Multivariate Microaggregation with Fixed Group Size Based TSP

Armando Maya, Agustí Solanas

Abstract—**Microaggregation is a clustering problem with cardinality constraints that originated in the area of statistical disclosure control for microdata. Microdata consist of sets of records containing information on individual respondents or business entities. To protect the anonymity of respondents (individuals, organizations), the access to microdata is restricted.
Microaggregation is a statistical disclosure control technique for microdata disseminated in statistical databases. The principle of microaggregation is to aggregate original database records into small group that preserve their statistical properties.
In recent years, a large number of heuristic methods have been developed to optimize the solution, achieving a balance between information loss and privacy protection of respondents. In this article, we propose a new heuristic technique for multivariate microaggregation. For this purpose, the solution to the traveling salesman problem (TSP) is used. The adaptations required to characterize the multivariate microaggregation problem are explained and justified. In addition, a real data set is used to compare the information loss and output data quality with the most relevant previous proposals.**

***Keywords*—Microaggregation, Traveling salesman problem, Privacy, Statistical disclosure control**

I. Introduction

The aim of statistical disclosure control (SDC) is to ensure that statistical outputs provide as much value as possible to the users while protecting the confidentiality of information concerning individuals or entities. SDC methods modify, summarize or perturb the data and there are a range of different methods that can be used to protect different outputs. SDC methods can be pre-tabular (applied to the underlying microdata) or post-tabular (applied to tables). Microaggregation is a family of methods for statistical disclosure control (SDC) of microdata, that is a perturbative data protection method. Given the original data file, it consists of constructing clusters from the data (each cluster should have between k and 2k elements) and then replacing each original data by the centroid of the corresponding cluster. The original records in the data set are partitioned into several groups in such a way that records in the same group are very similar to each other. The number of records in each group must be at least *k*. This parameter *k* can be considered a security parameter: the larger *k*, the more secure the microaggregation. The resulting clustering of the data set is known as k-partition.

Armando Maya is MISTIC student at the UOC.(amayal@uoc.edu)
Agusti Solanas is with the Department of Computer Engineering and Mathematics, Universitat Rovira i Virgili, Av. Paisos Catalans 26, 43007, Tarragona, Catalonia, Spain (email: agusti.solanas@urv.cat).

A. Basics of microaggregation

Microaggregation is a family of perturbative SDC methods originally designed for continuous numerical data. Formally, microaggregation can be defined as follows. Consider a microdata set V with p continuous numerical attributes and n records (i.e., the result of observing p attributes on n individuals). With these records, groups are formed with n_i records in the i-th group ($n_i \geq k$ and $n = \sum_{i=1}^{g} n_i$), where g is the number of resulting groups. Optimal microaggregation is defined as the one yielding a k-partition maximizing the within-groups homogeneity. The sum of squares criterion is commonly used for measuring the homogeneity in clustering. In terms of sums of squares, maximizing within-groups homogeneity is equivalent to finding a k-partition minimizing the within-groups sum of squares SSE defined [12] as:

$$SSE = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (x_{i,j} - \hat{x_i})(x_{i,j} - \hat{x_i})' \tag{1}$$

The total sum of squares [12] is:

$$SST = \sum_{i=1}^{n} (x_i - \hat{x})(x_i - \hat{x})' \tag{2}$$

Based on SSE, the microaggregation problem consists of finding a k-partition with minimum SSE. Whereas, that the sizes of groups in the optimal k-partition lie between k and 2k-1. A measure L of information loss [10] standardized between 0 and 1 can be obtained from:

$$L = \left(\frac{SSE}{SST} \right) \tag{3}$$

Optimal microaggregation is an NP-hard problem [2], for multivariate records. For univariate data, a polynomial-time optimal algorithm is know. This algorithm has complexity $O(k^2 n)$ and solves optimal univariate microaggregation as a shortest-path problem on a graph. Most of datasets are multivariate, therefore typically, microaggregation is multivariate and heuristic.

In microaggregation [6], the main types of heuristics are two:

- Fixed-size. These heuristics yield k-partition where all groups have size k, except perhaps one group which has size between k and 2k-1. See, figure 1.
- Variable-size. These heuristics yield k-partitions where all groups have sizes varying between k and 2k-1. The

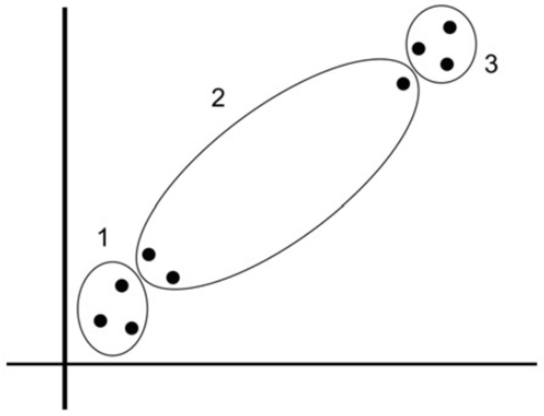

Fig. 1: Fixed-sized clustering with k=3

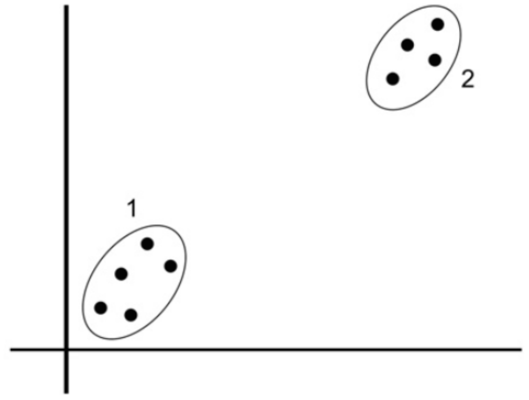

Fig. 2: Variable-sized clustering with k=3

challenge is how to enforce cardinality constraints on groups without substantially increasing SSE. As shows, figure 2.

Fixed-size microaggregation heuristics are computationally very efficient, however variable-size heuristics have lower information loss, increasing computational complexity, resulting in $O(n^3)$.

B. Privacy benefits of microaggregation

The records in an original dataset [2] [11] can be classified in four categories:

- Identifiers. These are attributes that unambiguously identify the respondent. Examples are full name, social security number,etc. These data must be pre-processed and eliminated.
- Key attributes. Also called quasi-identifiers are a set of attributes that can be linked with external information to identify the respondents. Examples are age, gender, job, zipcode, etc.
- Confidential outcome attributes. These are attributes which contain sensitive information on the respondent. For example, salary, religion, political affiliation, health condition, etc.

The usual practice in SDC is for the data protector to apply microaggregation to a restricted set of attributes rather than to entire records in a dataset.
A dataset is said to satisfy k-anonymity for k $>$ 1 if, for each combination of values of key attributes, at least k records exist in the dataset sharing that combination, in this case, a snooper attempting re-identification with an identified external source can only hope to map an identified record in source to a group of k records in microaggreagated dataset.

Microaggregation was proposed at Eurostat in the early nineties, and has since then been used by the British Office for National Statistics and other national agencies [4].

Example1 In this example we can see the use of microaggregation for k-anonymaity [2] [3]. The Table 1 show an original microdata set, for 11 companies in a certain town, the company name, the surface and the number of employees.
The 3-anonymous version of the dataset in Table I is Table II. The identifier ”company name” has been deleted and optimal bivariate microaggregation with k=3 was used on the key attributes ”Surface” and ”Number of employees”. Both attributes were standardized to have mean 0 and variance 1 before microaggregation, in order to give them equal weight.

TABLE I: Example1: original dastaset

Company Name	**Surface(m^2)**	**Number Employees**
Com1	790	55
Com2	710	44
Com3	730	32
Com4	810	17
Com5	950	3
Com6	510	25
Com7	400	45
Com8	330	50
Com9	510	5
Com10	760	52
Com11	50	12

Finally, we can see in Table II, that the 11 records were microaggregated into three groups.

TABLE II: Example1: 3-anonymous version of dastaset.

Surface(m^2)	**Number Employees**
747.5	46
747.5	46
747.5	46
756.67	8
756.67	8
322.5	33
322.5	33
322.5	33
756.67	8
747.5	46
322.5	33

In example 1, there are two attributes that can make a graphical representation in $\mathbb{R}^2$. In Fig.3 shows the results of three groups created.

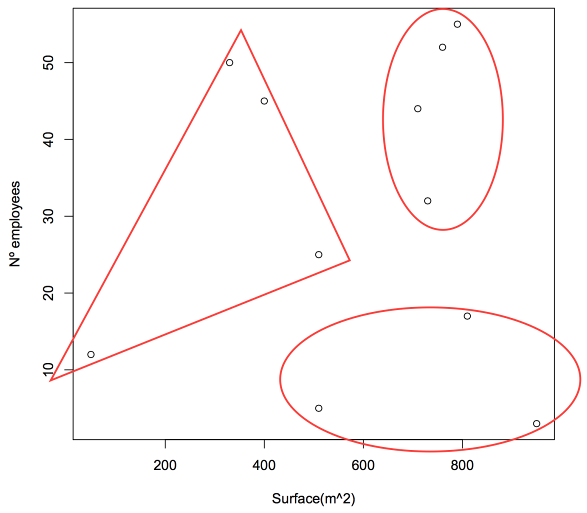

Fig. 3: Optimal 3-partition of dataset

C. Contribution and plan of this paper

In this paper we present a new method based on a problem from graph theory called Traveling Salesman Problem, that can be used over numerical dataset to group elements into similar groups for microaggregating a multivariate microdata. We will compare the results of the algorithms in terms of information loss and disclosure risk. In Section II the proposed algorithm is described. Section III presents experimental results. Finally, Section IV is a conclusion and future work.

II. The proposed method

The proposed method is based on a well-known and important NP-hard combinatorial optimization problem, the traveling salesman problem (TSP) [8]. The TSP is to find a shortest possible tour that visits each city exactly once for a given list of cities an back to starting city. The innovation of our method lies in the representation as a graph theoretic problem to find clusters of records, in the dataset. This is done through TSP.

A. Formulations of the TSP

In this section, we briefly summarize two important formulations of the TSP [8] [5]. On the one hand, we will describe as a permutation problem and on the other hand can also be formulated as a graph theoretic problem.

- Combinatorial optimization problem. The goal is to find the shortest tour that visits each city in a given list exactly once and then returns to the starting city. Formally, the TSP can be stated as follows. The distances between *n* cities are stored in a distance matrix *D* with elements $d_{i,j}$ where $i, j = 1, ..., n$ and the diagonal elemnets $d_{i,i}$ are zero. A *tour* can be represented by a cyclic permutation π of $\{$1,2,...,n$\}$ where $\pi(i)$ represents the city that follows city *i* on the tour. Therefore, the TSP is then the optimization problem to find a permutation π that minimizes the *length of the tour* denoted by:

$$\sum_{i=1}^{n} d_{i,\pi(i)} \tag{4}$$

  For this challenge, the tour length of $(n-1)!$ permutation vectors have to be compared. This results in a problem which is very hard to solve, we know that is NP-complete [8].

- Graph theory problem. The TSP is formulated by a complete graph $G = (V, E)$, where the cities correspond to the node set $V = \{1, 2, ..., n\}$ and each edge $e_i \in E$ has an associated weight w_i representing the distance between the nodes it connects. The goal is to find a *Hamiltonian cycle*, i.e, a cycle which visits each node in the graph exactly once, with the least weight in the graph. This formulation leads to procedures involving minimum spanning trees for tour construction or edge exchanges to improve existing tours.

Finding the exact solution to a TSP with n cities requires to check $(n-1)!$ possible tours [8]. However, solving TSPs is an important part of applications in many areas including vehicle routing, computing wiring, machine sequencing and scheduling, frequency assignment in communication networks. Applications in statistical data analysis include ordering and clustering objects. For example, clustering and ordering using TSP solves is currently becoming popular in biostatistics [7].

B. *Heuristics for the TSP*

The NP-hardness of the TSP already makes it more time efficient for small-to-medium size TSP instances to rely on heuristics in case a good but not necessarily optimal solution is sufficient (for example, figure 4). TSP heuristics typically fall into two groups:

- *Tour construction heuristics* which create tours from scratch, for instance, nearest neighbor algorithm and the insertion algorithms are tours constructors.
- *Tour improvement heuristics* which use simple local search heuristics to improve existing tours.

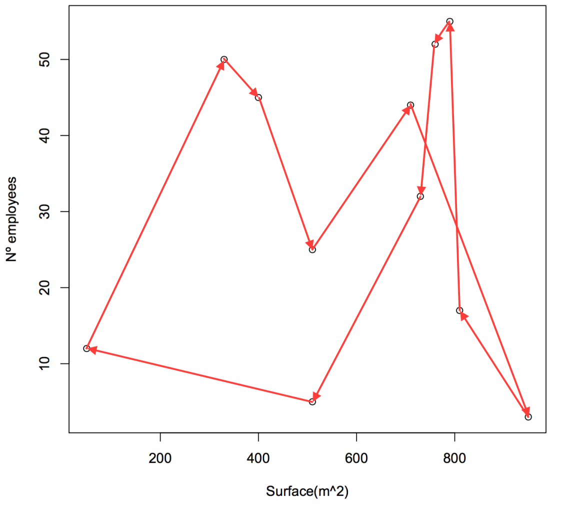

Fig. 4: Hamiltonian cycle heuristic solution to records in example 1

C. *The shortest Hamiltonian path*

The problem of finding the shortest Hamiltonian path through a graph (i.e., a path which visits each node in the graph exactly once, see figure 5) can be transformed into the TSP with cities an distances representing the graphs vertices and edge weights, respectively.
Finding the shortest Hamiltonian path through all cities disregarding the endpoints can be achieved by inserting a ”dummy city” which has a distance of zero to all other cities. The position of this city in the final tour represents the cutting point for the path. The path starts with the first city in the list after the ”dummy” city and ends with the city right before it. Before it, the distance matrix between cities can be modified to solve related shortest Hamiltonian path starting with a given city. All distances to the selected city are set to zero, forcing the evaluation of all possible paths starting with this city and disregarding the way back from the final city in the tour. Note that, the distance to return from the last city in the path to the start city does not contribute to the path length.

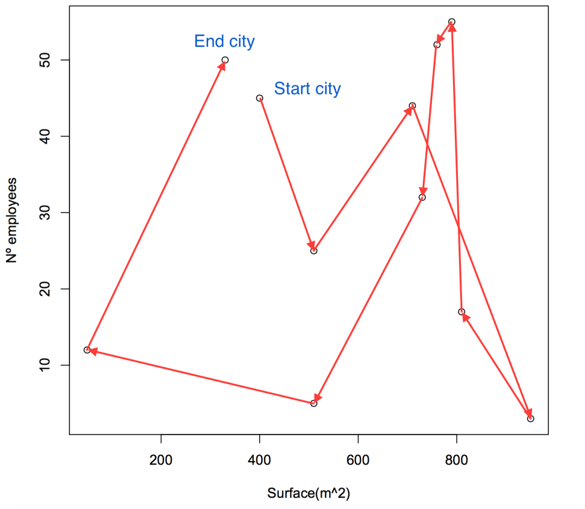

Fig. 5: Hamiltonian path heuristic solution to records in example 1

D. *Rearrangement clustering*

The TSP can be used to obtain a clustering object [5]. The idea is that objects in clusters are visited in consecutive order and from one cluster to the next larger jumps are necessary. This type of clustering is called rearrangement clustering [1] and suggests to automatically find the cluster boundaries of k clusters by adding k *dummy cities* which have constant distance c to all other cities and are infinitely far from each other. In the optimal solution of the TSP, the dummy cities must separate the most distant cities and thus represent optimal boundaries for k clusters.

TABLE III: Example1: 3 Hamiltonian paths solution.

Start City	Hamilton Paths										
Com1	1	2	3	4	5	6	7	8	9	10	11
Com8	8	7	6	2	10	1	5	4	3	9	11
Com10	10	2	6	7	8	11	9	3	4	5	1

Analyzing table 3, we can see that there are patterns that repeat. In mathematics, a *n-tuple* is a sequence of n elements [9], where n is non-negative integer. Continuing with example 1 where k-partition is 3-partition, the clustering problem is defined as finding tuples of k elements with greater frequency, in TSP solution. We can see, that the 3-tuple (3,4,5) appears in all paths. In the paths that start in Com10, the tuple is inverted as (5,4,3) but indicates proximity or a data clustering. It is also noted that the same applies to the 3-tuple (6,7,8), which appears on all Hamiltonian paths.

Algorithm 1 Multivariate Microaggregation with Fixed Group Size Based TSP

```
Require: D dataset with n p-dimensional data points
Require: k Minimum cardinality constraint
Ensure: M Microaggregated Dataset

$Compute\_Distances\_Matrix(D)$
for $i = 1$ to n do
    $d_i = Compute\_Dummy\_city(i, D)$
    $H_{path}(i) = Compute\_TSP\_Starting\_In(d_i)$
end for
for $i = 1$ to n do
    for $j = 1$ to n do
        $N_{i,j} = Search\_Neighbords\_inPaths(H_{path}(i), j)$
    end for
end for
while $Points\_To\_Assign > (2k - 1)$ do
    $max_{i,j} = The\_position\_of\_maximum\_value(N_{i,j})$
    $g_{cluster} = Build\_Group\_From\_Max(max_{i,j})$
    for $i = 1$ to $(k - 2)$ do
        $row\_max_{i,j} = The\_position\_max\_row\_value(N_{i,j})$
        $col\_max_{i,j} = The\_position\_max\_col\_value(N_{i,j})$
        if $(row\_max_{i,j} > col\_max_{i,j})$ then
            $g_{cluster} = Extend\_The\_Group(row\_max_{i,j})$
        else
            $g_{cluster} = Extend\_The\_Group(col\_max_{i,j})$
        end if
    end for
    $N_{i,j} = Delete\_Assigned\_Points(g_{cluster}, N_{i,j})$
end while
$Assign\_Remaining\_Points(N_{i,j}, g_{cluster})$
$M = Compute\_centroid\_Dataset(D, g_{cluster})$
return M
```

E. The new fixed-size microaggregation heuristic

A multivariate dataset consisting of n records and p numerical attributes can be represented as n points $x_1, ..., x_n$ in $\mathbb{R}^p$. In our case, each record is represented by a city and its attributes are the position where the city is in the graph. The new fixed-size heuristic proposed in this paper, described in Algorithm 1, is as follows:

- Find a Hamiltonian path $H_{path}(n)$ traversing all n points of the dataset, starting in city n. Let the $\pi_{H_{path}(n)}$ be the permutation of $\{1, ..., n\}$ expressing the order in which the points are traversed by $H_{path}(n)$.

- Search the neighbors that are visited in order, resulting a simmetryc matrix $N_{i,j}$ where the term i, j represents the number of times that i appears at a distance of $(k - 1)$ in the paths to j.

- The group generation starts searching the maximum value in $N_{i,j}$, that representing two neighboring cities that appear along the path. To find a k-partition, we search the maximum value in row and maximum value en column, select the largest and add it to the group. This last step is done in $(k - 2)$ times. At this moment, we have a *k-partiton* of the dataset int a number of groups.

- Compute the groups centroid. The centroids of the groups forming the matrix M, that is the microaggregated dataset.

Note that, if there are between k and $2k - 1$ unassigned points a new cluster is formed with these points, if there are less points to be clustered, they are assigned to their closest cluster. Therefore, as all clusters have k points, this does not affect the size constraint imposed by microaggregation (i.e. in an optimal k-partition, each cluster must contain a number of points between k and $2k - 1$).

F. Running example

The new method, described in Algorithm 1, is illustrated by the next toy example. Let be D our data set from *Example 1*, considering k = 3. The algorithm operates in the next steps:

- Find a Hamiltonian path $H_{path}(n)$ traversing all *n* points of the dataset, starting in city *n*. The result is shown in table IV.

TABLE IV: Hamiltonian paths on toy example.

Start City	Hamilton Paths										
Com1	1	3	4	6	7	11	10	8	9	5	2
Com2	2	1	3	5	4	6	9	8	10	11	7
Com3	3	1	2	5	4	6	7	11	10	8	9
Com4	4	6	7	11	10	8	9	5	2	1	3
Com5	5	2	1	3	4	6	9	8	10	11	7
Com6	6	5	2	1	3	4	7	11	10	8	9
Com7	7	8	10	11	4	3	1	2	5	6	9
Com8	8	10	11	7	9	5	2	1	3	4	6
Com9	9	6	5	2	1	3	4	11	10	8	7
Com10	10	11	8	7	9	6	4	3	1	2	5
Com11	11	10	8	7	9	6	4	5	2	3	1

- Search the neighbors that are visited in order, resulting a simmetryc matrix $N_{i,j}$ where the term i, j represents the number of times that i appears at a distance of $(k - 1)$ in the paths to j. For instance, we can see in table V, that Company 1 appears in 11 paths close to Company 2.

- The group generation starts searching the maximum value in $N_{i,j}$, that representing two neighboring cities that appear along the path. To find a k-partition, we search the maximum value in row and maximum value en column, select the largest and add it to the group. The table VI shows the result.

TABLE V: Hamiltonians paths on the toy example.

Neighborhood at a distance of (k-1)											
	Com1	Com2	Com3	Com4	Com5	Com6	Com7	Com8	Com9	Com10	Com11
Com1	-	11	11	8	11	0	1	0	1	1	1
Com2	11	-	10	2	10	3	2	0	4	1	1
Com3	11	10	-	9	3	5	1	1	1	0	3
Com4	8	2	9	-	3	8	4	1	4	2	3
Com5	11	10	3	3	-	6	2	1	6	1	2
Com6	0	3	5	8	6	-	7	4	7	1	3
Com7	1	2	1	4	2	7	-	6	5	10	9
Com8	0	0	1	1	1	4	6	-	10	11	8
Com9	1	4	1	4	6	7	5	10	-	6	3
Com10	1	1	0	2	1	1	10	11	6	-	11
Com11	1	1	3	3	2	3	9	8	3	11	-

TABLE VI: Generated group

Group	Record 1	Record 2	Record 3
Group 1	Com1	Com2	Com3
Group 2	Com10	Com8	Com11
Group 3	Com6	Com4	Com5
Remaining Points	Com7	Com9	

The figure 6 represents the generated group with the dataset from example 1.

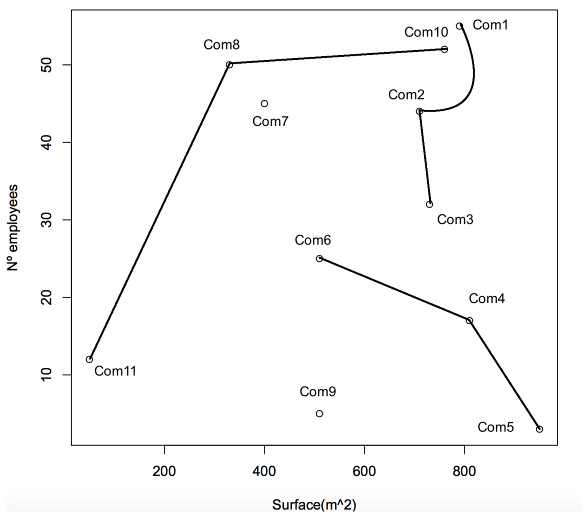

Fig. 6: Generated group with MF-TSP

The proposed method can leave some records unassigned at the end of the main loop. Thus, it is necessary to assign these records to a group before ending the algorithm. The remaining records are assigned to their closest group. Adding last points result, we can see in table VII.

TABLE VII: Addition of the last records

Group	Record 1	Record 2	Record 3	Record 4
Group 1	Com1	Com2	Com3	-
Group 2	Com10	Com8	Com11	Com7
Group 3	Com6	Com4	Com5	Com9

In the last step, compute the groups centroids. The groups centroids forming the matrix M, that is the microaggregated dataset. The table VIII shows the centroids:

TABLE VIII: Toy example: 3-anonymous version of dastaset.

Surface(m^2)	Number Employees
743.33	43.66
743.33	43.66
743.33	43.66
695	12.5
695	12.5
695	12.5
385	39.75
385	39.75
695	12.5
385	39.75
385	39.75

III. Experimental Results

In this section we present the experimental results of our proposed method. We have used a real microdata set called ”Census”. That microdata set contains 1,080 records with 13 numerical attributes.

Our method falls into the category of fixed-size microaggregation heuristics. Therefore, to study the information loss parameter we have varied the value of k in the range $[3, 4, 5, 10]$, we compared the results with those obtained with the MDAV and V-MDAV methods, for the same values of k. After microaggregating the data sets, we have analyzed the information loss introduced by each method.

From table IX, we can see that the results obtained by MDAV and V-MDAV are very similar, but our method has a greater information loss for any value of k, the error introduced grows proportionally to k. After analyzing these results, we can conclude that our method get good results for $k = 3$.

TABLE IX: Experimental results. Information loss caused by MDAV,V-MDAV and our method

Dataset	**Method**	$k = 3$	$k = 4$	$k = 5$	$k = 10$
Census	MDAV	5,66	7,51	9,01	14,07
	V-MDAV	5,69	7,52	8,98	14,07
	MF-TSP	6,00	9,24	11,6	25,49

IV. Conclusion and further work

MF-TSP, a new heuristic method for multivariate microaggregation has been proposed in this article. MF-TSP is a fixed group size method, the level of privacy required is controlled by a parameter k (minimum group size). Once k has been chosen, the data protector is interested in minimizing information loss, in our case that happens whit $k = 3$. A number of research issues remain open and will be addressed in future work,

- Modify the neighborhood search method based on a window $[2k - 1]$ to a new method.
- Modify the algorithm to a variable-size microaggregation method.
- Study groups formed comparing the result with MDAV.

References

[1] Climer, S., and Zhang, W. Rearrangement clustering: Pitfalls, remedies, and applications. *J. Mach. Learn. Res. 7* (Dec. 2006), 919–943.

[2] Domingo-Ferrer, J., Martinez-Ballesté, A., Mateo-Sanz, J., and Sebé, F. Efficient multivariate data-oriented microaggregation. *The VLDB Journal 15* (2006), 355–369.

[3] Domingo-Ferrer, J., Sebé, F., and Solanas, A. A polynomial-time approximation to optimal multivariate microaggregation. *Computers and Mathematics with Applications 55*, 4 (2008), 714 – 732.

[4] for National Statistics, O. Statistical disclosure control for 2011 census.

[5] Hahsler, M., and Hornik, K. Tsp - infrastructure for the traveling salesperson problem. *Journal of Statistical Software 23*, 2 (12 2007), 1–21.

[6] Herranz, J., Matwin, S., Nin, J., and Torra, V. Classifying data from protected statistical datasets. *Computers and Security 29*, 8 (2010), 875 – 890.

[7] Johnson, O., and Liu, J. A traveling salesman approach for predicting protein functions. *Source Code for Biology and Medicine 1* (2006). cited By 12.

[8] Liao, Y.-F., Yau, D.-H., and Chen, C.-L. Evolutionary algorithm to traveling salesman problems. *Computers and Mathematics with Applications 64*, 5 (2012), 788 – 797. Advanced Technologies in Computer, Consumer and Control.

[9] Martinez, S., Sanchez, D., and Valls, A. Semantic adaptive microaggregation of categorical microdata. *Computers and Security 31* (2012), 653–672.

[10] Mateo-Sanz, J., and Domingo-Ferrer, J. A comparative study of microaggregation methods. *QÜESTIIÓ 22*, 3 (1998), 511–526.

[11] Solanas, A., Gonzalez-Nicolas, U., and Martinez-Ballesté, A. A variable-mdav-based partitioning strategy to continuous multivariate microaggregation with genetic algorithms. In *Neural Networks (IJCNN), The 2010 International Joint Conference on* (2010).

[12] Solanas, A., and Martinez-Ballesté, A. Vmdav: A multivariate microaggregation with variable group size. *17th COMPSTAT Symposium of the IASC, Rome* (2006), 917–925.