( \pi_{\text{CONF}}(R_1) \), \( \pi_{\text{CONF}}(R_2) \) and \( \pi_{\text{CONF}}(R_3) \) are respectively:

\[(4,3,3), (4,3,3), (85,3,2,4,3,3,4,3,3).\]

Assume the sensitivity criterion in use is the “concentration rule” (2, 85 %), according to which a statistic is sensitive if there are two individual contributions whose sum constitutes more than the 85 % of the total.
Accordingly, no \(q_i\) is a sensitive, \(i = 1, 2, 3\). However, after answering \(q_1\) and \(q_2\), if the query system answered \(q_3\), then one could determine the value (90) of the query

\[
q^*; \text{ select SUM(SALARY) from EMP where DEP = Management}
\]

which is sensitive since the salaries of \(e_1\) and \(e_2\) sum up to 88 which is greater than the 85 \% of 90. Thus, if the employee coded by \(e_2\) were informed of the values of \(q_1\), \(q_2\) and \(q_3\), then he would get an accurate estimate (87) of the salary (85) of the employee \(e_1\).

Let \(q\) be the current query and assume that \(q\) is not sensitive. When deciding if \(q\) can be safely answered, the query-system must hypothesize a knowledgeable user (the “snooper”) who is able to disaggregate the values of answered queries to the least detail. Explicitly, such a user is assumed to have such a semantic competence that he guesses right which queries are overlapping and which ones are not, as well as such a computational competence that he knows how to find out all the aggregata data that are “inferable” from the values of answered queries. To repel the attacks of the snooper, the query-system will decide that \(q\) can be safely answered if and only if either

- the value of \(q\) is inferable from the values of previously answered queries, or
- there is no sensitive query among those whose values are inferable from the value of \(q\) and the values of previously answered queries.

The query-system succeeds in repelling the security attacks delivered by the snooper if its inference model is at least as powerful as the snooper's one. To this end, it must keep track of all previously answered queries by storing not only their answers but also the overlapping relationships among their query-sets. Based on these pieces of information, it will set up a system \(\Gamma\) of linear constraints which allows to identify every query whose value is inferable. If the underlying database is large, then the number of variables in \(\Gamma\) is always less (and is often far less) than the size of the database; however, it may be exponential in the number of previously answered queries, so that after answering a large number of queries, the query-system will spend a lot of time to decide if the current query can be safely answered. To overcome this difficulty, we introduce the following query-restriction criterion. Given a set \(Q\) of answered queries and a new nonsensitive query \(q\) with query-set \(R\), a necessary condition for \(q\) to be safely answerable is that either the value of \(q\) is inferable from \(Q\) or, for every two queries \(q_1\) and \(q_2\) from \(Q\), \(R \cap R_1 \cap R_2 \cap R\) be empty, where \(R_i\) is the query-set of \(q_i\) \((i = 1, 2)\). So, the number of variables in \(\Gamma\) is \(O(|Q|^2)\). The resulting auditing procedure is as follows

Input: A set \(Q\) of answered queries and a new query \(q\), all of the same type; the query-set \(R\) of \(q\).

Output: A Boolean variable \(safe\) which is \(\text{TRUE}\) if and only if \(q\) can be safely answered given the values of all the queries in \(Q\).

(Initialization) Set \(safe := \text{FALSE}\).

(Phase I) If the value of \(q\) is inferable from the answers to the queries in \(Q\), then set \(safe := \text{TRUE}\) and Exit.

(Phase II) If \(q\) violates the query-overlap constraint, then Exit.

(Phase III) If no query whose value is inferable from \(Q \cup \{q\}\) is sensitive, then set \(safe := \text{TRUE}\). Exit.

In this paper, the computational issues connected with the implementation of the auditing procedure above are discussed; explicitly, we show (but all proofs are omitted) that the problems related to Phases I and II can be solved efficiently, and the problem related to Phase III is provably intractable.

2. The Inference Model

Consider a (conceptual) file \(\Omega\) of individual records and let \(U\) be the set of the category (i.e., nonconfidential) attributes used to make queries of the type \(\text{SUM(CONF)}\) where \(\text{CONF}\) is a confidential numeric attribute. A subset of \(\Omega\) is \(\text{inquirable}\) using \(U\) (\(U\)-inquirable, for short) if it is the result of the interpretation in \(\Omega\) of some logical formula involving attributes in \(U\). A query of the type \(\text{SUM(CONF)}\) is specified by a logical formula \(f\), called the characteristic formula of query \(q\). The query-set of \(q\) is the subset of \(\Omega\) qualified by \(f\), and the value of \(q\) is the sum of the values of \(\text{CONF}\) over the records in the query-set of \(q\). If the query-set of \(q\) is empty, then the value of \(q\) is taken to be zero. Note that empty query-sets occur either by chance or by effect of semantic constraints. Let us consider a set of answered queries \(Q = \{q_1, \ldots, q_n\}\) of the type \(\text{SUM(CONF)}\) and look for all the information that is explicitly or implicitly released by the query system when answering the queries in \(Q\). Let \(R_v\) be the query-set of \(q_v\), \(v = 1, \ldots, n\). The coarsest partition \(P\) of \(\Omega\) such that each \(R_v\) can be recovered by taking the union of classes of \(P\) will be referred to as the partition of \(\Omega\) induced by \(Q\). A class of \(P\) is each nonempty set defined by any set expression such as \(\cap_v \rho_v\), where \(\rho_v\) can be either \(R_v\) or \(R_v := (\cap_j \rho_j)\), and will be indexed by \(e = \{v: \rho_v = R_v\}\). Note that each class \(P_e\) of \(P\) is \(U\)-inquirable since it is qualified by the formula \(g_e = \land_v \varphi_v\), where \(\varphi_v\) is either \(f_v\) or \(\neg f_v\) depending on whether \(v\) is or is not in \(e\). Moreover, if \(\cap_v \rho_v\) is a class of \(P\) then it is indexed by the empty set and will be referred to as the external class of \(P\); the remaining classes of \(P\) are said to be internal. Let \(V = \{1, \ldots, n\}\) and let \(E\) be the set of the indices of internal classes of \(P\). By the the map of \(Q\) we mean the hypergraph \(H(Q) = (V, E)\), where each edge \(e\) is labelled by the formula
Given the map $H(Q)$, the query-system can control the explicit and implicit information conveyed by $Q$ as follows. For each edge $e$ of $H(Q)$, a variable $x_e$ standing for the value $(d_e)$ of $\text{SUM} (\text{CONF})$ over the records in $P_e$ is introduced. Data additivity then allows the following equation system to be set up

$$\sum_{e: v \in e} x_e = \sum_{e: v \in e} d_e \quad (\text{for each } v \in V)$$

Note that the coefficient matrix of the above equation system is nothing more than the incidence matrix of $H(Q)$, say $A$, and the constant terms are the values $t_v$. Thus, we write the equation system above in the matrix form

$$A \ x = t . \quad \text{(1)}$$

By $\Gamma$ we denote the system of linear constraints composed of equation system (1) and possibly (when $\text{CONF}$ is of nonnegative type) of the $|E|$ nonnegativity constraints $x_e \geq 0$. We say that an edge set $S$ in $H(Q)$ is unprotected if, for every feasible solution $a$ of $\Gamma$, one has

$$\sum_{e \in S} a_e = \sum_{e \in S} d_e ,$$

and protected, otherwise.

Checking recoverability is an easy task which can be carried out as follows.

**Algorithm 2.1**

Input: $H(Q)$, the characteristic formula $f$ of $q$, the query-set $R$ of $q$, and the size $r$ of $R$.

Output: A Boolean variable recoverable whose value is true if and only if the recoverability property holds; in this case, the set variable $S$ makes condition (i) true.

1. recoverable := false; $S := \emptyset$.
2. For each edge $e$ of $H(Q)$

   determine the subset $R'$ of $R$ qualified by $g_e$ and let $r' = |R'|$.
   if $r' > 0$ then
     if $p_e > r'$ then Exit
     else set $r := r-r'$ and $S := S \cup \{e\}$.
   end if
3. If $r = 0$ then set recoverable := true. Exit.

We now state a characterisation of unprotected edge sets by distinguishing the two cases of real and nonnegative data type. To achieve this, we need to introduce the notion of an “algebraic” edge set [8, 10]. Consider a subset $S$ of $E$; the incidence vector of $S$ is the binary vector $s$ with $s_e = 1$ if and only if $e$ is in $S$. $S$ is algebraic in $H(Q)$ if $s$ is a linear combination of rows of $A = (a_v)_{v \in V}$, that is, if the equation system

$$\sum_v \lambda_v a_v = s \quad \text{(2)}$$

has a solution $(\lambda_v)_{v \in V}$. Note that every algebraic edge set is unprotected. It is well-known [8, 9] that checking algebraicity requires a time polynomial in the size of the equation system (2).

Data of real type. In this case, $\Gamma$ reduces to equation system (1). Then,

**Proposition 1.** An edge set is unprotected if and only if it is algebraic.

So, by Proposition 1, one can test if an edge set is unprotected in polynomial time.

Data of nonnegative type. In this case, $\Gamma$ consists of equation system (1) and nonnegativity constraints. Let $Z$ be the set of (unprotected) edges $e$ of $H(Q)$ such that $x_e$ is zero for every feasible solution of $\Gamma$. The set $Z$ and the hypergraph $H(Q) - Z$ will be referred to as the zero-invariant of $H(Q)$ and the reduced map of $Q$, respectively.
Proposition 2. Let $Z$ be the zero-invariant of $H(Q)$. An edge set $S$ in $H(Q)$ is unprotected if and only if the edge set $S-Z$ is algebraic in $H(Q)-Z$.

The zero-invariant $Z$ of $H(Q)$ can be determined as follows. Let $E_0 = \{e \in E : d_e = 0\}$. For each edge $e$ in $E_0$, maximize $x_e$ subject to $\Gamma$ and join $e$ to $Z$ if the maximum of $x_e$ is zero. Once $Z$ has been determined, by Proposition 2, checking if an edge set is unprotected in $H(Q)$ requires a polynomial time. To sum up, given $H(Q)$ and a query $q$, we have an efficient procedure for deciding if the value of $q$ is in $Q$. Unfortunately, the input size, which is proportional to the number of edges of $H(Q)$, may be exponential to the size of $Q$, which would vanish the efficiency of the test for unprotectedness. Thus, in Section 4 we shall introduce a query-restriction which makes every query map a graph where self-loops may occur.

3. Algebraic Edge Sets

As we saw in the previous section, the property of algebraicity for an edge set in a query map plays a key role in the inference model used by the query-system to control the amount of information released in the query-answering process. In this section, we state some properties of algebraic edge sets. First of all, the family of algebraic edge sets is closed under disjoint union and proper difference; moreover, if the whole edge set is algebraic, then the family of algebraic edge sets is also closed under complementation. As a consequence of the first two properties, one has that every nonempty algebraic edge set is the disjoint union of minimal (with respect to set inclusion) nonempty algebraic edge sets. A further useful property can be stated by considering the incidence matrix $A$ of query map $H(Q)$ and the vector space $K(A)$ of the solutions of the homogeneous equation system $A\ y = 0$ associated to (1). Since an edge set is algebraic if and only if its incidence vector is orthogonal to $K(A)$, if we are given a vector basis of $K(A)$, then we can decide if an edge set is algebraic simply by checking that the scalar product of its incidence vector with each basis vector of $K(A)$ is zero. Now, a basis of $K(A)$ can be found as follows. Let $m$ be the number of columns of $A$ and $r$ its rank.

ALGORITHM 3.1

1. Select a set $B$ of $r$ edges of $H(Q)$ corresponding to columns of $A$ that are linearly independent.

2. Let $N = E-B$. Cast the equation system $A\ y = 0$ in the canonical form [5]:

$$y_e + \sum_{e' \in N} c_{e,e'} y_{e'} = 0 \quad (e \in B)$$

(3) For each edge $e$ in $N$, take the vector $y$ with: $y_e = 1$ and for each $e' \neq e$, $y_{e'} = 0$ if $e'$ is in $N$, and $y_{e'} = +e_{e'}$ if $e'$ is in $B$.

The $m-r$ vectors obtained at Step (3) form a basis of $K(A)$, called the basis associated to $B$. Sets such as $B$ are called basic edge sets of $H(Q)$.

Proposition 3. Every nonempty algebraic set in $H(Q)$ has a nonempty intersection with each basic edge set of $H(Q)$.

It is convenient to re-state Proposition 3 in the matroid-theoretic language in view of its application to the problem of recognising unprotected edge sets. Some basic matroid-theoretic notions are now recalled [13]. Let $M(E)$ be the matroid on the edge set $E$ of $H(Q)$, where a set $S$ of edges of $H(Q)$ is taken to be “independent” (or “dependent”) if the columns of $A$ corresponding to the edges in $S$ are linearly independent (dependent, respectively). Maximal independent sets are the bases of $M(E)$, minimal dependent sets are the circuits of $M(E)$, and minimal edge sets having a nonempty intersection with each base of $M(E)$ are called the cocircuits of $M(E)$. Thus, the bases of $M(E)$ are exactly the basic edge sets of $H(Q)$ and, by Proposition 3, every nonempty algebraic edge set is a (proper or improper) superset of some cocircuit of $M(E)$. Consider now the basis vectors of $K(A)$ determined at Step (3) of ALGORITHM 3.1 and, for a given edge $e$ in $N$, let $y$ be the basis vector determined by taking $y_e = 1$. The support of $y$ (that is, the edge set $\{e \in E : y_e \neq 0\}$) is a circuit of $M(E)$. The $[N] = m-r$ circuits thus obtained are called the fundamental circuits associated to $B$ and can be “directed” as follows. If $C$ is such a circuit, then each edge $e$ in $C$ is labelled by “+” (or “−”) if $y_e > 0$ ($y_e < 0$, respectively). Finally, the fundamental cocircuits associated to $B$ are defined as the $[B] = r$ cocircuits each one of which contains one and only one edge which is in $B$. Explicitly, if $e$ is an edge in $B$, the fundamental cocircuit corresponding to $e$ is composed of $e$ and those edges not in $B$ which when added to $B$ lead to a fundamental circuit containing $e$.

As we said at the end of Section 2, from the next section onwards we shall assume that $H(Q)$ is a graph. Thus, before closing this section, we first review some known results on the matroid $M(E)$ when $E$ is the edge set of a graph, and then we state some results which provide an insight into minimal algebraic edge sets of a graph. Without loss of generality, we assume that $H(Q)$ is connected. In this case, it is well-known [5: Theorem 1, p. 421] that the rank of $A$ is $r = n-p$, where $n$ is the number of vertices of $H(Q)$, and $p = 1$ if $H(Q)$ is bipartite (that is, if $H(Q)$ contains no odd cycles) and 0 otherwise. Moreover, the independent sets of the matroid $M(E)$ are the $L$-forests [3] of $H(Q)$, an $L$-forest being a graph obtained from a forest by adding at most one edge $e$ to each component of
the forest, and the cycle containing $e$ is odd. The circuits of $M(E)$ are the even cycles and the $L$-odd sets [3] of $H(Q)$, an $L$-odd set being a pair of two odd cycles joined by a (possibly degenerate) simple path. The cocircuits of $M(E)$ are the minimal $C$-$B$ sets [3] of $H(Q)$, a $C$-$B$ set being an edge set whose removal creates one more bipartite component. So, if $H(Q)$ is bipartite, then the family of bases of $M(E)$ is the family of spanning trees of $H(Q)$, the family of circuits of $M(Q)$ is the family of even cycles of $H(Q)$, and the family of cocircuits of $M(Q)$ is the family of bonds (i.e., minimal edge cuts) of $H(Q)$.

Example 2. Consider the bipartite graph shown in Figure 2.

![Figure 2](image1)

Figures 3 and 4 show a spanning tree and two even cycles which are the fundamental circuits associated to it, respectively.

![Figure 3](image2)

![Figure 4](image3)

Figure 5 shows two bonds which are the fundamental cocircuits associated to the spanning tree of Figure 3 and corresponding to the tree-edges (3,4) and (4,5), respectively.

![Figure 5](image4)

Example 3. Consider the nonbipartite graph shown in Figure 6.

![Figure 6](image5)

Figures 7 and 8 show a base (i.e. a maximal $L$-tree) and the three fundamental circuits (one $L$-odd set and two even cycles) associated to it, respectively.

![Figure 7](image6)

![Figure 8](image7)
Figure 8 shows two minimal C-B sets which are the fundamental cocircuits corresponding to the edges (1,3) and (1,4), respectively.

Figure 9 shows two minimal C-B sets which are the fundamental cocircuits corresponding to the edges (1,3) and (1,4), respectively.

As to minimal algebraic edge sets of \( H(Q) \), we have the following. If \( H(Q) \) is bipartite, then

(i) every minimal algebraic edge set is a cocircuit of \( M(E) \), i.e. a bond of \( H(Q) \), and

(ii) a cocircuit of \( M(E) \), i.e. a bond of \( H(Q) \), is algebraic if and only if it is a "simple" [8] (or "bipartite" [7]) bond.

Figure 10 shows what a simple bond in a bipartite graph is like. Note that every star is a simple bond. From the foregoing it follows that an edge is (an) algebraic (cocircuit) if and only if it is a bridge.

If \( H(Q) \) is not bipartite, then it is quite another thing for a minimal algebraic edge set may contain many cocircuits and need not be the disjoint union of cocircuits.

Example 4. Consider the nonbipartite graph shown in Figure 11. The edge set \{ (1,2), (2,3), (3,4) \} is a minimal algebraic edge set and includes the three cocircuits \{ (1,2), (2,3) \}, \{ (1,2), (3,4) \} and \{ (2,3), (3,4) \}.

However, with the following definition of a "simple cocircuit" for an arbitrary graph we can generalize property (ii) above and have that a cocircuit is algebraic if and only if it is simple. Let \( H(Q) = (V, E) \) be a connected graph and let \( C \) be a cocircuit of \( M(E) \). If \( H(Q) \) is bipartite, then \( C \) is simple if \( C \) is a simple bond. If \( H(Q) \) is not bipartite, then we distinguish two cases depending on whether the graph \( H' \) obtained from \( H(Q) \) by deleting all edges in \( C \) is or is not connected. In the former case, we say that \( C \) is simple if
— the endpoints of the edges in \( C \) are all on the same side of \( H' \), and
— if \( C \) contains a self-loop, then each edge in \( C \) is a self-loop.

In this case, a simple circuit is like those shown in Figure 12.

In the latter case, let \( B \) be the bipartite component of \( H' \). We say that \( C \) is simple if
— the endpoints of the edges in \( C \) that are vertices of \( B \) are all on the same side of \( B \), and
— the edges (if any) in $C$ with both endpoints in $B$ are all self-loops.

In this case, a simple circuit is like those shown in Figure 13.

![Figure 13](image)

4. The Query-Overlap Constraint

As noticed at the end of Section 2, the number of edges of the map of $Q$ may be exponential in the size of $Q$. To overcome this difficulty, we introduce the following query-restriction criterion: given $Q$ and a nonsensitive query $q$ with query-set $R$, a necessary condition for $q$ to be safely answerable is that either the value of $q$ be inferable from $Q$ or, for every two queries $q_1$ and $q_2$ from $Q$, $R_1 \cap R_2 \cap R$ be empty, where $R_i$ is the query-set of $q_i$ ($i = 1, 2$). As a result, we have that $H(Q)$ is always a graph, and hence that the number of its edges is $O(|Q|^2)$. A further advantage is that the problem of recognising algebraic edge sets in $H(Q)$ can be optimally solved. To prove it, we exhibit a linear algorithm which is the generalization of that used in [10, 12] for bipartite graphs, and is based on the fact that equation system (2) has at most $\infty^1$ solutions. Given a connected graph $H(Q) = (V, E)$ with $|V| = n$ and $|E| = m$, a vertex $s$ and a subset $S$ of $E$, the following algorithm determines in linear time if $S$ is algebraic. There, the edges of $H(Q)$ that are not self-loops are called links.

**ALGORITHM 4.1**

1. $T := \emptyset$; $\exp(s) := \lambda$.

2. Start a DFS traversal of $H(Q)$ at vertex $s$. During the traversal of $H(Q)$, if vertex $v$ is first visited using link $e = (u, v)$, then set $T := T \cup \{e\}$ and $\exp(v) := \exp(u) + s_e$.

3. Set up the system $\Lambda$ of $m-n+1$ equations in the unknown $\lambda$, as follows. For each edge $e$ in $E-T$,

   — if $e$ is a link, say $(u, v)$, then add to $\Lambda$ the equation $\exp(u) + \exp(v) = s_e$;

   — if $e$ is a self-loop, say $e = (v, v)$, then add to $\Lambda$ the equation $\exp(v) = s_e$.

4. Conclude that $S$ is algebraic if and only if $\Lambda$ is consistent.

Finally, we address the problem of determining the zero-invariant $Z$ of $H(Q)$ when data is of nonnegativity type. Let $E_o = \{e \in E: d_e = 0\}$. If $H(Q)$ is bipartite, then $Z$ can be determined in an optimal way [6]. If not, instead of solving $|E_o|$ linear-programming problems (i.e., max $x_e$), we can apply the following algorithm which has as input $E_o$ and a system of fundamental circuits, for which the following notion of “traversability” is used. Let $S$ be an edge set; a circuit $C$ is $S$-traversable if every two edges in $S \cap C$ are labelled by the same sign.

**ALGORITHM 4.2**

1. Set $Z := E_o$.

2. Repetedally apply the following operation until $Z$ cannot be further reduced: for each fundamental circuit $C$ that is $Z$-traversable, set $Z := Z-C$.

Example 5. Suppose we are given the following query map, where for each edge $e$ the value $d_e$ is reported.

![Figure 14](image)
5. The Implementation

Given a safe sequence of answered queries \( q_1, \ldots, q_n \)
and a new nonsensitive query \( q \), our implementation of the auditing procedure makes use of a graph \( G \) which is the
(possibly) reduced map of a suitable subset of \( \{q_1, \ldots, q_n\} \). More precisely, let \( V \) be the subset of \( \{1, \ldots, n\} \) such that \( v, 1 \leq v \leq n, \) belongs to \( V \) if and only if the value of \( q_v \) is not inferable from the queries answered before \( g_v \). Let \( Q = \{q_v: v \in V\} \) and let \( G = (V, E) \) be the (possibly reduced) map of \( Q \), where an edge \( e \) is “marked” if \( d_e \) is the total of a sensitive statistic. Given \( G \), the query-set \( R \) of \( q \), the characteristic formula \( f(q) \), the value \( t \) of \( q \) and the size \( r \) of \( R \), our auditing procedure works as follows.

With the first five steps, the query-system decides if the value of \( q \) is inferable from \( Q \), and, if not, if \( q \) violates the query-overlap constraint. To this end, the edges \( e \) of \( G \) such that \( P_e \cap R \neq \emptyset \) are identified and grouped in two sets: \( S \) and \( S' \). Set \( S \) contains the edges for which \( P_e \subseteq R \) and set \( S' \) contains the edges for which \( P_e \not\subseteq R \). Thus, the value of \( q \) is inferable from \( Q \) if and only if \( S' = \emptyset \). \( R = \cup_{e \in S} P_e \) and \( S \) is an algebraic edge set in \( G \). If not, then \( q \) violates the query-overlap constraint if and only if \( S \cup S' \) contains at least one link.

(1) A new graph \( G^* = (V^*, E^*) \) is constructed with \( V^* = V \cup \{n+1\} \) and \( E^* = E \cup R \) if \( q \) is insensitive from \( Q \).

(2) For each edge \( e \) in \( E \)

- determine the subset \( R' \) of \( R \) qualified by \( g_e \);
  - let \( r' \) be the size of \( R' \) and \( t' \) be the total of the statistic \( \pi_{CONF}(R') \):
    - if \( r' > 0 \), then set \( g^*_e := g_e \);
    - otherwise, set \( R := R - R' \); \( r := r - r' \); \( t := t - t' \);
    - if \( p_e = r' \) then
      - add \( e \) to \( S' \);
    - if \( e \) is a self-loop \( v \), then
      - add \( e = (v, n+1) \) to \( E^* \);

(3) If \( S' = \emptyset \) and \( r = 0 \) and \( S \) is an algebraic edge set in \( G \) then conclude the value of \( q \) is inferable from \( Q \). Exit.

(4) If \( S \cup S' \) contains at least one link, then conclude \( q \) violates the query-overlap constraint. Exit.

(5) If \( S \cup S' \) contains at least one link, then conclude \( q \) violates the query-overlap constraint. Exit.

(6) If \( r > 0 \), then

- add the self-loop \( (n+1, n+1) \) to \( E^* \);
- set \( g^*_{(n+1,n+1)} := f \wedge (v \in S \cup S', g_e) \);
- set \( p^*_{(n+1,n+1)} := r \);
- determine the subset \( T \) of \( S \) qualified by \( g^*_{e} \);
- set \( t := t - t' \);
- add the edge \( (n+1, n+1) \) of \( G^* \) if \( \pi_{CONF}(R) \) is sensitive.

At this point, \( G^* \) is the map of \( Q \cup \{q\} \). The next step is performed only in the case of nonnegative data and makes \( G^* \) the reduced map of \( Q \cup \{q\} \).

(6) Let \( E_0 \) be the set \( \{e \in E^* : d^*_e = 0\} \). Determine the zero-invariant of \( Z = (G^* - E_0) \) and delete \( Z \) from \( E^* \).

Finally, the next three steps are performed to decide if there is a sensitive query whose value is inferable from \( Q \cup \{q\} \). To achieve this, we need the following mapping \( \tau \) from \( E \) to the powerset of \( E \). For each edge \( e \in S \cup S' \), \( \tau(e) = \{e\} \).

For each self-loop \( e = (v, v) \) in \( S \cup S' \), if \( e \) is in \( S \) then \( \tau(e) = \{(v, n+1)\} \); otherwise, \( \tau(e) = \{(v, v), (v, n+1)\} \). Accordingly, the image of a subset \( A \) of \( E \) is taken to be \( \cup_{e \in A} \tau(e) \).

(7) If no edge of \( G^* \) is marked, then go to (9).

(8) For each marked edge \( e \) of \( G^* \),
for each minimal algebraic edge set \( A \) in \( G^* \) that contains \( e \) and is the image of no algebraic edge set in \( G \):

determine the set \( R' \) of records qualified by the formula \( \forall e \in A \, g^* e \):

if \( \pi_{CONF}(R') \) is sensitive, then conclude that \( q \) cannot be safely answered and Exit.

(9) Conclude that \( q \) can be safely answered and set \( G := G^* \). Exit.

The following is an illustrative example.

Example 6. Consider queries of the type \( \text{SUM}(\text{CONF}) \), where \( \text{CONF} \) is of real type, with characteristic formulas involving the only attribute \( \text{DEP} \) which has six values \( 1, 2, 3, 4, 5, 6 \). Assume that there is at least one employee in each department and the only sensitive query is that having the characteristic formula \( \text{DEP}=6 \). Consider the sequence of six queries \( q_1, q_2, q_3, q_4, q_5, q_6 \) whose characteristic formulas are:

\[
\begin{align*}
f_1 &= \text{DEP}=1 \lor \text{DEP}=2 \lor \text{DEP}=6, \\
f_2 &= \text{DEP}=2 \lor \text{DEP}=3, \\
f_3 &= \text{DEP}=3 \lor \text{DEP}=4, \\
f_4 &= \neg \text{DEP}=5, \\
f_5 &= \text{DEP}=1 \lor \text{DEP}=4 \lor \text{DEP}=5, \\
f_6 &= \text{DEP}=5 \lor \text{DEP}=6.
\end{align*}
\]

The queries \( q_1, q_2 \) and \( q_3 \) are recognised as being safely answerable. Figure 16 shows the map of \( \{q_1, q_2, q_3\} \) and \( \{q_1, q_2, q_3\} \).

The value of query \( q_4 \) is inferable from \( \{q_1, q_2, q_3\} \) for the query-set of \( q_4 \) is recoverable by taking the union of the internal classes of the partition induced by \( \{q_1, q_2, q_3\} \), and the whole edge set of the map of \( \{q_1, q_2, q_3\} \) is algebraic (it is the disjoint union of the stars of vertices 1 and 3). The value of query \( q_5 \) is not inferable from \( \{q_1, q_2, q_3\} \); the map of \( \{q_1, q_2, q_3, q_5\} \), shown in Figure 17, contains one sensitive edge, the self-loop \((1,1)\). The minimal algebraic edge sets containing \((1,1)\) are \((1,1), (1,2), (1,5), (1,6)\). For each minimal algebraic edge set \( A \) in \( G^* \) that contains \( e \) and is the image of no algebraic edge set in \( G \):

The correctness of our implementation stems from the following facts. For convenience, we say that an edge set \( A \) is **sensitive** if the statistic \( \pi_{CONF}(R') \) is sensitive, where \( R' \) is the set of records qualified by the formula \( \forall e \in A \, g^* e \). Then, one has that: (1) every formal sensitivity definition refers to a subadditive set function and a statistic is sensitive if the corresponding value of the sensitivity function is (strictly) positive \([4]\), and (2) every algebraic edge set in \( G^* \) that is the image of an algebraic edge set in \( G \) is not sensitive. From fact (1) it follows that: (1.1) a necessary and sufficient condition for a query \( q \) to be safely answerable given a safe set \( Q \) of answered queries is that no minimal algebraic edge set in \( G^* \) be sensitive, and (1.2) if a minimal algebraic edge set in \( G^* \) is sensitive then it contains at least one marked edge.

Unfortunately, the complexity of the problem related to Step 8 is not polynomial since the number of the minimal algebraic edge sets in \( G^* \) that contain a given marked edge and are the images of no algebraic edge sets in \( G \) may be exponential as shown in the following example.

Example 7. Let \( G^* \) be the graph shown in Figure 19. Let \( A \) be the family of the edge sets that are obtained by taking the union of \( \{(u,v), (u',v')\} \) with \( k \) edges of the form

```
  o
 /|
/ \\
/  \\
/   \\
/     \\
/       \\
\     \\
\   \\
\  \\
\ o
```

one for each column. It is easily seen that (1) $|A|$ is an exponential number, (2) each edge set in $A$ is a minimal algebraic edge set, and (3) each edge set in $A$ contains the marked edge $(u,v)$. Moreover, if $u$ represents the new query and $G$ is the (reduced) map of the set of remaining queries, then no edge set in $G$ is the image of an algebraic edge set in $A$ and, therefore, the sensitivity test will be executed an exponential number of times.

### 6. Conclusions

We have proposed an implementation of the auditing strategy for sum-queries restricted according a query-set-overlap control. The data structure used in the implementation is a graph with labelled and weighted edges (the “query map”), whose unprotected edge sets correspond to the data that are explicitly and implicitly released by the query-system when queries are answered. Efficient algorithms were provided for some tasks; however, the problem of detecting the existence of a sensitive minimal algebraic edge set in the updated query map is provably intractable. So, one could be content with some less stringent security requirement [4, 6, 14]; for example, every sensitive edge of a query map must be protected. Then, we can efficiently solve the above-mentioned detection problem. The same procedure can also be applied to the case in which the query-system has to decide on the safety of a single query asking for not a value but a sum table $T$ with both cells and marginal totals. If sensitive sums can occur in cells only, then several heuristics can be adopted by choosing some ordering of the entries in $T$; for example, the first are the sensitive cells, then the marginal totals come up and, the last are the remaining cells (possibly ordered by nonincreasing values). Note that the query maps that are constructed when nonsensitive cells are examined are not graphs, but they can be reduced to graphs as follows: when the value of a (nonsensitive) cell is released, the query map is not augmented with a new cell-vertex but the labels of two marginal-vertices are updated and the edge joining them is deleted.

### References


[9] F.M. Malvestuto and M. Moscarini, Query evaluability in statistical databases, IEEE Transactions on Knowledge and Data Engineering 2 (1990), 425-430.


