Inference of Three-Way Table Entries From Two-Dimensional Projections

Stephen F. Roehrig (roehrig@andrew.cmu.edu)
George T. Duncan (gd17@andrew.cmu.edu)
Ramayya Krishnan (rk2x@andrew.cmu.edu)
Rema Padman (rpadman@andrew.cmu.edu)

The H. John Heinz III School of Public Policy and Management
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Abstract

Multi-dimensional tables of sensitive information are often summarized and made public by means of lower-dimensional projections, which are intended to prevent any disclosure of confidential data. Multiple projections of the same underlying table are linked over common attributes, however, so there is concern about the possibility of recovering sensitive data by combining projections. In this paper, we present an algorithm that gives tight upper and lower bounds on cell values of three-dimensional data when the three two-dimensional projections of that data are available.

1 Introduction

An interesting and important version of the statistical database disclosure problem is to consider how accurately the cell values of a three-dimensional table of microdata can be inferred from the release of its two-dimensional projections [12, pp. 108, 130]. If the sensitivity of a datum results from the fact that it links three attributes, and knowledge of the connection between any two of those attributes is itself not a disclosure, we can ask to what extent the triple is compromised by knowledge of the corresponding projections.

This is a problem of linked tables, since each of the three two-dimensional projections contains some portion of the total information content of the underlying three-dimensional table. None of the projections, by itself, may be sufficiently revealing to warrant protective measures, such as cell suppression, noise addition, or some other form of masking. However, examples (such as that in [12, p. 130] and others presented here) show that under some circumstances it is possible to take advantage of the linked nature of the two-dimensional tables to infer either the exact values, or very precise estimates, of cells in the three-dimensional table.

To illustrate the problem of linked tables, consider the example in Figure 1. These tables present data about self-employed shopkeepers in three towns, broken out by race and income level. Each table, by itself, is free from sensitive disclosure, in that no individual shopkeeper can be positively associated with any particular attribute value. However, when the three tables are taken together, a number of potentially revealing conclusions may be drawn. For example, we show later that these data imply that no Black shopkeeper in either Greene or Belmont has income over $250K.

In this paper we present a new algorithm for determining bounds on cells of a confidential three-dimensional table given its three two-dimensional projections. These bounds are tight, and are an improvement over previous methods such as Frechet bounds [8, 10, 5] and the Buzzigoli-Giusti algorithm [5]. These earlier methods do provide bounds which are computed from the actual data (as opposed to being determined only by the structure of the data tables themselves [6]), but they can be shown to sometimes produce bounds which are looser than the optimal bounds our method provides. As a by-product, our analysis yields simple conditions for exact disclosure, that is, the case where the upper and lower bounds on a cell entry are equal.
2 Problem Formulation

To model the disclosure problem, let

\[ T = [x_{ijk}], \quad x_{ijk} \geq 0, \ i = 1 \ldots I, \ j = 1 \ldots J, \ k = 1 \ldots K \]

represent the underlying sensitive three-dimensional datafile. Suppose that the three two-dimensional projections

\[ A_{ij} = \sum_{k=1}^{K} x_{ijk} \]  
\[ B_{ik} = \sum_{j=1}^{J} x_{ijk} \]  
\[ C_{jk} = \sum_{i=1}^{I} x_{ijk} \]

are released. By using these known projections, it is easy to construct a pair of integer programs to determine upper and lower bounds on any cell in the underlying table \( T \) (Willeborg and de Waal 1996). For bounds on \( x_{ijk} \), simply solve:

\[
\begin{align*}
\text{max} / \text{min} & \quad x_{ijk} \\
\text{subject to} & \quad \sum_{i} x_{ijk} = A_{ij}, \quad i \in I, j \in J \\
& \quad \sum_{j} x_{ijk} = B_{ik}, \quad i \in I, k \in K \\
& \quad \sum_{k} x_{ijk} = C_{jk}, \quad j \in J, k \in K \\
& \quad x_{ijk} \in \mathbb{Z}^+ 
\end{align*}
\]

However, for even moderate size problems, this integer programming approach is too computationally intensive to be of practical use. One might suspect that solutions to an LP relaxation of this problem will be integral, because of successes in applying network techniques to related problems. This suspicion turns out to be true, but a direct argument involving known guarantees of integrality (such as total unimodularity) appears to be difficult. Further, even though the LP relaxation is far less computationally intensive than the IP, it still requires more computation than is practical, especially for large tables and the dynamic databases that are increasingly available via the Internet. This motivates an attempt to find a specialized algorithm to solve this disclosure problem.

Following the lead of Chowdhury et al. [1], we present a network formulation, and then show, by means of a technique similar to the Ford and Fulkerson maximal flow algorithm [7], that the maxima and minima of the \( x_{ijk} \) are integer. This technique yields a fast algorithm for finding the extrema.

3 A Network Representation

Figure 2 shows a network for a \( 2 \times 2 \times 2 \) problem. The two-dimensional projections are represented by nodes \( A_{ij}, B_{ik}, C_{jk} \). The three-dimensional microdata cell values, which we will denote by \( x_{ijk} \) in the remainder of the paper, are more succinctly represented in the figure by nodes labeled \( ijk \). In this graph there are two subgraphs, labeled \( \cal{I}_1 \) and \( \cal{I}_2 \), which are linked together by the nodes \( C_{jk} \). In general, (that is, in cases larger than \( 2 \times 2 \times 2 \)) there will be \( I \) such subgraphs. By the way the graph decomposes, there will be \( JK \) central nodes \( C_{jk} \), each linked by an arc to each of \( I \) subgraphs. Within a subgraph \( i \in I \), the nodes \( x_{ijk} \) are also connected to nodes

\[ A_{ij}, \quad j \in 1 \ldots J, \quad \text{and} \]
\[ B_{ik}, \quad k \in 1 \ldots K. \]

Note, however, that within a subgraph (that is, apart from arcs joining it to nodes \( C \)), each \( x_{ijk} \) is connected to exactly two such nodes, \( A_{ij} \) and \( B_{ik} \).

This network is a valid representation of a three-dimensional table with the given projections, provided it is consistent according to the following definition.

**Definition 1** A network representation of the three-dimensional disclosure problem is consistent if

1. the values of \( A_{ij}, B_{ik}, C_{jk} \) equal the projection table entries.

---

<table>
<thead>
<tr>
<th>Race</th>
<th>Income</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>White</td>
<td>Chinese</td>
</tr>
<tr>
<td>Total</td>
<td>19</td>
<td>26</td>
</tr>
</tbody>
</table>

Figure 1: Race, Income, and Location of Self-Employed Shopkeepers

<table>
<thead>
<tr>
<th>Race</th>
<th>Greene</th>
<th>Belmont</th>
<th>Oliver</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>12</td>
<td>3</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Chinese</td>
<td>5</td>
<td>15</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>White</td>
<td>4</td>
<td>9</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Race</th>
<th>Greene</th>
<th>Belmont</th>
<th>Oliver</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>21</td>
<td>27</td>
<td>25</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Cross-tabulation of characteristics of self-employed shopkeepers
2. the values of the $x_{ijk}$ are non-negative, and
3. each projection node (e.g., $A_{ij}$) is the sum of all the $x_{ijk}$ directly linked to it by an arc.

These considerations suggest that, starting with the known microdata values, we can modify the graph incrementally, leaving the projection values unchanged, in an attempt to maximize or minimize a particular $x_{ijk}$. Any modification satisfying the three conditions above represents a consistent set of values of the sensitive data. Upper and lower bounds on a microdata cell are, from this perspective, just the largest and smallest values obtainable over the collection of all consistent sets.

We can also view the disclosure problem as a network flow problem. Unfortunately, this view includes side conditions that make it difficult to solve with standard algorithms. To see this, first add to Figure 2 a source node $S$ connected by directed arcs outgoing from it to each microdata node $x_{ijk}$, and a sink node $T$ with incoming arcs from each of the projection nodes $A_{ij}$, $B_{ik}$, $C_{jk}$. Convert each arc to a directed arc pointing from a microdata node to a projection node. Finally, add an arc from $T$ to $S$. Then the upper bound problem for node $x_{ijk}$, for example, is a maximal flow problem on the augmented graph, subject to the following conditions:

1. the flow on each arc from a projection node to $T$ must be equal to the corresponding value in the projection table,
2. the flow on the return arc from $T$ to $S$ carries a constant flow equal to three times the grand total of the three-dimensional table, and
3. for each node $x_{ijk}$, the flow along all the arcs leaving it are equal.

The first condition simply says that the flow must be consistent with the projection data. The second says that, because each of the three two-dimensional projections sums to the grand total of the microdata table, the total flow in the entire network must be three times that grand total. The third condition reflects the fact that each microdata node contributes the same amount to each projection, this amount being its present value. This last requirement imposes constraints which are not part of the standard network flow model, and makes it difficult to apply standard algorithms to solve the bounding problem.

If one writes network flow equations for this model, it turns out that the third condition above introduces what amounts to a second set of network constraints; algebraically, we must solve two network problems simultaneously. From this set of equations one can also show, by means of a lemma due to Ghouila-Houri [9], that the optimization problem is not totally unimodular. Total unimodularity is well known as a sufficient condition for integer solutions. Thus it is important to find another way to prove that this problem is guaranteed to have integer solutions.

While it would be possible to describe our algorithm in terms of flows, nothing is gained by it, so our discussion will be in terms of consistent graphs. However, we will use some of the terminology associated with flows in graphs when it aids the exposition.

### 4 A Numerical Example

Consider again the graph in Figure 2. Suppose we wish to determine the maximum value for the node $x_{111}$. We approach this problem by considering whether or not its value can be increased by one. For this to be possible, it is necessary that

1. there be a consistent adjustment of $x$ values in $I_1$ which leaves the $A$ and $B$ nodes unchanged but increases $x_{111}$ by one, and
2. there exist a complementary consistent adjustment in $I_2$ such that the values of the $C$ nodes are unchanged.

To make the discussion clearer, and to relate it to well-known ideas from graph theory (cite), designate an arc to be:

1. *Green* if it joins a node $x_{ijk}$ with an $A$, $B$, or $C$ projection node having the same value. In this case, the value of $x_{ijk}$ cannot be increased.
2. *Black* if it joins a node $x_{ijk}$ with value zero to an $A$, $B$, or $C$ projection node. In this case the value of $x_{ijk}$ cannot be decreased.

![Figure 2: A Specific Problem](image-url)
3. Red otherwise.

Thus in Figure 2, the arcs \(x_{112} - B_{12}, x_{121} - A_{12}\) and \(x_{222} - C_{22}\) are green, arcs \(x_{122} - A_{12}, x_{122} - B_{12}\) and \(x_{122} - C_{22}\) are black, and the remainder are red. An arc can be colored both green and black if it connects an \(x\) node with value zero to a projection node also with value zero.

The idea of coloring the arcs is to distinguish how the existing values of a microdata node may change. We proceed by labeling certain arcs with a “+” or a “−”, indicating that the \(x_{ijk}\) node it touches will either increase or decrease by one. Thus arcs colored green may be labeled “+”, those colored black may be labeled “−”, while red arcs may be labeled with either sign.

An arc colored both green and black cannot be labeled with either “+” or “−”, since the \(x\) node it touches can neither increase or decrease.

Starting at \(x_{111}\), we look for an elementary circuit through \(I_1\) which gives each arc a label compatible with the current coloring, and which

1. respects the constraints that the \(A_{ij}\) and \(B_{ik}\) values remain unchanged (implying that one entering arc be labeled “+” and one outgoing arc be labeled “−”), and

2. labels each arc leaving any \(x_{ijk}\) uniformly (i.e., if one arc leaving \(x_{ijk}\) is labeled “+” (resp. “−”), the remainder are as well).

An example of a labeling which satisfies these conditions is shown in Figure 3. Thus when considering subgraph \(I_1\) alone, it is possible to increase \(x_{111}\) by at least one (in this example, an increase of one turns out to be the largest possible; for other examples, it may be possible to effect a greater increase).

Having succeeded in finding a consistent increase for \(x_{111}\) within \(I_1\), the second question is whether a compatible adjustment exists in the remainder of the graph. (In this example, there is only one other subgraph, but in general there may be many.) This can be determined as follows. A “+” (resp. “−”) on a local node \(x_{ijk}\) will be “mirrored” as a “+” (resp. “−”) across a \(C_{jk}\) node linking the current subgraph to the rest. So by labeling the \(x_{ijk}\) nodes in other subgraphs in this fashion, this becomes a problem of determining if, in the remainder of the graph, a consistent adjustment matching this labeling exists.

In the current example, there is only one other subgraph, \(I_2\). Thus the problem is to determine if it can be labeled in accordance with the constraints illustrated in Figure 4. It turns out that this is possible, and the final adjustment graph is shown in Figure 5.

The preceding analysis shows that \(x_{111}\) may be increased by (at least) one. By continuing this process, we arrive at an upper bound in a finite number of steps, because all the data of the problem are integral, and because there is \textit{a priori} a finite upper bound, the grand total of the three-dimensional microdata table. At this point, we can update the graph, incorporating the circuit of adjustments just discovered. The updated graph is that of Figure 6. For the node \(x_{111}\) the upper bound has in fact been achieved. Note that after the update, the arc \(x_{111} - B_{11}\) is now colored green, indicating that \(x_{111}\) may not be increased.

Figure 3: A Consistent Cycle

\[ I_1 \]

\[
\begin{array}{c}
7 \quad A_{11} + \\
3 \quad B_{11} + \\
1 \quad C_{11} + \\
5 \quad D_{11} +
\end{array}
\]

Figure 4: Labeling of the Right Subgraph

\[ I_2 \]

\[
\begin{array}{c}
2 \quad A_{21} - \\
3 \quad B_{21} - \\
1 \quad C_{21} - \\
2 \quad D_{21} -
\end{array}
\]

Figure 5: A Complete Compatible Adjustment

\[ I_1 \]

\[
\begin{array}{c}
7 \quad A_{11} + \\
3 \quad B_{11} + \\
1 \quad C_{11} + \\
5 \quad D_{11} +
\end{array}
\]

\[ I_2 \]

\[
\begin{array}{c}
2 \quad A_{21} - \\
3 \quad B_{21} - \\
1 \quad C_{21} - \\
2 \quad D_{21} -
\end{array}
\]
At this point we note that by following the procedure given above, it may be possible to increase the value of an \(x_{ij}k\) node by more than just one unit. Obviously it is straightforward to keep track of the maximum increase or decrease allowable for each arc along a cycle. The total change possible at a single step of the procedure would then be the minimum of these. We also note that at any stage of the procedure, we may discover that one or more nodes \(x_{ij}k\) have achieved their upper or lower bounds. In a program designed to iteratively find bounds for all cells in the micro-data table, logging these occurrences as they arise may save considerable time.

5 Algorithm Development

Formally, we denote by \(I_p\), \(p = 1, \ldots, l\) the subgraph consisting of nodes \(A_{pj}, B_{k}, x_{ij}k\), and arcs \(x_{ij}k = A_{pj}, x_{ij}k = B_{k}\), for all \(j = 1, \ldots, l, k = 1, \ldots, K\). Suppose that the arcs in the subgraph \(I_p\) have been colored either green, black or red.

5.1 Consistent Circuits

**Definition 2** A consistent circuit through \(I_p\) is an elementary circuit in which each \(A\) and \(B\) node in the circuit has exactly one adjacent arc labeled “+” arc and one adjacent arc labeled “−”, and in which no green arc is labeled “+” and no black arc is labeled “−”. A consistent increasing (resp. decreasing) circuit for \(x_{ij}k\) is a consistent circuit containing \(x_{ij}k\) for which the arcs leaving it are both labeled “+” (resp. “−”).

**Lemma 1** If \(x_{pq}r\) is less than its minimum value then a consistent increasing circuit exists for it in \(I_p\).

**Proof:** Let \(x_{pq}r\) be the current values of the \(x\) nodes in \(I_p\), and let \(x'_{pq}r\) be a set of (not necessarily unique) values for these nodes when \(x_{pq}r\) is maximized. Since \(x_{pq}r\) is less than its maximum, both arcs leaving it must be non-green, otherwise no increase would be possible. Label both these arcs “+”. One of these arcs enters \(A_{pq}\) and the other enters \(B_{pr}\). Each of these latter nodes must have a non-black arc leaving it, since \(x'_{pq}r\) does not change their values. Find \(q'\) and \(q''\) such that \(x_{pq}r' = A_{pq}\) and \(x_{pq}r'' = B_{pr}\) are non-black, and such that \(x'_{pq}r < x_{pq}r'\) and \(x'_{pq}r < x_{pq}r''\). Once again, this is possible by the existence of \(x_{pq}r\). Label \(x_{pq}r' = A_{pq}\) and \(x_{pq}r'' = B_{pr}\) “+”.

Since \(x_{pq}r' < x_{pq}r'\) and \(x'_{pq}r < x_{pq}r''\), arcs \(x_{pq}r = B_{pr}\) and \(x_{pq}r' = A_{pq}\) cannot be black. Label them “−”. Finally, arcs \(x_{pq}r' = A_{pq}\) and \(x_{pq}r'' = B_{pr}\) cannot be green, since if they were, arcs \(x_{pq}r = A_{pq}\) and \(x_{pq}r'' = B_{pr}\) would have been black. Thus they may be labeled “+”. This completes a consistent increasing circuit for \(x_{pq}r\). \(\square\)

The complementary result for consistent decreasing circuits requires a slightly different proof.

**Lemma 2** If \(x_{pq}r\) is greater than its minimum value then a consistent decreasing circuit exists for it in \(I_p\).

**Proof:** As before, let \(x_{pq}r\) be the current values of the \(x\) nodes in \(I_p\), and let \(x_{pq}r\) be a set minimizing \(x_{pq}r\). Arcs leaving \(x_{pq}r\) must be non-black; label them “−”. One of these arcs enters \(B_{pr}\), so there must be a node \(x_{pq}r'\) for which \(x_{pq}r' < x_{pq}r\), implying that arcs entering it cannot be green. Label them “−”. One of these non-green arcs enters \(A_{pq}\), so there must be a node \(x_{pq}r''\) which has decreased. It is thus possible to label arcs leaving it “−”, and these arcs are not black. One of these arcs enters \(B_{pr}\). Finally, arc \(x_{pq}r'' = B_{pr}\) cannot be green, for if it were, \(x_{pq}r'' = B_{pr}\) would have to be black, a contradiction. Similarly, arc \(x_{pq}r'' = A_{pq}\) cannot be green, for otherwise arc \(x_{pq}r'' = A_{pq}\) would have to be black. Thus we can label both \(x_{pq}r'' = B_{pr}\) and \(x_{pq}r'' = A_{pq}\) “−”, completing a consistent decreasing cycle for \(x_{pq}r\). \(\square\)

Note that in both proofs a circuit of length eight was constructed, containing exactly four \(x\) nodes, two \(A\) nodes, and two \(B\) nodes. We will show that these are the most complex circuits that need be considered in a search for upper and lower bounds. To do this we will use a slight rearrangement of the subgraphs.

Figure 7 shows an arbitrary subgraph redrawn to emphasize the connectivity of the \(x\) nodes. If we ignore the \(x\) nodes altogether, this is a complete bipartite graph with partition \(\{A_{ij}, B_{k}\}\). An arc in this bipartite graph consists of two arcs of the earlier graph,

![Figure 6: The Updated Graph](attachment:image)
but since those two arcs must be labeled the same way (i.e., either both with “+” or both with “-”), we can label an arc in the new graph with a single “+” or “-”. In this graph, the circuit constructed in Proposition 1 consists of four arcs, two labeled “+” and two “-”.

**Lemma 3** Any consistent circuit through a subgraph I_i which increases x_ijk can be reduced to one with just four arcs.

**Proof:** Since the circuit is increasing for x_ijk, it contains A_{ij} - B_{ik}. Choose A_{ik} and B_{ik} such that B_{ik} - A_{ij} and A_{ij} - B_{ik} are in the circuit. Both of these arcs are necessarily labeled “-”. Remove all other arcs but these three, add the arc A_{ik} - B_{ik}, and label it “+”. This is possible by the same reasoning used in the proof of Proposition 1. \[Q.E.D.\]

Once again, a similar result holds for decreasing circuits. We further note that the amount by which x_ijk may be increased or decreased by using the reduced circuit is at least as large as that for the original. This leads to the following.

**Proposition 1** At most (J - 1)(K - 1) circuits must be examined to determine if the upper or lower bound for x_ijk has been achieved.

**Proof:** From the proof of Lemma 3, we need only consider circuits including nodes A_{ij}, B_{ik}, A_{ik}, and B_{ik} for j' \neq j, k' \neq k. There are thus J - 1 choices for j' and K - 1 choices for k', resulting in (J - 1)(K - 1) total choices. \[Q.E.D.\]

A consistent circuit including only four projection nodes (or equivalently, four x nodes) will be called minimal. In what follows we will sometimes use the notation [A_{ijj}, B_{ik}, A_{ik'}, B_{ik'}] to specify a circuit for the node x_{ijk}. An equivalent alternative is to specify the microdata nodes rather than the projection nodes; the notation [x_{ijj}, x_{ik'}, x_{ik'}, x_{ijk}] specifies the nodes in their order in the circuit. The context will make clear whether an increasing or decreasing circuit is meant.

### 5.2 Complementary Circuits

Having characterized increasing and decreasing consistent circuits for a subgraph, it must next be determined if such a circuit can be matched by complementary changes in the rest of the network describing the disclosure problem. Searching through the remaining I - 1 subgraphs, one at a time, may produce a complementary circuit, but in general the problem is more difficult.

Figure 8 presents a disclosure problem for which there exists exactly one consistent increasing circuit in I_1 for x_{132}. However, it can be verified that no complementary circuit exists in either I_2 or I_3 alone. There is, though, a consistent way to increase x_{132}. The three circuits [x_{132}, x_{123}, x_{132}, x_{132}], [x_{232}, x_{213}, x_{212}, x_{222}], and [x_{332}, x_{313}, x_{312}, x_{333}], in subgraphs I_1, I_2, and I_3 respectively, show that x_{132} can be increased by one. We now present a systematic way of searching for such multi-subgraph circuits.

Suppose, without loss of generality, that x_{111} is to be maximized, and that [x_{111}, x_{11q}, x_{1q}, x_{11r}] is a consistent increasing circuit for it. Note, first of all, that a sequence of usable circuits in other subgraphs must be in one or the other of the following two forms.

1. [x_{i11}, x_{iq}, x_{iq}, x_{i11}] for a sequence i = i_1, ..., i_q, or
2. [x_{ij1}, x_{ijr}, x_{ijr}, x_{ij1}] for i = i_1, ..., i_r.\[Q.E.D\]

This is necessarily so because if any subgraph contained a circuit containing either x_{111} and x_{1q}, or x_{iq} and x_{1r}, that circuit would necessarily be a complement to the one found in I_1, and no further search would be required. This means that a search for a complementary sequence of subgraph circuits can be divided into two parts, corresponding to the two types enumerated above.

To search for a sequence of the first type, enumerate for each subgraph I_i, i \neq 1, pairs (k, k') for which a consistent circuit of type 1 above exists. There may be as many as K(K - 1) of these for each i. Next, construct a graph whose vertices are the pairs (i, k), i = 1, ..., I, k = 1, ..., K. For each i > 1, add arcs (i, k) \rightarrow (i, k') for each pair (k, k').
just identified. For each $1 < i < I - 1$, also add arcs $(i, j) = (i+1, j)$, $j = 1, \ldots, J$. See Figure 9. Then to find a sequence of circuits which complements that for $x_{111}$ in $I_3$, we attempt to find a path connecting the vertices $(1, 1)$ and $(1, r)$. If such a path exists, and its arrangement of increases and decreases match those for $I_3$, we have found a sequence which increases $x_{111}$.

6 Computational Complexity

The procedure outlined in the previous section takes advantage of the structure that results from the interlocked nature of the three-dimensional disclosure problem. To find an upper bound for a given microdata cell value $x_{ijk}$, it attempts to find a consistent circuit in the subgraph containing the $x$ variable, and then searches for a complementary circuit in the remainder of the graph. If both circuits are found, then $x_{ijk}$ may be incremented by at least one. Because $x_{ijk}$ is a priori bounded above by the grand total of the microdata table $T$, there is a natural bound on the number of iterations of this procedure that must be performed. Analogous remarks apply to the lower bounding case, because the count data we deal with are inherently bounded below by zero.

Within the original subgraph, at most $(J-1)(K-1)$ circuits must be examined. Each of these may be done in constant time. For any consistent circuit discovered, $I - 1$ remaining subgraphs must be analyzed for possible complementary circuits, and each of these requires considering $K(K-1)$ pairs of vertices. Each of these pairs may be processed in constant time as well. Thus $I(J-1)(K-1)^2 K$ constant time operations must be performed in the worst case to increment $x_{ijk}$ by at least one. If $U$ is the grand total of $T$, then the upper bounding procedure for any $x_{ijk}$ is of order $BIJK^3$.

For lower bounds, the maximum number of iterations is given by the value of $x_{ijk}$ itself, but still depends on the size of the table according to the factor $IK^3$. Computational experience to date suggests that these bounds are never even approached. For tables of sizes $3 \times 3 \times 3$ through $7 \times 7 \times 7$, we find that processing between three and five consistent circuits will yield a guaranteed upper or lower bound. Having found a consistent circuit, our current implementation first looks for a simple complementary circuit within a single subgraph. This is almost always sufficient; only in rare instances is it necessary to construct the full
Other efficiencies are also being explored. It is obviously duplicative to construct the graph in Figure 9 for each $x_{i\cdot}\cdot$. Similarly, the evaluation of the complementarity of a local circuit with one found in the remainder of the graph can no doubt be simplified. More sophisticated data structures can likely store much of this information and eliminate the need to constantly reconstruct it.

7 Conclusions

The problem addressed in this paper—inferring bounds on entries in three-way tables given two-dimensional projections—is of considerable importance to data collection agencies worldwide. Although this problem has been discussed in both the scientific and public policy literatures, the results presented here are the first practical means to assess the disclosure potential of such projections. We have given a polynomial-time algorithm that guarantees optimal bounds on sensitive cells, allowing database administrators to judge the extent to which confidentiality might be compromised by their release.

To build on these results, a number of avenues are open. First, the algorithm is in an early stage of development. Improvements in data structures and sequencing of operations are clearly possible. Second, extensions to higher dimensions have not been explored here, but may be easily obtained. Third, the connection between the network algorithm presented here and the statistical methods of Frechet and Bonferroni bounds needs to be determined. Finally, we can ask if the network formulation used to derive our algorithms leads to a way to minimally mask published projections (through cell suppression, for instance) so that unacceptable disclosure cannot occur.

References


