A Polynomial Algorithm for Optimal Microaggregation

Stephen Lee Hansen

Premiere Technologies, Inc.,
824 South Military Trail, Deerfield Beach, FL 33442
Email: hansensl@acm.org

and

Sumitra Mukherjee

Graduate School of Computer and Information Sciences, Nova Southeastern University
6100 Griffin Road, Davie, Florida 33314-4416
Email: sumitra@nova.edu

Abstract - Microaggregation is a technique that is used by statistical agencies to limit disclosure of sensitive microdata. Noting that no polynomial time algorithms are known to microaggregate optimally, Domingo-Ferrer and Mateo-Sanz have presented heuristic methods based on hierarchical clustering and genetic algorithms to identify sub-optimal solutions. We present an efficient polynomial time algorithm to solve the univariate microaggregation problem. Optimal partitions are shown to correspond to shortest paths in a network.

Index Terms - Statistical databases, microdata protection, microaggregation, clustering, network optimization, information loss.

1. INTRODUCTION

Microaggregation is a disclosure limitation technique that is widely used by statistical agencies. Under microaggregation, records are aggregated into groups. Instead of releasing the actual values of sensitive values for individual records, the mean of the group to which the observation belongs is released. The confidentiality of individual data subjects is protected by ensuring that each group has at least a minimum number of observations, *k*. Domingo-Ferrer and Mateo-Sanz [3] have investigated microaggregation methods that attempt to minimize information loss by grouping similar observations together, rather than using a fixed group size. The objective is to partition the data into groups that minimize information loss due to aggregation. The sum of squared errors (SSE) due to aggregation is taken as a measure of information loss. They observe that "no polynomial algorithms are known to date to microaggregate optimally". Hence they present heuristic methods based on hierarchical clustering and genetic algorithms for obtaining sub-optimal solutions.

We present an efficient polynomial algorithm to microaggregate optimally. Optimal partitions are shown to correspond to shortest paths in a network. The complexity of the resulting algorithm is of O(n log n) (where n is the number of records in the database) and hence of value even for large practical problems. Section 2 describes the univariate microaggregation problem as formulated in [3] and summarizes their results that we build on. Our new algorithm is developed in section 3. Section 4 presents results of some experiments using our algorithm and compares its performance to those presented in [3]. Section 5 summarizes our contributions.

2. THE PROBLEM AND PRIOR RESULTS

The univariate microaggregation problem, as formulated in [3], can be stated as follows: Consider a sensitive attribute X within a microdata set with n observations. Microaggregation involves

partitioning the n observations into g groups, such that each group contains at least k observations. Using the sum of within-group squared errors as a measure of similarity between observations, the objective is to find a partition that minimizes the sum of the within-group squared error, $SSE = \sum_{j=1}^{g} \sum_{i=1}^{n_j} (x_{ij} - \bar{x}_j)^2$, where x_{ij} is the i^{th} number in the j^{th} group, $i = 1, 2, \ldots, n_i$, $j = 1, 2, \ldots, g$, and $\bar{x}_j = \frac{1}{n_j} \sum_{i=1}^{n_j} x_{ij}$ is the mean for group j.

Domingo-Ferrer and Mateo-Sanz [3] exploit certain characteristics of the problem to reduce the number of candidate optimal solutions to the problem. We summarize these results as *Results* 1 and 2 and build on it.

Result 1. Under an optimal partitioning the observations in each group are contiguous if the observations are sorted in an ascending order.

Result 2. In every optimal partition, each group has between k and $2k-1$ observations.

Using these results Domingo-Ferrer and Mateo-Sanz [3] present two heuristic methods for suboptimal microaggregation. The first, called k-Ward algorithm, is based on Ward hierarchical clustering and the second uses a genetic algorithm based approach. Assuming that the data set *is already sorted* on the variable of interest, they show that the computational complexity of the k-Ward algorithm is of $O(n^2)$ and that of the genetic algorithm is linear in n.

3. A SHORTEST-PATH ALGORITHM

We formulate the microaggregation problem as a shortest-path problem on a graph. We first construct the graph, then show that the optimal microaggregation corresponds to the shortest path in this graph in a natural way. Each arc of the graph corresponds to a possible group that may be part of an optimal partition. Each arc is labeled by the error that would result if that group were to be included in the partition. The algorithm takes advantage Results 1 and 2 to limit the number of groups that must be considered, hence limiting the number of arcs in the graph.

3.1 Graph Construction

Let $\boldsymbol{V} = V_1, \ldots, V_n$ be a vector of length n consisting of the observation values sorted into ascending order. V_1 is the least element of $\boldsymbol{V}$, and V_n is the greatest element of $\boldsymbol{V}$. Let k be an integer such that $1 \le k < n$. For each k and n, we construct the graph $\boldsymbol{G}_{n,k}$ as follows:

For each element V_i of $\boldsymbol{V}$, the graph has a node with label i. The graph also has one additional node with label 0. For each pair of graph nodes (i, j) such that $i + k \leq j < i + 2k$, the graph has a directed arc (i, j) from node i to node j.

For each arc (i, j), the corresponding group $C_{(i,j)}$ is the set of values $\{V_h : i < h \leq j\}$. We say that group $C_{(i,j)}$ corresponds to arc (i, j).

For each arc (i, j), let the length $L_{(i,j)}$ of the arc be the within-group sum squared error for the corresponding group $C_{(i,j)}$. This may be computed as:

$L_{(i,j)} = \sum_{h=i+1}^{j} (V_h - M_{(i,j)})^2$ where $M_{(i,j)} = \frac{1}{j-i} \sum_{h=i+1}^{j} V_h$ is the mean of the numbers in group $C_{(i,j)}$.

3.2 The Optimal Partition

We now show that the optimal partition is exactly the set of groups that correspond to the arcs of a shortest path from node 0 to node n on this graph.

Theorem 1.

Every group in each optimal partition corresponds to one of the arcs of the graph.

Proof:

This follows immediately from Results (1) and (2) and the construction of the graph. Each group consists of at least k and at most $2k$ - 1 contiguous observations. Contiguous groups of observations are contiguous groups of elements in the sorted vector $\boldsymbol{V}$. For every contiguous group of at least k and at most $2k$ - 1 elements of the vector $\boldsymbol{V}$, there is an arc in the graph.

Theorem 2.

Each optimal partition corresponds to a path from node 0 to node n in the graph.

Proof:

From theorem (1) we know that each group in the optimal partition corresponds to an arc in the graph. Further, if the optimal partition does not correspond to a path from node 0 to node n, then either some V_i is left out, or some V_i is included more than once.

Theorem 3.

The length of a path from node 0 to node n is exactly the *SSE* for the partition that corresponds to that path.

Proof:

This follows directly from the definitions of $L_{(i,j)}$ and *SSE*. Each $L_{(i,j)}$ is just the within-group sum-squared error for the group corresponding to the arc (i, j). The total length of a path is the sum of the lengths of the individual arcs.

It follows immediately that the minimal *SSE* is exactly the length of a shortest path from node 0 to node *n*, and that the partition corresponding to each shortest path is an optimal partition. We may now assert the correctness of algorithm *MA*, presented below, for the microaggregation problem.

Algorithm: MA

Input: Integer parameters k and n, and vector $\boldsymbol{V} = V_1, \ldots, V_n$ of real numbers.

Output: An optimal microaggregation of $\boldsymbol{V}$.

Complexity: $O(\max(n \log(n), k^2 n))$

Procedure:

1. Sort $\boldsymbol{V}$ into ascending order.
2. Construct the graph $\boldsymbol{G}$ as described in section 3.2.
3. Let path P be a shortest path in the graph $\boldsymbol{G}$ from node 0 to node n.
4. For each arc in the path P, output the corresponding group.

3.3 Complexity

The complexity of algorithm *MA* is determined by the size of the graph and the complexity of the chosen shortest-path algorithm. The graph is a directed acyclic graph with $n + 1$ nodes and fewer than kn arcs. The computation of the length of each arc requires $O(k)$ arithmetic operations, so the graph construction is $O(k^2 n)$. A shortest path algorithm based on dynamic programming is available with complexity $O((k+1)n)$ for this graph [1]. For $k < \sqrt{\log(n)}$, the complexity is dominated by the complexity of sorting the elements before starting the search, so the total complexity of the algorithm is $O(\max(n \log(n), k^2 n))$. The algorithm is efficient enough for use even on very large data sets since for practical applications k is typically small. Note that the computational complexity of the algorithms estimated by Domingo-Ferrer and Mateo-Sanz [3] assume that the data set is sorted. Hence our algorithm for optimal microaggregation is at least as efficient as published approximate methods.

4. EXPERIMENTAL RESULTS

We implemented algorithm *MA* in C++ to compare the performance of our algorithm to the computational results presented in [3]. Domingo-Ferrer and Mateo-Sanz [3] used a data set with 834 records. Not having access to this data set, we simulated sets of $n = 1000$ records for our experiments. In each case our algorithm took less than a second to execute on a desktop computer. Even for much larger data sets ($n > 10{,}000$) the computational time was negligible. Since computational time is not an issue, we used information loss as a measure to compare our algorithms.

A fixed-group size approach results in lower information loss than every heuristic algorithm proposed in [3] in more than 15% of the cases reported. Even in cases where the heuristic methods out-perform the fixed size microaggregation, the differences in information loss are not very significant. We use the ratio of the sum of squared errors (SSE) due to grouping obtained under the fixed-size group approach to those obtained under our algorithm as a measure of performance. Since our algorithm minimizes the SSE of a partition, this ratio is always greater than or equal to 1. The higher the ratio, the more beneficial it is to use our approach.

In our computational experiment we used group size $k = 3$ and input vector length $n = 1000$. Input vectors were generated with three different distributions: uniform (min = 0; max = 1000); normal ($\mu = 500$; $\sigma = 150$); and exponential ($\mu = 500$). For each distribution, 1000 input vectors were generated using the pseudo-random number generator R250 [2].

Table 1 gives summary statistics of the resulting SSE ratios. For each distribution, the mean, standard deviation, minimum, and maximum of the observed SSE ratios are given. These results indicate that our new algorithm performs significantly better than the fixed-group-size algorithm. The largest improvements occur when the sample has a disproportionate number of outliers. This is particularly significant since confidentiality concerns about data subjects with outlier values for sensitive attributes are typically much greater.

Table 1. *SSE ratios of partitioning algorithms, for three input distributions. (k=3, n=1000, 500 runs)*

Distribution	SSE Ratio			
	Mean	Std. Dev.	Minimum	Maximum
Uniform	1.7964	0.18447	1.3605	2.4208
Normal	1.1533	0.22668	1.0072	3.5602
Exponential	1.1514	0.39815	1.0003	5.0515

Table 2. *SSE ratios of partitioning algorithms, for different group sizes k. (n=1000, 500 runs)*

Group Size Size K	SSE Ratio			
	Normal Mean	Normal Maximum	Uniform Mean	Uniform Maximum
2	1.3177	7.0436	2.253	3.4144
3	1.1537	2.6761	1.7867	2.3866
4	1.085	1.7621	1.5358	1.876
5	1.0542	1.4708	1.4136	1.8215
6	1.0441	1.3747	1.3415	1.6334
7	1.0338	1.2883	1.3001	1.5869
8	1.0254	1.1786	1.2723	1.6431

Table 2 shows the results of several additional experiments to investigate the sensitivity of this result to the choice of k. For these experiments, we used normal (μ=500, σ=150) and

uniform(max=1000) input data, vector length n = 1000, and 500 runs for each parameter value. The results of these sensitivity tests indicate that as k decreases, the relative performance of our proposed algorithm improves.

5. CONCLUSIONS

No polynomial time algorithms have been published that microaggregate optimally. We have provided a shortest path based algorithm to solve this problem. For small k the complexity of our algorithm is O(n log(n)), which is dominated by the time required to sort the data. Hence, the algorithm is at least as efficient as published heuristic methods and can be used on very large data sets. It is superior to published heuristic algorithms since it guarantees minimum information loss. We ran experiments to compare the performance of our algorithms to existing fixed-size-group algorithms and found that the improvement is substantial. While our algorithm focuses on univariate data, it can be used on multivariate data when the data vectors are sorted one variable at a time or are projected to a single axis, as described in [2].

REFERENCES

[1] T.H. Cormen, C.E. Leiserson and R.L. Rivest, *Introduction to Algorithms*, Cambridge, MA: MIT Press, 2nd ed, 1990.

[2] S. Kirkpatrick and E.P. Stoll, "A Very Fast Shift-Register Sequence Random Number Generator," *Journal of Computational Physics*, vol 40, no. 2, pp. 517-526, 1981.

[3] J.M. Mateo-Sanz and J. Domingo-Ferrer, "Practical Data-Oriented Microaggregation for Statistical Disclosure Control", *IEEE Trans. on Knowledge and Data Engineering*, accepted and processed as a "preprint" paper, posted to the Digital Library at: http://dlib.computer.org/tk/books/tk2001/pdf/106236.pdf