CONTRIBUTIONS TO STATISTICAL DISCLOSURE CONTROL: ENHANCING MULTIVARIATE MICROAGGREGATION USING GRAPH THEORY

Armando Maya-López

UNIVERSITAT
ROVIRA i VIRGILI

Contributions to Statistical Disclosure Control: Enhancing Multivariate Microaggregation using Graph Theory

---

ARMANDO MAYA-LÓPEZ

DOCTORAL THESIS
2023

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO STATISTICAL DISCLOSURE CONTROL: ENHANCING MULTIVARIATE MICROAGGREGATION USING GRAPH THEORY
Armando Maya-López

Universitat Rovira i Virgili

Department of

Computer Engineering and Mathematics

Doctoral Thesis

Contributions to Statistical Disclosure Control: Enhancing Multivariate Microaggregation using Graph Theory

Author:

Armando Maya-López

Thesis Advisors:

Dr. Fran Casino

Dr. Antoni Martínez-Ballesté

Dr. Agustí Solanas

Doctoral thesis submitted to the Department of Computer Engineering and Mathematics in partial fulfillment of the requirements of the degree of Ph.D in Computer Science

UNIVERSITAT ROVIRA i VIRGILI

Tarragona, 2023

Departament d'Enginyeria
[DΣIM]
Informàtica i Matemàtiques

Avinguda dels Països Catalans, 26
Campus Sescelades
43007, Tarragona

I STATE that the present study, entitled "Contributions to Statistical Disclosure Control: Enhancing Multivariate Microaggregation using Graph Theory", presented by Armando Maya-López for the degree of Doctor of Philosophy in Computer Science, has been carried out under my supervision at the Department of Computer Engineering and Mathematics of this University.

Tarragona, 10th of April 2023

Doctoral Thesis Supervisors

Dr. Fran Casino,
Dr. Antoni Martínez-Ballesté,
Dr. Agusti Solanas

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO STATISTICAL DISCLOSURE CONTROL: ENHANCING MULTIVARIATE MICROAGGREGATION USING GRAPH THEORY
Armando Maya-López

Resum

Aquesta tesi doctoral estudia la microagregació com a tècnica per al control de la divulgació estadística. La investigació té com a objectiu millorar l'eficiència i la qualitat de la microagregació de mida fixa i variable mitjançant l'ús de l'algorisme del problema del venedor ambulant (TSP). L'estudi presenta quatre contribucions: (1) microagregació de mida fixa basada en TSP, (2) microagregació de mida variable basada en TSP (3) tècniques de postprocessament de dades per a l'optimització de conjunts de dades microagregades i (4) estratègies de reducció de conjunts de dades per a microagregació basada en TSP. Els mètodes proposats s'avaluen mitjançant experiments i es comparen amb les tècniques existents. Els resultats de la investigació revelen que els mètodes basats en TSP proposats superen els existents en termes d'utilitat de dades i temps de càlcul. Aquesta tesi proporciona un estudi integral de l'estat de l'art al control de divulgació estadística i ofereix solucions pràctiques per millorar el rendiment de la microagregació per a la publicació de dades preservant la privadesa.

Resumen

Esta tesis doctoral se centra en la microagregación como técnica para el control de la divulgación estadística. La investigación tiene como objetivo mejorar la eficiencia y la calidad de la microagregación de tamaño fijo y variable mediante el uso del algoritmo del problema del vendedor ambulante (TSP). El estudio presenta cuatro contribuciones: (1) microagregación de tamaño fijo basada en TSP, (2) microagregación de tamaño variable basada en TSP, (3) técnicas de postprocesado de datos para optimización de conjuntos de datos microagregados y (4) estrategias de reducción de conjuntos de datos para microagregación basada en TSP. Los métodos propuestos se evalúan a través de experimentos y se comparan con las técnicas existentes. Los resultados de la investigación revelan que los métodos basados en TSP propuestos superan a los existentes en términos de utilidad de datos y tiempo de cálculo. Esta tesis proporciona un estudio integral del estado del arte en el control de divulgación estadística y ofrece soluciones prácticas para mejorar el rendimiento de la microagregación para la publicación de datos preservando la privacidad.

Abstract

This PhD dissertation focuses on microaggregation as a technique for statistical disclosure control. The research aims to enhance the efficiency and quality of fixed-size and variable-size microaggregation through the use of the Travelling Salesman Problem (TSP) algorithm. The study presents four contributions: (1) TSP-based fixed-size microaggregation, (2) TSP-based variable-size microaggregation, (3) data postprocessing techniques for optimization of microaggregated datasets and (4) dataset reduction strategies for TSP-based microaggregation. The proposed methods are evaluated through experiments and compared with existing techniques. The research findings reveal that the proposed TSP-based methods outperform the existing ones in terms of both data utility and computation time. This dissertation provides a comprehensive study of the state-of-the-art in statistical disclosure control, and offers practical solutions to enhance the performance of microaggregation for privacy-preserving data publishing.

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO STATISTICAL DISCLOSURE CONTROL: ENHANCING MULTIVARIATE MICROAGGREGATION USING GRAPH THEORY
Armando Maya-López

Contents

CHAPTER 1

Introduction

This chapter introduces the issues faced in this doctoral thesis. In addition, it briefly describes the solutions we propose to tackle those issues. Finally, the structure and organisation of the thesis are outlined.

Contents

1.1 Motivation

The massive use of information technologies, pervasive electronic devices, and telecommunications in all areas of our society has opened the door to gather huge amounts of data. To obtain information and knowledge from these data [35], new disciplines focused on data analysis have been created, namely Data Science, Data and Process Mining [3], Big Data Analytics, Deep Learning, and so on. Although the collected data might include only small portions of personal and private data, they must be protected. Otherwise, due to the capabilities of big-data-based technologies, sensible information, trends, patterns, and behaviours could be revealed, thus, endangering people's privacy.

The Internet of Things (IoT) is paving the way for the deployment of large and complex, highly-sensorised scenarios that aim to provide a wide range of services: from efficient monitoring and control of the environment to smart transportation, including ambient assisted living, smart buildings, cognitive health and other areas that will become a reality in the coming years. At the bottom level of an IoT application, the *perception layer*: *i.e.* a layer of sensors that collect data at a high rate. At the top- level, the concept of *cloud* stores and provides access to the data while offering a variety of services. 5G communication technologies are the icing on the cake: beyond

providing smartphone users with low-latency, high-bandwidth communications, they will enable the rapid interconnection of virtually thousands of devices. As a result, the IoT ecosystem will generate an unprecedented amount of Big Data, typically in the form of multivariate microdata.

Microdata can be defined as a set of records that contain information about individual respondents or business entities. Consider a microdata set D with p continuous numerical attributes and n records (*i.e.* the result of observing p attributes on n individuals). The attributes in the original microdata set D can be classified into four categories:

- **Direct Identifiers** are variables that uniquely identify statistical the respondent, such as social insurance number, full name, passport,etc.
- **Key variables** are defined as a set of variables that, in combination, can be linked with external information to re-indentify respondents in the microdata set D. Key variables are also called *quasi-identifiers* or *indirect identifiers*. For example, gender, nationality or occupation variables may not reveal the identity of any respondent in a big city, but in a small town, in combination, they may uniquely identify respondents.
- **Sensitive variables** are attributes whose values contain confidential information on the respondent, the concept of confidential is often subject to legal and ethical concerns. Examples are salary, religion, sexual behaviour,etc.
- **Non-confidential variables** are attributes which contain non-sensitive information about the respondent.

These categories, shown in Figure 1.1, are not necessarily disjoint. Note that the variables may be categorical variables where the attribute takes values over a finite set, such as gender, or continuous variables that can be used to perform arithmetic operations on real numbers, such as salary or weight.

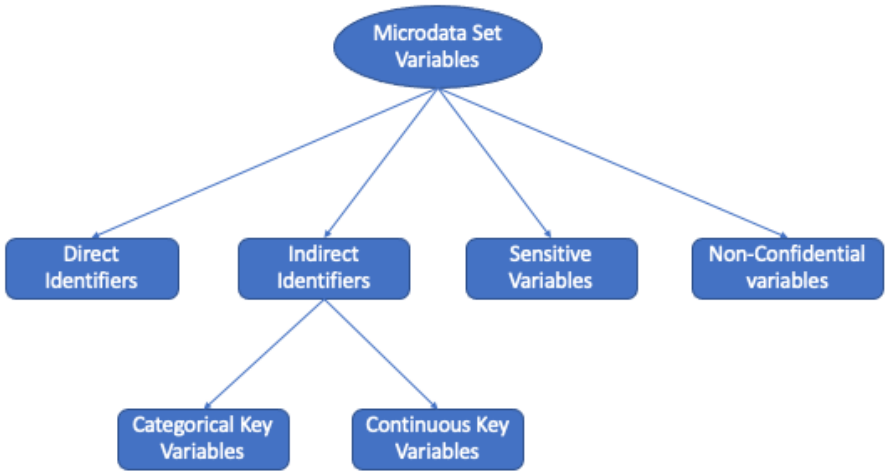


Figure 1.1: Microdata variables categories

Ongoing advances in information and communication technologies (ICT) and efficient processing of data enable the extraction of new knowledge through the discovery of non-obvious patterns and relationships. Nevertheless, such knowledge extraction procedures may threaten individual privacy if proper measures are not taken to protect them [23,43,45]. For instance, an attacker can use publicly available datasets to gain insights about individuals and extract knowledge by exploiting correlations that were not obvious when examining a single dataset [2].

Notwithstanding, *per se*, datasets are not useful unless they are analysed using techniques like data mining (*e.g.* to determine the behaviour of attributes, to identify patterns), process mining (*e.g.* to discover processes, to check the conformance between existing process models and those reflected in the data) and, ultimately, feeding machine learning systems. Companies can analyse *their* data on their own, but they can also delegate (*i.e.* release) datasets to third parties. Note that sensitive information can be inferred from records and values in the dataset: consumer habits, location tracking, health issues, etc. In order to mitigate the *Big Brother* effect, data protection regulations aim at protecting against data misuse, especially those related to individuals. In this line, individuals must be informed about the use and lifecycle of the data. In addition, techniques like anonymisation or pseudonymisation, *i.e.* replacing personally identifiable information (*e.g.* name, car plate number, electrical supply contract numbers) by artificial identifiers, must be applied before releasing data [47]. A wide variety of privacy models and protection mechanisms have been proposed in the liter-

ature to guarantee anonymity (at different levels depending on the utilised model) when disclosing data [47]. Since most privacy protection methods are based on modifying/perturbing/deleting original data, their main drawback is that they negatively affect the utility of the data. Hence, there is a need to find a proper trade-off between data utility and data privacy.

Recognising the aforementioned risks, governments have reformed existing regulations to legally guarantee people's privacy. For example, consider the General Data Protection Regulation (GDPR), which regulates the processing of personal data of individuals in the EU. Once data are collected, they have to be anonymised before Big Data analytics techniques are applied.

One of the best-known disciplines dealing with methods to protect private information is Statistical Disclosure Control (SDC), which aims to anonymise microdata sets in such a way that it is not possible to re-identify the respondent corresponding to a particular record in the published microdata set [9].

SDC is a set of techniques and procedures used to protect the confidentiality of sensitive data in statistical analyses: It aims to ensure that statistical outputs do not reveal any information about individual respondents or entities in the dataset while maintaining the data accuracy and usefulness for analysis. By applying various methods (*e.g.* data suppression, aggregation, and randomisation), these techniques reduce the risk of disclosure while preserving the overall utility of the data and, therefore, can be commonly used in official statistics, public health, finance and the like.

Among these privacy preserving techniques, microaggregation is one of the most consolidated and used because it guarantees the property of k-anonymity. This means that each record in the dataset cannot be distinguished from other $k-1$ records. As a result, the unique identification of a single respondent/individual is impossible [32]. Microaggregation [42] perturbs microdata sets by aggregating the attribute values of groups of k records (fixed-size microaggregation) or *at least* k records (variable-size microaggregation) to reduce the risk of re-identification by achieving $k-$anonymity. It is one of the most commonly used methods of SDC. It is typically applied by statistical agencies to limit disclosure of sensitive microdata and has been used to protect data in a variety of domains, namely healthcare [37], smart cities [21] or collaborative filtering applications [4], to name a few.

Although the univariate microaggregation problem can be optimally solved in polynomial time, optimal multivariate microaggregation is an NP-hard problem [10]. Thus, finding a solution for the multivariate problem

requires heuristic approaches that aim to minimise the amount of data distortion (often measured in terms of information loss), whilst guaranteeing a desired privacy level (typically determined by a parameter k that defines the cardinality of the aggregated groups). Note that the word cluster and group will be used interchangeably throughout the text. In this context, a cluster is a group of data points that share similar characteristics or features.

1.2 Contributions

The main contributions of this dissertation are the following:

1. **Contributions to fixed-size microaggregation:** As mentioned earlier, optimal multivariate microaggregation is an NP-hard problem. This type of problem is significant in computer science and mathematical research because many practical optimisation problems can be reduced to an NP-hard problem, such as microaggregation, where the goal is to maximise the statistical utility of the information publicly disseminated in the microdata sets while minimising the risk of identifying a respondent. In Chapter 3, a new heuristic for solving the microaggregation problem is proposed, which is inspired by the well-known Travelling Salesman Problem and reduces the information loss compared to other known approaches for a fixed cluster size.

2. **Contributions to Variable-size Microaggregation:** Although no optimal solution exists for the multivariate microaggregation problem, there is an optimal solution to the univariate version, known as Hansen and Mukherjee's algorithm. This algorithm requires a set of sorted records to obtain the optimal solution. This sorting for the univariate case is solved in polynomial time. However, for the multivariate case, it becomes an NP-hard problem. There are several proposals in the literature on how to sort a set of records in a multidimensional space and then apply the Hansen and Mukherjee microaggregation algorithm. In Chapter 4, a novel solution for the multivariate microaggregation problem is proposed, inspired by the heuristic solutions of the Travelling Salesman Problem and the use of Hansen and Mukherjee's optimal univariate microaggregation algorithm. We propose that a possible ordering for the records in $\mathbb{R}^p$ is determined by the Hamiltonian path resulting from the solution of the Travelling Salesman Problem (TSP), where the goal is to find the path that traverses all elements of a set only once while minimising the total length of the path. Our intuition is therefore that good heuristic solutions of the TSP (*i.e.* those with

shorter path lengths) would yield a Hamiltonian path that can be used as an ordered vector for Hansen and Mukherjee's optimal univariate microaggregation algorithm, leading to a good multivariate microaggregation solution.

3. **Contributions to Dataset Reduction Strategies:**

   NP-hard problems, such as microaggregation, are decision problems that require a number of operations that grow exponentially with the size of the input, making the solution impossible for large instances. Therefore, researchers often focus on developing efficient algorithms and heuristics to solve or approximate NP-hard problems. However, in many cases, solving these problems can still require significant computational resources and time. Dataset reduction strategies are techniques that can reduce the size or complexity of a dataset while maintaining its integrity and usefulness for analysis. The choice of dataset reduction strategy depends on the specific requirements of the analysis and the nature of the dataset. It is important to carefully consider the potential benefits and limitations of each strategy before applying it.

   In Chapter 5, we present a method to compress a microdata set, which can be used to solve the Travelling Salesman Problem (TSP), allowing us to optimise its resolution cost. This strategy involves compressing the data to reduce its size while preserving the relevant information. To the best of our knowledge, this is the first time that an optimisation method for TSP-based microaggregation heuristics (*i.e.* optimising both the computational time and the quality of the groups) is presented in the literature.

4. **Contributions to Microaggregation Optimisation:**

   Optimisation problems involve finding the best solution among a set of possible solutions to a problem. The goal is to maximise or minimise a certain objective function while satisfying constraints. In many cases, optimisation problems can be solved using various optimisation algorithms, such as gradient descent, simulated annealing, and genetic algorithms.

   A common problem that can arise when solving optimisation problems is the occurrence of a local minimum. A local minimum is a solution that is the best solution in the immediate vicinity of the current solution, but is not the overall best solution for the problem. This can occur when the optimisation algorithm is stuck in a particular region

of the search space instead of exploring the entire space of possible solutions.

When an optimisation algorithm falls into a local minimum, recovering and finding the true global minimum may be difficult. One common way to address this issue is to use techniques such as random restarts, which involve starting the optimisation algorithm multiple times from different initial points. Another approach is to use more advanced optimisation techniques that are less likely to get stuck in local minima, such as particle swarm optimisation, differential evolution, or Bayesian optimisation.

Falling into a local minimum is a major obstacle in solving optimisation problems, and it is crucial to recognise this issue and utilise effective strategies to mitigate it. Chapter 6 presents a novel post-processing technique specifically aimed at improving the results of a well-known microaggregation technique (MDAV, Maximum Distance to Average Vector) and preventing local minima.

1.3 Organisation

This thesis has been organised as follows:

- Chapter 2 provides the reader with fundamental knowledge on Statistical Disclosure Control and microaggregation. Also, it introduces the basics of the Travelling Salesman Problem and an overview of the existing heuristics to solve it.

- Chapter 3 is dedicated to fixed-size microaggregation. A TSP-inspired method to solve the microaggregation problem is described. The new heuristic proposed is illustrated using a microdata set that simulates a study at a hospital. Finally, two datasets were used as benchmarks to show the overall performance.

- Chapter 4 is dedicated to variable-size microaggregation. A TSP tour construction heuristic is used as input to Hansen and Mukherjee's optimal univariate microaggregation algorithm. To practically validate the usefulness of the multivariate microaggregation proposal, it was thoroughly tested on six datasets that serve as benchmarks. A study on the stability of the solutions was carried out using box plot diagrams. Furthermore, the Pearson correlation between the Hamiltonian path length obtained by all the heuristics studied and the SSE of the resulting microaggregation is thoroughly analysed.

- Chapter 5 presents a dataset reduction strategy for microaggregation. A compression strategy for efficient TSP-based microaggregation is described. Moreover, the approach's efficacy was tested on three datasets that serve as benchmarks. The analysis included aspects of the method's computational time and data utility.

- Chapter 6 presents a microaggregation optimisation strategy, Random Cluster Shuffling (RCS), a post-processing technique to improve microaggregation results. This technique is tested using GPS traces collected from two cities, Barcelona and Madrid.

- Finally, Chapter 7 summarises our contributions and describes possible future research lines.

CHAPTER 2

Background and State of the Art

This chapter provides the reader with the essential context needed to understand the topics covered in this dissertation. Section 2.1 introduces the needs that led to the development of the field of Statistical Disclosure Control. Section 2.2 provides the reader with some basic knowledge about microaggregation. Finally, the metrics commonly used to compare different microaggregation methods and the datasets used for benchmarking are presented.

Contents

2.1 Statistical Disclosure Control

Statistical Disclosure Control (SDC) is a cornerstone of data privacy, particularly when publishing sensitive information in a statistical context. SDC methods aim to protect the confidentiality of individuals and organisations by controlling the risk of identifying sensitive information released in statistical outputs.

SDC aims to preserve the statistical properties of datasets whilst minimizing the privacy risks related to the disclosure of confidential information

from individual respondents. SDC is a critical issue in releasing statistical data, as it seeks to protect the confidentiality of sensitive information. One of the most widely used techniques for SDC is microaggregation, which involves aggregating individual-level data into groups or clusters. This method reduces the risk of identification by reducing the precision of the data while preserving its statistical properties. The usual practice in SDC is for data protectors to apply microaggregation to a restricted set of attributes rather than to entire records in a microdata set. From a mathematical point of view, one can say that what is microaggregated are the projections of records on this restricted set of attributes, which may include key attributes and confidential attributes [8].

This chapter provides an overview of microaggregation algorithms, their applications in SDC and their strengths and limitations. The aim of this chapter is to provide a comprehensive understanding of microaggregation as an SDC method and to lay the groundwork for the subsequent chapters of this thesis, in which we examine the latest developments and advances in the field.

2.1.1 Disclosure Taxonomy

Suppose a hypothetical snoop has access to some public microdata and attempts to identify a particular respondent. If the intruder can reveal previously unknown information about a respondent, a *disclosure* has occurred. *disclosure*, also called *re-identification*, can be divided into three categories, as follows:

- **Identity Disclosure** happens when the snooper manages to link a known person to the microdata record. For example, suppose that a microdata set for the city of Tarragona contains an attribute called 'occupation' whose value is 'mayor', this will certainly lead to disclosure. Even more, unusual combinations of key attributes in public microdata records pose a potentially high risk. For example, if a respondent in Tarragona has the combination *nationality* = *German*, *occupation* = *GP* and *gender* = *male*, this person is likely to be unique in the population and therefore at high risk of disclosure.

- **Attribute Disclosure** occurs when the snooper is able to determine some new characteristics of a person from the information available in the disclosed data. For example, imagine a hospital that publishes a microdata set showing that all male patients aged 60 to 63 have COVID -19, then a snooper knows the medical condition.

- **Inferential Disclosure** occurs when the snooper is able to determine the value of a sensitive characteristic of a person from the data released. Inferential disclosure occurs when the sensitive characteristics of an individual can be well predicted from a good model applied to the microdata set released. For example, if a snooper applies a predictive regression model, he could infer an employee's salary.

In the following toy example, we have used functions of the mathematical software R to show how an intruder can infer the value of any feature of a respondent with some accuracy using a linear regression model. For this purpose, we use the Tarragona dataset, a set of real data that includes the numbers of 834 companies. In particular, our attacker wants to infer the variable named 'PAID.UP. CAPITAL' of a company with the following characteristics:

```
> library("sdcMicro")
> intrudersTarragona <- data.frame(Tarragona[10,])
> intrudersTarragona[,-5]
> t(intrudersTarragona)

FIXED.ASSETS           91
CURRENT.ASSETS     183865
TREASURY            22670
UNCOMMITTED.FUNDS   34555
SHORT.TERM.DEBT    149402
SALES              637815
LABOR.COSTS          4085
DEPRECIATION            0
OPERATING.PROFIT     8504
FINANCIAL.OUTCOME       0
GROSS.PROFIT         3117
NET.PROFIT           1955

>
```

Listing 2.1: Information known by snooper

In Listing 2.1 we have taken the company data corresponding to the microdata with position 10 in the Tarragona dataset as data from a company not present in the dataset. For this reason, we have removed the record from the dataset. Once we have taken a microdata from the dataset, we remove the parameter that we want to derive. So we have removed column 5 of the microdata which corresponds to 'PAID.UP. CAPITAL'.

```
> Tarragona <- Tarragona[-10,]
> mod1 <- lm(log(PAID.UP.CAPITAL)    ~  CURRENT.ASSETS
                                     + TREASURY
```

```
                                        + UNCOMMITTED.FUNDS
                                        + LABOR.COSTS
                                        + SHORT.TERM.DEBT
                                        + SALES
                                        + DEPRECIATION
                                        + OPERATING.PROFIT
                                        + FINANCIAL.OUTCOME
                                        + GROSS.PROFIT
                                        + NET.PROFIT,
            data=Tarragona[Tarragona[,"PAID.UP.CAPITAL"] > 0, ])
> s1 <- summary(mod1)
> s1$r.squared
[1] 0.2847402
> exp(predict(mod1, intrudersTaragona))
9977.568
> data(Tarragona)
> Tarragona[10, "PAID.UP.CAPITAL"]
[1] 10000
```

Listing 2.2: Snooper's Estimate.

We see that inferential disclosure is possible in this scenario, because with a model with a value $R^2 = 0.2847$ we can predict the value of the variable of 9977.568, where the real value is 10^4, so we have an accuracy of 99.775%.

2.1.2 Disclosure Risk

Disclosure risk is defined based on assumptions about disclosure scenarios, *i.e.* how the intruder could exploit the released data to reveal information about the respondent. For example, an intruder could do this by linking the released file to another data source that contains the same respondents and identifying variables. Disclosure risk arises when a given dataset is released. The risk r is assumed to take a non-negative real value ($r \geq 0$), and a risk of zero ($r = 0$) means no risk.

One of the most important tasks in SDC is to estimate the disclosure risk of individuals and a global risk for the entire dataset. The concept of uniqueness and the concept of k-anonymity and l-diversity are important and will be outlined first. *Special Uniques Detection Algorithm* (SUDA) extends the concept of k-anonymity. It also searches for uniqueness in subsets of key variables.

Risk measures are important in deciding whether the dataset is sufficiently protected to be released. Certain anonymisation methods can reduce the risk of disclosure if the dataset is not sufficiently protected. In general, methods for determining disclosure risk differ between categorical

and continuous key variables. The calculation of frequency counts serves as the basis for many disclosure risk estimation methods. The frequency count can be calculated for a combination of q variables that gives the distribution of frequency counts.

2.1.3 Principles of k-anonymity, l-diversity and SUDA

Assuming that sample uniques are more likely to be re-identified, one way to protect confidentiality is to ensure that each distinct pattern of key variables is possessed by at least k record in the sample. Formally, a microdata set satisfies k-anonymity for $k > 1$, if, for each combination of values of key attributes, at least k records exist in the dataset with that combination [8]. A typical practice is to set $k = 3$, which ensures that the same pattern of key variables is possessed by at least three records in the sample. Using the above notation, 3-anonymity means that at least 3 records in the microaggregated microdata set share the same value for their variables.

In `R`, the sdcMicro package provides the function `freqCalc()`, which can be used to compute an estimate of the sample and population frequency counts. EIA (a well-known benchmarking microdata set composed of 4092 observations with 15 variables), of which three are categorical variables (STATE, YEAR, MONTH). If we apply `freqCalc()` on the EIA categorical variables, we obtain the results shown below, where we can see that 24 objects violate 3-anonymity.

```
> freqCalc(EIA, keyVars = c("STATE","YEAR", "MONTH"))

 ----------------------------

0 obs. violate 2-anonymity

24 obs. violate 3-anonymity

 ----------------------------

>
```

Listing 2.3: Frequency Counts.

The concept of l-diversity addresses the limitations of k-anonymity. It was introduced as a stronger notion of privacy where a group of observations with the same pattern of key variables is l-diverse if contains at least l distinct values for each group of observations with the same pattern of key variables. There are three different l-diversity types of measures:

- **Distinct l-diversity** as the simplest definition that ensures that at

least l distinct values for the sensitive field in each key.

- **Entropy l-diversity** as the most complex definition, which defines entropy of a key where the fraction of observations that have a sensitive value.

- **Recursive l-diversity** as a compromise definition that ensures the most common sensitive value does not appear too often in a key while less common sensitive values are ensured not to appear too infrequently in the same key.

An alternative approach for defining disclosure risk is based on special uniqueness. An observation is a special unique concerning a variable set Q, if it is a sample unique on Q and a subset of Q. A set of computer algorithms, SUDA, was designed to detect and grade special uniques comprehensively. SUDA take a two-step approach. In the first step, all unique attributes set up to a user-specified size are located for each observation. SUDA considers only *Minimal Sample Uniques* (MSUs), which are unique variable sets without any unique subsets within a sample. Once all MSUs have been found, a SUDA score is assigned to each observation indicating the risk using the size and distribution of MSUs within each observation. The potential risk of the records is determined based on two issues:

1. The smaller the number of variables spanning the MSU within an observation, the higher the risk of the observation.

2. The larger the number of MSUs in an observation, the higher the risk of the observation.

The concept of uniqueness might not apply to continuous key variables, especially those with an infinite range, since almost every dataset record will be identified as unique. In this case, a more applicable method is to assess risk based on record linkages. Essentially, the record linkage approach assesses to what extent records in the perturbed data file can be correctly matched with those in the original data file.

2.2 Microaggregation

Microaggregation is a family of SDC methods for microdata, which use data perturbation as a protection strategy. The general idea is that, given an original microdata file D and a privacy parameter k, a microaggregation process consists in constructing a k- partition of the dataset, *i.e.* a set of disjoint clusters (whose cardinality is between k and $2k-1$) and replacing each original data record by the centroid (*i.e.* the average vector) of the cluster to which it belongs, hence creating a k-anonymous dataset D', this is a dataset where for each combination of values, at least k records exist int the dataset sharing that combination. With the aim to reduce the information loss caused by the aggregation, the clusters are created so that the records in each cluster are similar.

Formally, microaggregation can be defined as follows: Consider a microdata set D with p continuous numerical attributes and n records (*i.e.* the result of observing p attributes on n individuals). Groups (also called clusters or subsets in this context) of D 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, and k a cardinality constraint. Optimal microaggregation is defined as the one yielding a k- partition maximizing the within- groups homogeneity. Optimal microaggregation is an NP -hard problem [10] for multivariate data and it requires heuristic approaches, which can be classified into two main families:

- *Fixed-size microaggregation.* These heuristics yield k-partitions where all subsets/groups have size k, except perhaps one group which has size between k and $2k-1$, when the total number of records is not divisible by k.

- *Variable-size microaggregation.* These heuristics yield k-partitions where all groups have sizes in $(k, 2k-1)$. Note that, it is easy to show that any group with size larger than $(2k-1)$, could be divided in several smaller groups of size between k and $2k-1$ whose overall within-group homogeneity is better than that of the single larger group.

The fixed-size microaggregation heuristic is very computationally efficient due to its simplicity. In contrast, variable-size heuristics can obtain less information loss, since they can adapt the size of the clusters to the structure of the dataset.

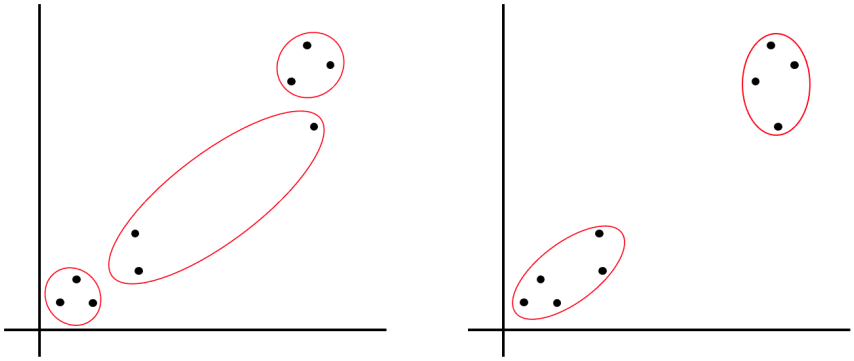


Figure 2.1: Fixed and variable size microaggregation.

Toy example. In this example, we can see the use of microaggregation for k-anonymity [8] [10]. Table 2.1 shows an original microdata set with the name, the surface and the number of employees of 11 companies in a given town.
The 3-anonymous version of this dataset is shown in Table 2.2. 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 standardised to have mean 0 and variance 1 before microaggregation to give them equal weight.

Company Name	**Surface(m^2)**	**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

Table 2.1: Original microdata set used for the toy example

Finally, we can see in Table 2.2, that the 11 records were microaggregated into three groups.

Surface(m^2)	**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

Table 2.2: 3-anonymous version of microdata set for the toy example

In this toy example, there are two attributes that can make a graphical representation in $\mathbb{R}^2$. Figure 2.2 shows the results of three groups created.

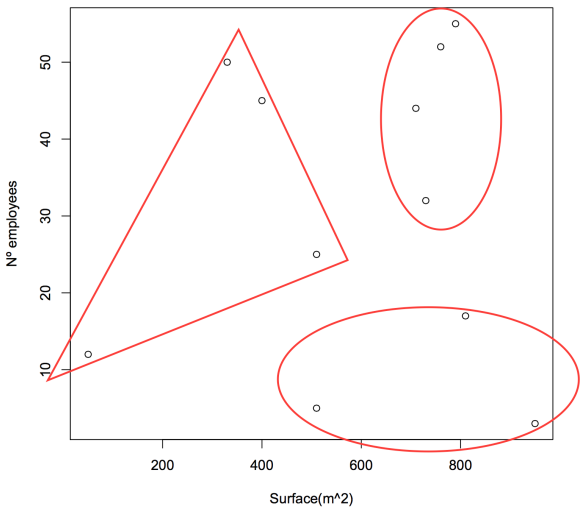


Figure 2.2: Optimal 3-partition of dataset

2.2.1 Microaggregation Techniques

There is a wide variety of heuristics to solve the multivariate microaggregation problem in the literature. One of the most well-known methods is the Maximum Distance to Average Vector (MDAV), proposed by Domingo-Ferrer et al. [8]. This method iteratively creates clusters of k members considering the furthest records from the dataset centroid. A variant of MDAV was proposed by Laszlo et al., namely the Centroid-Based Fixed Size method (CBFS) [17], which also has optimised versions based on kd-tree neighbourhood search, such as KD-CBFS and KD-CBFSapp [40]. The Two Fixed Reference Points (TFRP) method was proposed by Chang et. al [6]. It uses the two most extreme points of the dataset at each iteration as references to create clusters. Differential Privacy-based microaggregation was explored by Yang et al. [44], which created a variant of the MDAV algorithm that uses the correlations between attributes to select the minimum required noise to achieve the desired privacy level. Also, V-MDAV, a variable group-size heuristic based on the MDAV method was introduced by Solanas et al. in [38] with the aim to relax the cardinality constraints of fixed-size microaggregation and allow clusters to better adapt to the data.

Laszlo and Mukherjee [17] approached the microaggregation problem through minimum spanning trees, aimed at creating graph structures that can be pruned according to each node's associated weights to create the groups. Lin et al. proposed a Density-Based Algorithm (DBA) [19], which first forms groups of records in density descending order, and then fine-tunes these groups in reverse order. The successive Group Selection based on sequential Minimization of SSE (GSMS) method [30], proposed by Panagiotakis et al., optimises the information loss by discarding the candidate cluster that minimises the current SSE of the remaining records. Some methods are built upon the HM algorithm. For example, Mortazavi et al. proposed the IMHM method [26]. Domingo-Ferrer et al. [8] proposed a grouping heuristic that combines several methods such as Nearest Point Next (NPN-MHM), MDAV-MHM, and CBFS-MHM.

Other approaches have focused on the efficiency of the microaggregation procedure, for example, the Fast Data-oriented Microaggregation (FDM) method proposed by Mortazavi et al. [25] efficiently anonymises large multivariate numerical datasets for multiple successive values of k. The interested readers can find more detailed information about microaggregation in [11, 47].

Finally, methods based on the advantages of Hansen and Mukherjee's technique for optimal microaggregation of univariate data are further devel-

oped next.

2.2.2 Methods Based on Optimal Univariate Microaggregation

A number of methods are built upon the Hansen and Mukherjee algorithm for optimal univariate microaggregation (HM). In a nutshell, they work as follows: addressing the multivariate records in the dataset (with p columns or variables) as if they were points in a p-dimensional space, first, use a technique to create a path traversing all the records. This path is a permutation of the records in the dataset, which are now reduced to univariate data because only the distance between the records is considered. Afterwards, use this permutation to feed the HM technique, that will create the k-partition. This technique is known as the *Multivariate Hansen-Mukherjee* (MHM) and is described in [13].

The MHM algorithm can be defined as, let $X = \{x_1, x_2, ..., x_n\}$ be an ordered dataset with n records where each record x_i contains the values of p attributes. Let k be an integer group size such that $1 \leq k < n$. Then, a graph $G_{n,k}$ is constructed as follows:

1. For each value x_i in X, create a node with label i. An additional node with label 0 is created.

2. For each pair of graph nodes (i, j) such that $i + k \leq j < i + 2k$, create a directed $arc(i, j)$ from node i to node j.

3. Map each $arch(i, j)$ to the group of values $C_{(i,j)} = \{x_h : i < h \leq j\}$. Let the length $L_{i,j}$ of the arc be the within group sum squares for $C_{(i,j)}$, that is;

$$L_{(i,j)} = \sum_{h=i+1}^{j} (x_h - \bar{x}_{(i,j)})'(x_h - \bar{x}_{(i,j)}) \tag{2.1}$$

   where $\bar{x}_{(i,j)}$ is a p-dimensional record computed as the centroid of records in $C_{(i,j)}$.

Like in the univariate HM algorithm, the k-partition output by MHM is computed as the one whose groups correspond to the arcs in the shortest path between nodes 0 and n.

Lemma 1. *For a fixed path traversing a dataset of multivariate points, MHM yields the best k-partition compatible with the ordering of points induced by the path.*

Proof. The lemma follows from the optimality of the univariate Hansen-Mukherjee algorithm proven in [13] □

Several ways of constructing a path traversing a multivariate dataset can be found in the literature, where each records contain the values of p numerical attributes and are represented as points in $\mathbb{R}^p$. Two main groups of techniques exist for path construction:

1. Method based on connecting nearby points as those proposed by Domingo-Ferrer et al. [8]. In this set of path construction method, except for the first one, namely Nearest Point Next (NPN), the rest of methods are essentially existing fixed-size microaggregation heuristics used for ordering multivariate points, *i.e.* Maximum Distance, MDAV and CBFS.

2. Method based on graph theory, where each record in the dataset represents a vertex in a graph. For the construction of the path, it will be necessary to apply algorithms that connect all vertices, in other words, obtaining a Hamiltonian path over the graph.

The Nearest Point Next method will be described below, since its simplicity allows its use as an example of path construction. The path is constructed as follows:

1. The dataset centroid $\bar{x}$ is computed.

2. Select the most distant record r from $\bar{x}$, which is taken as the first point in the path.

3. The next record selected is the closest to the previous record, and so on until all n records have been added to the path.

Figure 2.3 illustrates the NPN construction on a toy dataset in $\mathbb{R}^2$, where one can be seen that the path constructor starts in the record more distant from the centroid of all records in the dataset.

The NPN construction orders multivariate records based on p-dimensional Euclidean distance without making any assumption on the minimum group size k to be used later by MHM for microaggregation. It can be proved that both, MDAV and CBFS, with $k = 1$ are equivalent to NPN.

In the second group of methods, those based on graph theory, we highlight those that use solutions to the Travelling Salesman Problem. These methods, due to their importance for this thesis, will be detailed in Section 2.2.3.

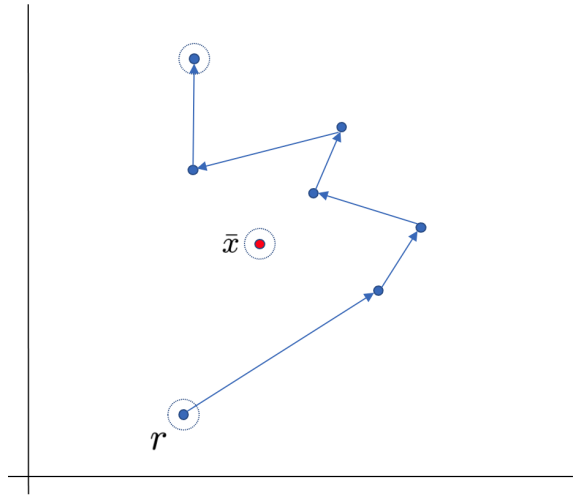


Figure 2.3: Nearest Point Next path constructor

2.2.3 Path construction based on TSP heuristics

The Travelling Salesman Problem (TSP) [34] consists of finding a particular *Hamiltonian cycle* on a weighted graph. The problem can be stated as follows: a salesman leaves from one city and wants to visit (exactly once) each other city in a given group, and finally return to the starting city. The salesman wonders in what order these cities must be visited so as to travel the shortest possible total distance.

In terms of graph theory, the TSP can be modelled by a graph $G = (V, E)$, where cities are the nodes in set $V = \{v_1, v_2, ..., v_n\}$ and each edge $e_{ij} \in E$ has an associated weight w_{ij} representing the distance between nodes i and j. The goal is to find a *Hamiltonian cycle*, *i.e,* a cycle which visits each node in the graph exactly once, with the least total weight. An alternative approach to the *Hamiltonian cycle* to solve the TSP is finding the *Shortest Hamiltonian path* through a graph (*i.e.* a path which visits each node in the graph exactly once). As an example, Figure 2.4 shows a short Hamiltonian path for the *Eurodist* dataset which contains the distance (in km) between 21 cities in Europe.

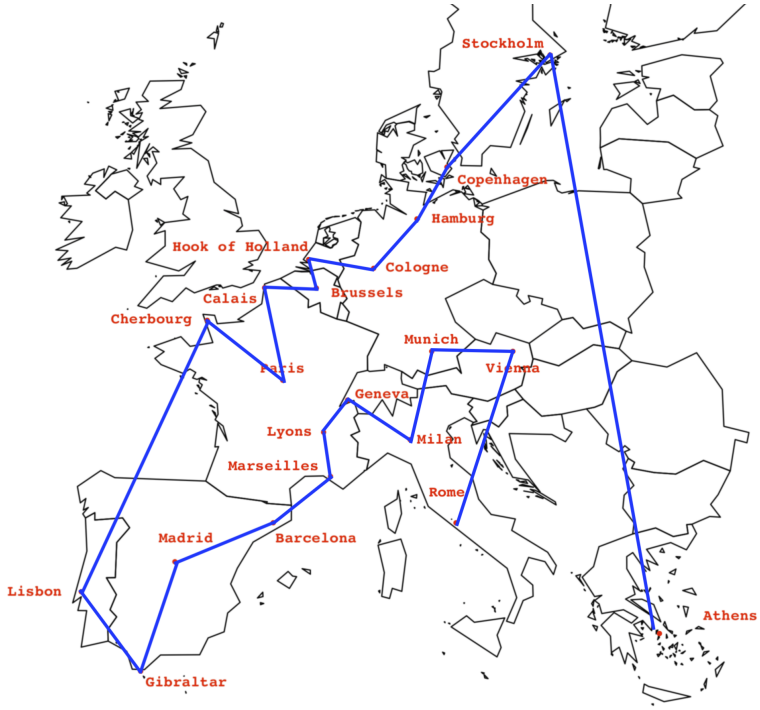


Figure 2.4: A Hamiltonian path for the Eurodist dataset.

Finding an optimal solution to the TSP is known to be NP-Hard. Hence, several heuristics to find good but sub-optimal solutions have been developed. TSP heuristics typically fall into two groups: those involving minimum spanning trees for tour construction, and those with edge exchanges to improve existing tours. There are numerous heuristics to solve the TSP [1, 31]. We have selected a representative sample of heuristics including well-known approaches and top performers from the state-of-the-art.

Within the group *tour construction heuristics*, it is necessary to mention the following:

- *Nearest Neighbour algorithm*: The algorithm starts with a tour containing a randomly chosen node and appends the next nearest node iteratively.

- *Repetitive Nearest Neighbor*: The algorithm is an extension of the Nearest Neighbor algorithm. In this case, the tour is computed n times, each one considering a different starting node and then selecting the best tour as the outcome.

- *Insertion Algorithms*: All insertion algorithms start with a tour that originated from a random node. In each step, given two nodes already inserted in the tour, the heuristic selects a new node that minimises the increase in the tour's length when inserted between such two nodes. Depending on the way such the next node is selected, one can find different variants of the algorithm. For instance, Nearest Insertion, Farthest Insertion, Cheapest Insertion and Arbitrary Insertion.

- *Concorde*: This method is currently one of the best implementations for solving the symmetric TSP. It is based on the *Branch-and-Cut* method to search for optimal solutions.

Tour improvement heuristics are simple local search heuristics which try to improve an initial tour, the most important heuristics in this group are:

- ***k-Opt* heuristics**. The idea is to define a neighborhood structure on the set of all admissible tours. Typically, a tour t' is a neighbor of another tour t if t' can be obtained from t by deleting k edges and replacing them by a set of different feasible edges (a k-Opt move). The resulting tour represents a local optimum which is called k-optimal. Typically, 2-Opt and 3-Opt heuristics are used in practice.

- **LK heuristic**. This heuristic [20] does not use a fixed value for k for its k-Opt moves, but tries to find the best choice of k for each move. The heuristic uses the fact that each k-Opt move can be represented as a sequence of 2-Opt moves.

In figure 2.5, the different heuristics are compared through the calculation of the path length for the Eurodist dataset. A more in-depth study can be found in the work done by Hashsler and Hornik at [12].

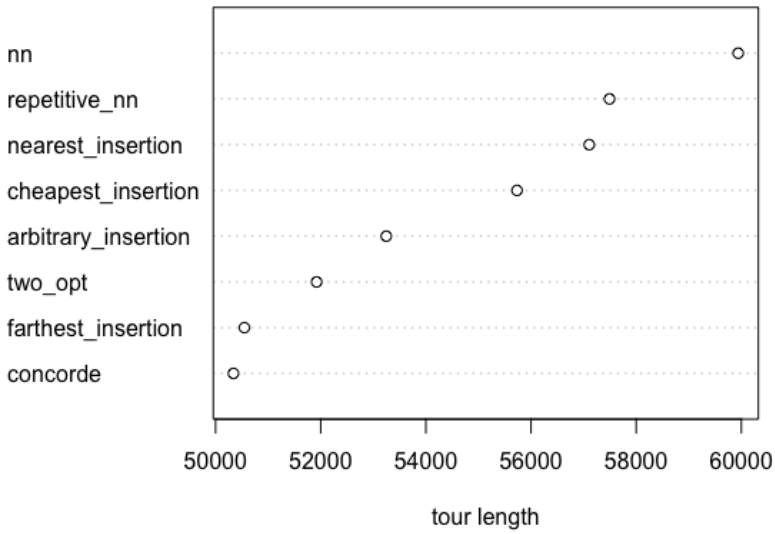


Figure 2.5: Comparison tour lengths for Eurodist dataset

TSP-based microaggregation focus on using the TSP heuristics to generate an ordered sequence in $\mathbb{R}^n$ of the records in a dataset. Such sequence can be used to create a k-partition (by using *e.g.* the MHM method). Heaton and Mukherjee [14] used the TSP tour optimisation heuristics (*e.g.* 2-opt, 3-opt) to refine a path created with the information of a multivariate microaggregation method (*e.g.* MDAV, MD, CBFS).

2.3 Metrics and Benchmarking

Once the microaggregation method has been applied, the original dataset is modified searching for low the disclosure risk, but the data must retain their statistical properties with a low information loss, this is what is known as data utility. In order to conduct tests aiming at comparing techniques and assessing their validity, we need to establish a set of measures. Moreover, tests must be conducted on benchmarking datasets.

The sum of squared errors (SSE) is commonly used for measuring the homogeneity in each group. In terms of sums of squares, maximising within-groups homogeneity is equivalent to finding a k-partition minimising the within-groups sum of square error (SSE) [38] defined as:

$$SSE = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (x_{i,j} - \bar{x_i})(x_{i,j} - \bar{x_i})' \tag{2.2}$$

, where $x_{i,j}$ is the j-th record in group i, and $\bar{x_i}$ is the average record of group i. The total sum of squares (SST), an upper bound on the partitioning information loss, can be computed as follows:

$$SST = \sum_{i=1}^{n} (x_i - \bar{x})(x_i - \bar{x})' \tag{2.3}$$

, where x_i is the i-th record in D and $\bar{x_i}$ is the average record of D. Note that all the above equations use vector notation, so $x_i \in \mathbb{R}^p$.

The microaggregation problem consists in finding a k-partition with minimum SSE, this is, the set of disjoint subsets of D so that $D = \bigcup_{m=1}^{g} s_m$, where s_m is the m-th subset and g is the number of subsets, with minimum SSE. However, a normalised measure of information loss (expressed in percentage) is also used:

$$\mathbf{I_{loss}} = \frac{SSE}{SST} \times 100 \tag{2.4}$$

In terms of information loss, the worst case scenario for microaggregation would happen when all records in D are replaced in D' by the average of the dataset (*i.e.* $SSE = SST \rightarrow I_{loss} = 100$), and the best case scenario implies that $D = D'$ (*i.e.* $k = 1$, no aggregation) which leads to $SSE = I_{loss} = 0$. Obviously, the latter case is optimal in terms of information loss but it offers no privacy protection, at all. Hence, values for the protection parameter k are greater than one, typically: $k = 3, 4, 5, or\ 6$, and are chosen by privacy experts in statistical agencies so as to adapt to the needs of each particular dataset.

We used six datasets as benchmarks for our experiments. We can classify those datasets into two main groups: The first group comprises three well-known SDC microdata sets that have been used for years as benchmarks in the literature, namely "Census", "EIA" and "Tarragona". The second group comprises three mobility datasets containing real GPS traces from three Spanish cities, namely "Barcelona", "Madrid" and "Tarraco"[1]. The features of each dataset are next summarised:

The **Census** dataset was obtained using the public *Data Extraction System of the U.S. Bureau of the Census*. It contains 1,080 records with 13 numerical attributes. The **Tarragona** dataset was obtained from the Tarragona Chamber of Commerce. It contains information on 834 companies in the Tarragona area with 13 variables per record. The **EIA** dataset was obtained from the U.S. Energy Information Authority and it consists of 4092 records with 15 attributes. More details on the aforementioned datasets can be obtained in [41].

The **Barcelona**, **Madrid** and **Tarraco** datasets consist of OpenStreetMap [28] GPS traces collected from those cities: **Barcelona** contains the GPS traces of the city of Barcelona within the area determined by the parallelogram formed by latitude (41.3726866, 41.4078446) and longitude (2.1268845, 2.1903992). The dataset has 969 records with 30 GPS locations each. **Madrid** contains the GPS traces of the city of Madrid within the area determined by the parallelogram formed by latitude (40.387613, 40.483515) and longitude (-3.7398145, -3.653985). The dataset has 959 records with 30 GPS locations each. **Tarraco** contains the GPS traces of the city of Tarragona within the area determined by the parallelogram formed by latitude (41.0967083, 41.141174) and longitude (1.226008, 1.2946691). The dataset has 932 records with 30 GPS locations each. In figure 2.6 a set of GPS traces obtained in the area around Puerta de Atocha station in Madrid published by OpenStreetMap [28] for public use can be seen. In the image on the left we see a satellite photo obtained by GoogleMaps and on the right, each line represents the trajectory of an object collected by OpenStreetMap [28] over the same area.

[1]Notice that we use the term "Tarraco" – old Roman name for the city of Tarragona, in other to avoid confusion with the classic benchmark dataset "Tarragona".

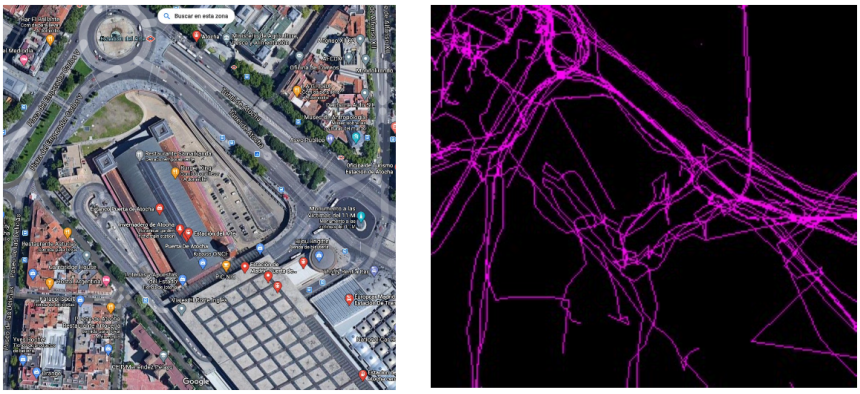


Figure 2.6: OpenStreetMap GPS trace around Madrid Puerta Atocha Station

In all trajectories datasets, each record consists of 30 locations represented as (latitude and longitude), figure 2.7 shows an example of a trajectory represented in a two-dimensional space where the axes represent latitude and longitude and each point represents an object traveling in a plane. Hence, each record has 60 numerical values. These locations were extracted from each corresponding parallelogram according to the amount of recorded tracks and their length. Due to their different density, we considered a subset of the locations collected by each track, so that we could create datasets with the same dimensionality. In the case of Barcelona, we allowed one point every 100 meters. In Madrid, we collected one point every 150 meters. Finally, in Tarraco, due to the lack of tracks, we considered one point every 5 meters.

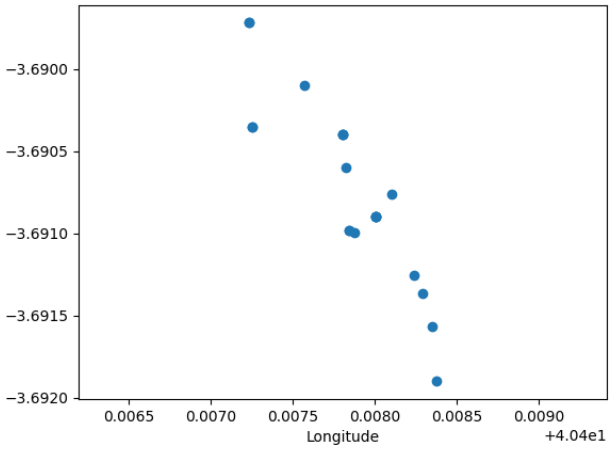


Figure 2.7: Trajectory-based microdata

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO STATISTICAL DISCLOSURE CONTROL: ENHANCING MULTIVARIATE MICROAGGREGATION USING GRAPH THEORY
Armando Maya-López

Chapter 3

Contributions to Fixed-size Microaggregation

As stated in previous chapters, optimal multivariate microaggregation is an NP-hard problem. This type of problem is significant in computer science and mathematical research because many practical optimisation problems can be reduced to an NP-hard problem, such as microaggregation, where the goal is to maximise the statistical utility of the information that is publicly distributed in the microdata sets, minimising the risk of identifying a respondent. This chapter proposes a new heuristic inspired by the well-known Travelling Salesman Problem to solve the microaggregation problem, which reduces the information loss with respect to other well-known approaches for a fixed cluster size.

Contents

3.1 TSP for Fixed-size Microaggregation

The content of this Chapter is related to Multivariate Microaggregation with fixed k-partitions size (Chapter 2). In the next sections, we propose a novel approximation to microaggregation, which is based on the Travelling Salesman Problem (TSP). Therefore, given a microdata set, we apply a TSP heuristic to create a set of paths, which will then be processed iteratively to create a neighbourhood matrix, reflecting the proximity of each record. Next, we apply a clustering process to such microdata to generate a set of groups of at least k records. Finally, the values each group's records of each

group will be substituted by the average value of each corresponding group. At the end of this process, each record will be indistinguishable from at least $k-1$ others.

This is the first work that formally proposes the use of TSP to process an input microdata set and generate a neighbourhood matrix. Moreover, the clustering process that we apply to such data has been adapted to the particularities of the TSP method. The adaptations required to characterise the multivariate microaggregation problem are explained and justified. A running example on a dataset will be shown for a synthetic dataset.

3.2 Our Proposal

In this section, we present our microaggregation algorithm based on the TSP, named Multivariate Fixed-size TSP (MF-TSP). The algorithm has two differentiated steps:

1. The creation of a neighborhood matrix by finding the *Shortest Hamiltonian Paths* and aggregating their corresponding adjacency matrices.

2. The clustering process.

Due to a microdata dataset is a multivariate dataset consisting of n records and p numerical attributes which can be represented as n points $x_1, ..., x_n$ in $\mathbb{R}^p$, MF-TSP represents the records as a graph with the following properties:

- Each microdata are represented as nodes in a weighted and complete graph.

- The weight of each edge represents the Euclidean distance between its nodes.

As we have seen in Chapter 2, finding a *Shortest Hamiltonian Path* on a graph is a NP-hard problem. Therefore, we will use heuristics to approach the solution. Such heuristics are susceptible to changes in the initial conditions, resulting in variations in the order of the nodes when computing a Hamiltonian path to solve the TSP. For instance, we can modify the initial conditions by starting the TSP solver in a different node at each iteration. The method provides us with the probability that an edge will be visited to create the TSP path.

The new microaggregation heuristic proposed in this paper, described in Algorithm 1, works as follows:

1. The first step is the standardization of the original dataset D to give all variables at the same statistical weight.

2. Next, we compute a distance matrix considering each pair of nodes in the original dataset. This matrix represents the adjacency matrix A of the complete weighted graph $W = (W(V), W(E))$ where each edge's weight $w_{i,j}$ represents the distance between nodes i and j. This is a preprocessing step required to calculate the TSP solution, in which the original dataset is represented as a graph (line 2 Algorithm 1).

3. For each node,and given a TSP heuristic, we find the Hamiltonian path that traverses the rest of nodes (*i.e.* $n-1$) in W. At the end of this iteration, a set of n Hamiltonian paths is obtained, namely H_{path}. For instance, $H_{path}(i)$ is the permutation of $\{1, 2, \ldots, n\}$ expressing the order in which the nodes are traversed by the ith Hamiltonian path (line 3-5 Algorithm 1).

4. Given a H_{path} set, build a ***neighbourhood matrix*** C, so that C is a squared matrix $n \times n$, where each element $c_{i,j}$ represents the number of times a node i is connected to a node j in the path. More concretely, we can define matrix C as a summation of a set of adjacency matrices AH, where each adjacency matrix AH_i is created using the ith element of H_{path}. Such procedure is graphically described in Figure 3.1. To build the adjacency matrices, the following possibilities have been considered:

   - If the adjacency matrix represents a Hamiltonian path, all its nodes have degree 2, except the starting node and the ending node which have degree 1, resulting in a semi-eulerian graph. Alternatively, if the adjacency matrix represents a hamiltonian cycle, all its nodes have degree 2. In this case, the starting node and the ending node are connected, resulting in an Eulerian graph.
   - If the graph that represents each element of H_{path} is directed, the adjacency matrix keep memory of the order of the nodes by each permutation, result an asymmetric adjacency matrix. On the opposite, if it is desired to count the adjacent nodes, a non-directed graph must be considered by each permutation existing in H_{path}.

   Therefore, C can be defined as an adjacency meta-matrix of a new graph $C = (C(V), C(E))$, where microdata are the nodes set $C(V) =$

Figure 3.1: For each Hamiltonian path, we generate the corresponding adjacency matrix. Next, all the adjacency matrices are aggregated to create the neighbourhood matrix.

$\{1, 2, \ldots, n\}$ and each edge $c_{i,j} \in C(E)$ represents the probability that the nodes (i, j) will be visited in the set H_{path} of Hamiltonian paths. As described in algorithm 1, lines: 6 - 10

5. Given a graph C, while $\exists\ c_{i,j} \neq 0 \in C(E)$, generate clusters of nodes of size k (*i.e.* k-partition groups). The procedure starts by finding the maximum value $c_{i,j} \in C(E)$, and assigning the nodes (i, j) to the first cluster. Next:

   - The maximum edge value $max(c_{i,x}, c_{y,j}) \in C(E), \forall (X \neq J, Y \neq I) \in C(V)$ is found and node x or y, as appropriate, is added to the cluster. In other words, the next most frequent node connected with i or j is added to the cluster.
   - Next, this procedure is repeated $(k-2)$ times more to create each cluster. Every time that a node is added to a cluster, we delete it from $C - \{C(V)\}$ graph to create disjunct groups.

   As seen in algorithm 1, lines: 12-24

6. When $\nexists\ c_{i,j} \neq 0 \in E$, the connectivity cannot be guaranteed for the resulting graph $C - \{C(V)\}$, because every time a node is added to a cluster, then the node is removed from the graph C and the edges that connect the node. Therefore the disconnected nodes can be found in one of the situations described as follows:

   - Nodes with degree 0 can appear, that is, nodes isolated that do not have edges connecting them to other nodes.
   - Subgraphs with less than k nodes can appear, so they form clusters of less than k elements.

   To assign the disconnected nodes to a cluster, we proceed as follows:

   - Let $R(i)$ be the vector containing the unassigned records, *i.e*, isolated nodes and clusters of size less than k.
   - The C matrix, which was deleted each time a node was added to a group, is restored. The elements in vector $R(i)$ are prevented from being grouped together by assigning a zero value in the matrix C for each pair nodes in R to represented that the nodes are disconnects.
   - Elements in R are added to the groups of k-elements that were created in the previous stage, creating groups of $(2k-1)$ elements.

As described algorithm 1, lines: 25

7. Finally, to obtain a microaggregated dataset D' from D, we compute the centroid (*i.e.* the average vector) of each cluster and replace each record d_i in D by each corresponding centroid $\hat{d}_g$ of the cluster to which it belongs. In algorithm 1, lines: 26

Note that, since all clusters have at most $(2k-1)$ records, the latter does not affect the size constraint imposed by microaggregation (*i.e.* in an optimal k-partition, each cluster must contain a number of records between k and $2k-1$). Finally, figure (3.2) shows the process described in algorithm 1 using a flowchart.

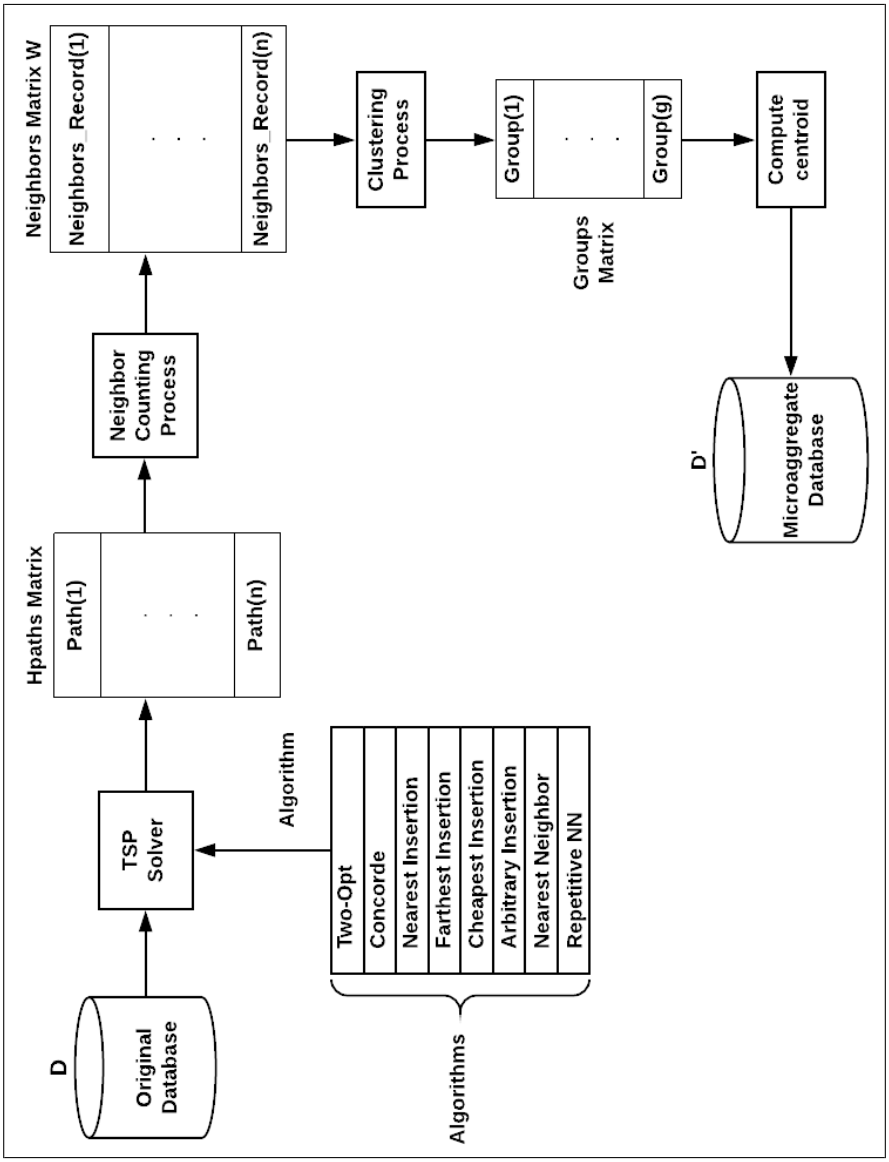


Figure 3.2: TSP based microaggregation process.

3.3 Running Example

The new method, decribed in Algorithm 1, is illustrated by the next running example. The table 3.1 show an illustrative microdata dataset from a study at a fictitious hospital to relate patients' lifestyles to their heart problems. To do this, continuous monitoring is performed by means of an smartwatch that periodically takes the Heart Rate(HR), in the table HR is represented with its maximum and minimum values.

Name	Gender	Age	Height	Weight	Min	Max	ZipCode	Salary	State
Alice Adams	female	59	169	56	50	84	99001	21000	M
Barbara Brown	female	68	181	94	74	86	99002	35000	W
Claire Cooper	female	39	175	78	63	99	99003	42000	M
Destiny Davis	female	32	176	69	61	84	99004	28000	S
Evelyn Evans	female	63	173	74	55	83	98456	24000	W
Fiona Foster	female	56	184	83	67	97	98610	53000	M
Grace Green	female	45	179	61	57	81	98543	80000	S
Hayden Holland	female	27	178	90	62	91	98789	39000	M
Ivy Italy	female	74	165	68	53	82	99009	22000	W

Table 3.1: Cardiac arrhythmia patient monitoring dataset

The records in cardiac arrhythmia microdata dataset can be classified in three categories, as seen in Chapter 1:

1. Identifiers. These are attributes that unambiguously identify the respondent. In our example the attribute 'Name' is a identifier, these data must be pre-processed and eliminated.

2. Key attributes. Also called quasi-identifiers are a set of attributes that can be linked with external information to identify the responders. In the Cardiac arrhytmia dataset, key attibutes are: Gender, Age, Height, Weight, Minimun Heart Rate, Maximun Heart Rate and ZipCode.

3. Confidential outcome attributes. These are attributes which contain sensitive information on the respondent, in Table 3.1 are: Annual Salary and Marital Status.

As indicated in Chapter 2, microaggregation is defined for continuous variables, therefore we will apply our algorithm to the following key attributes: Age, Height, Weight, Minimun Heart Rate and Maximun Heart Rate.

In this section, we will compute the MF-TSP solution for $k = 3$, looking for a 3-anonymous version of the original dataset. To do this, we will apply

Algorithm 1 to the Cardiac Arrhythmia dataset for the different heuristics to solved the TSP that we have presented in Chapter 2.

If we focus on the 'Concorde' algorithm to solve TSP, the MF-TSP heuristic operate in the next steps:

1. Find a Hamiltonian path $H_{path}(n)$ traversing all n microdata in dataset, starting in city n. The result matrix is shown above.

$$H_{path} = \begin{pmatrix} 1 & 9 & 5 & 7 & 8 & 3 & 4 & 6 & 2 \\ 2 & 6 & 3 & 4 & 7 & 1 & 9 & 5 & 8 \\ 3 & 8 & 4 & 7 & 9 & 1 & 5 & 6 & 2 \\ 4 & 8 & 3 & 7 & 1 & 9 & 5 & 6 & 2 \\ 5 & 9 & 1 & 7 & 3 & 6 & 2 & 4 & 8 \\ 6 & 2 & 3 & 1 & 9 & 5 & 7 & 4 & 8 \\ 7 & 6 & 3 & 8 & 4 & 1 & 9 & 5 & 2 \\ 8 & 3 & 5 & 9 & 1 & 7 & 4 & 6 & 2 \\ 9 & 1 & 5 & 2 & 6 & 7 & 4 & 3 & 8 \end{pmatrix} \tag{3.1}$$

2. Search the neighbors that are visited in order, resulting a matrix W_{ij} described in section 3.2, where the term i, j represents the number of time that i appears as a neighbour of j. The W matrix is presented below:

$$W = \begin{pmatrix} 0 & 0 & 0 & 0 & 2 & 0 & 2 & 0 & 5 \\ 0 & 0 & 1 & 1 & 0 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 & 1 & 1 & 1 & 3 & 0 \\ 1 & 0 & 1 & 0 & 0 & 2 & 2 & 3 & 0 \\ 0 & 2 & 0 & 0 & 0 & 2 & 2 & 1 & 2 \\ 0 & 6 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 0 & 1 & 3 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 3 & 2 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.2}$$

3. 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 in column, select the largest and add it to the group.

$$Groups\ MF-TSP_{Concorde} \begin{cases} G_1 = \{1, 5, 9\} \\ G_2 = \{2, 3, 6\} \\ G_3 = \{4, 7, 8\} \end{cases} \tag{3.3}$$

The result $\mathbf{I_{loss}}$ are shown in Table 3.2 :

Method	$\mathbf{I_{loss}}$
MF-TSP(Nearest Insertion)	0.507721
MF-TSP(Farthest Insertion)	0.551541
MF-TSP(Cheapest Insertion)	0.479268
MF-TSP(Arbitrary Insertion)	0.634988
MF-TSP(Nearest Neighbor)	0.390691
MF-TSP(Repetitive NN)	0.523018
MF-TSP(2-Opt)	**0.367941**
MF-TSP(Concorde)	**0.367941**

Table 3.2: Measurement of I_{loss} for different TSP calculation algorithms

We can appreciate that the information loss value I_{loss} for MF-TSP using the 'Concorde' and '2-Opt' algorithms are the best solution. It is worth mentioning the result offered by MF-TSP when the algorithm to compute the TSP is 'Nearest Neighbor', where MF-TSP performs a variable size microaggregation with 2k-1 element in cluster, being the worst result in table 3.2. The cluster created are:

$$\text{Max}(I_{loss}) \text{ cluster} \begin{cases} G_1 = \{6, 2, 3, 5, 8\} \\ G_2 = \{7, 4, 1, 9\} \end{cases} \tag{3.4}$$

Table (3.3) presents the microaggregate dataset.

Gender	**Age**	**Height**	**Weight**	**Min**	**Max**	**ZipCode**
female	65.33	169	66	52.66	83	9****
female	54.33	180	85	68	94	9****
female	54.33	180	85	68	94	9****
female	34.66	177.66	73.33	60	85.33	9****
female	65.33	169	66	52.66	83	9****
female	54.33	180	85	68	94	9****
female	34.66	177.66	73.33	60	85.33	9****
female	34.66	177.66	73.33	60	85.33	9****
female	65.33	169	66	52.66	83	9****

Table 3.3: Cardiac arrhythmia patient microaggregate dataset

In figure (3.3) shows the results of three groups created for Cardiac arrhythmia patient monitoring dataset:

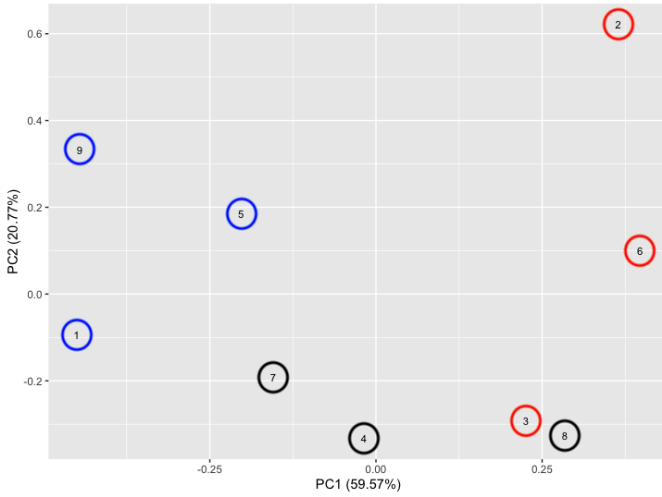


Figure 3.3: Optimal 3-partition for Cardiac Arrhythmia dataset

Note that Cardiac Arrhythmia dataset is a multivariate dataset, hence to have a $\mathbb{R}^2$ representation the principal components must be used.

3.4 Experimental Setup

In this section, we present the experimental results of ***MF-TSP*** heuristics to microaggregate microdata disseminated in statistical databases. We have compared our approach with a well-known microaggregation algorithm called **MDAV** (described in Chapter 2) over two real microdata set which are described in section2.3, on one side ***Census***, on the other side ***Tarragona***. Representing each record as a node in a graph in a multidimensional space $\mathbb{R}^{\mathbf{13}}$, allows us to analyse which techniques to solve TSP problem obtains the minimum Hamiltonian path length.

Our method is a fixed-size microaggregation heuristic. Therefore, to study the information loss for several group sizes, we have varied k in the range $[3, 4, 5, 10]$ (which are the typical values used for statistical agencies), and we compared the results with those obtained by MDAV and V-MDAV, for the same values of k.

The results are shown in Table 3.4, it can be observed that our approach performs better than MDAV and V-MDAV for $k = 3$ in *Census* and *Tarragona* and, for $k = 4$ for *Tarragona*. In a nutshell, we have an initial indication

that our method could lead to better solutions for small values of k while it yields to worse results for larger cardinalities.

3.5 Conclusions

We have proposed a new fixed-size multivariate microaggregation method inspired in the heuristic solutions of the TSP, that helps to guarantee individuals privacy through k-anonimity.

After introducing the basics on Microaggregation and the TSP, we have described our algorithm and we have empirically shown that it performs better than off-the-shelf, well-known microaggregation methods for low cardinalities over benchmark datasets frequently used in the literature. Our proposal represents the first step towards the creation of a more solid TSP-based microaggregation algorithm that would outperform current methods, not only for small cardinalities but for any k as well, and it opens the door to a fruitful research line in the field of SDC.

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(2-Opt)	**5.30**	8.47	10.01	19.13
	MF-TSP(Concorde)	5.34	8.14	10.25	24.82
	MF-TSP(Nearest Insert)	5.38	9.51	11.23	17.01
	MF-TSP(Farthest Insert)	5.68	7.90	11.14	21.12
	MF-TSP(Cheapest Insert)	6.23	8.72	12.46	29.72
	MF-TSP(Arbitrary Insert)	5.55	8.58	10.27	17.84
	MF-TSP(Nearest Neighbour)	6.29	11.08	13.79	46.83
	MF-TSP(Repetitive NN)	6.32	11.80	13.02	43.54
Tarragona	MDAV	16.96	19.70	**22.88**	**33.26**
	V-MDAV	16.96	19.70	22.88	33.26
	MF-TSP(2-Opt)	15.76	**18.68**	23.66	34.29
	MF-TSP(Concorde)	15.36	19.83	24.68	53.99
	MF-TSP(Nearest Insert)	**15.08**	18.92	24.85	36.57
	MF-TSP(Farthest Insert)	19.59	22.49	25.43	38.01
	MF-TSP(Cheapest Insert)	20.28	27.10	33.80	43.64
	MF-TSP(Arbitrary Insert)	15.26	27.09	25.22	36.16
	MF-TSP(Nearest Neighbour)	20.91	27.27	43.34	50.68
	MF-TSP(Repetitive NN)	26.10	33.19	38.07	43.44

Table 3.4: Information loss obtained by MDAV,V-MDAV and our method (MF-TSP)

Algorithm 1 Multivariate Microaggregation with Fixed Group Size Based TSP

function MULTIVARIATE FIXED-SIZE TSP(D Dataset with n records, k Minimum cardinality constraint)
  $W = Compute\_Distances\_Matrix(D)$
  for $i = 1$ to $length(D)$ **do**
    $H_{path}(i) = Compute\_TSP\_Starting\_In(d_i)$;
  end for
  for $i = 1$ to n **do**
    for $j = 1$ to n **do**
      $C = Search\_Neighbords\_inPaths(H_{path}(i), j)$
    end for
  end for
  while $Points\_To\_Assign > (2k - 1)$ **do**
    $max_{i,j} = The\_position\_of\_maximum\_value(C)$
    $g_{cluster} = Build\_Group\_From\_Max(max_{i,j})$
    for $i = 1$ to $(k - 2)$ **do**
      $row\_max_{i,j} = The\_position\_max\_row\_value(C)$
      $col\_max_{i,j} = The\_position\_max\_col\_value(C)$
      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
    $C = Delete\_Assigned\_Points(g_{cluster}, C)$
  end while
  $Assign\_Remaining\_Points(C, g_{cluster})$
  $D' = Compute\_centroid\_Dataset(D, g_{cluster})$
return D'
end function

Chapter 4

Contributions to Variable-size Microaggregation

*Although no optimal solution exists for the multivariate microaggregation problem, an optimal solution exists for the univariate version and is known as the Hansen and Mukherjee algorithm. This algorithm needs a set of sorted records to obtain the optimal solution. This sorting for the univariate case is solved in polynomial time. However, for the multivariate case, it becomes an NP-hard problem. In the literature, several proposals exist on how to sort a set of records in a multidimensional space and then apply the Hansen and Mukherjee microaggregation algorithm. In this Chapter, a novel solution for the multivariate microaggregation problem has been proposed, inspired by the heuristic solutions of the Travelling Salesman Problem and the use of the optimal univariate microaggregation algorithm of Hansen and Mukherjee. Our intuition is that well-performing heuristic solutions of the TSP (*i.e. *those with shorter path lengths) would provide a Hamiltonian path that could be used as an ordered vector for the Hansen and Mukherjee optimal univariate microaggregation algorithm, resulting in a good multivariate microaggregation solution.*

Contents

4.1 TSP for Variable-size Microaggregation

In this Chapter, we propose a heuristic solution to the multivariate microaggregation problem inspired by the TSP and the optimal univariate microaggregation solution HM. Given a multivariate dataset, first, we apply a TSP-tour construction heuristic to generate a Hamiltonian path through all dataset records. Next, we use the order provided by this Hamiltonian path (*i.e.* a given permutation of the records) as input to the HM algorithm, virtually transforming it into a multivariate microaggregation solver we call Multivariate Hansen-Mukherjee (MHM). Our intuition is that good solutions to the TSP would yield Hamiltonian paths allowing the HM algorithm to find good solutions to the multivariate microaggregation problem. We have tested our method with the well-known benchmark datasets defined on section 2.3. Moreover, with the aim to show the usefulness of our approach to protecting location privacy, we have tested our solution with real-life trajectories datasets, too. We have compared the results of our algorithm with those of the best performing solutions, and we show that our proposal reduces the information loss resulting from the microaggregation. Overall, results suggest that transforming the multivariate microaggregation problem into its univariate counterpart by ordering microdata records with a proper Hamiltonian path and applying an optimal univariate HM solution, leads to a reduction of the perturbation error whilst keeping the same privacy guarantees.

The rest of the Chapter aims to answer the research questions above, and it is organised as follows: Section 4.2 describes our proposal, which is later thoroughly tested and compared with well-known classical and state

of the art microaggregation methods in Section 4.3. Section 4.4 discusses the research questions and the main benefits of our proposal. The Chapter concludes in Section 4.5 with some final remarks.

4.2 Our Proposal

Our proposal is built upon two main building blocks: a TSP tour construction heuristic (H), and the optimal univariate microaggregation algorithm HM. As we have already explained in Chapter 2, the HM algorithm is applied to univariate numerical data, because it requires the input elements to be in order. However, we virtually use it with multivariate data and, thus, when we do that we refer to it as Multivariate Hansen-Mukherjee (MHM), although in practice the algorithm is univariate. Since our proposal is based on a Heuristic (H) to obtain a Hamiltonian Path and the MHM algorithm, we have come to call it HMHM-microaggregation or $(HM)^2$-Micro for short.

Given a multivariate microdata set (D) with p columns and r rows, we model it as a complete graph $G(N, E)$, where we assume that each row is represented by a node $n_i \in N$ (or a city, if we think in terms of the TSP) and each edge $e_{ij} \in E$ represents the Euclidean distance between n_i and n_j (or the distance between cities in TSP terms). Hence, we have a set of nodes $N = \{n_1, n_2, \dots n_r\}$ each representing rows of the microdata set in a multivariate space $\mathbb{R}^p$.

In a nutshell, we use H over G to create a Hamiltonian path (H_{path}) that travels across all nodes. H_{path} is a permutation ($\Pi^N = \{\pi_1^N, \pi_2^N, \dots \pi_r^N\}$) of the nodes in N, and *de facto* it determines an order for the nodes (*i.e.* it provides a sense of precedence between nodes). Hence, although D is multivariate, its rows represented as nodes in N can be sorted in a univariant permutation H_{path} that we use as input to the MHM algorithm. As a result, the MHM algorithm returns the optimal univariate k-partition of H_{path}, this is, the set of disjoint subsets $S = \{s_1, s_2, \dots s_t\}$ defining the clusters of N. Hence, since each node n_i represents a row in D, which is indeed multivariate, we have obtained a multivariate microaggregation of the rows in D and provided a solution for the multivariate microaggregation. Notice that, although MHM returns the optimal k-partition of H_{path}, it does not imply that the resulting microaggregation of D is optimal[1]. A schematic of our solution is depicted in Figure 4.1.

Although the foundation of our proposal described above is pretty straightforward, it has the beauty of putting together complex mathematical

[1] Actually, in most cases it is not.

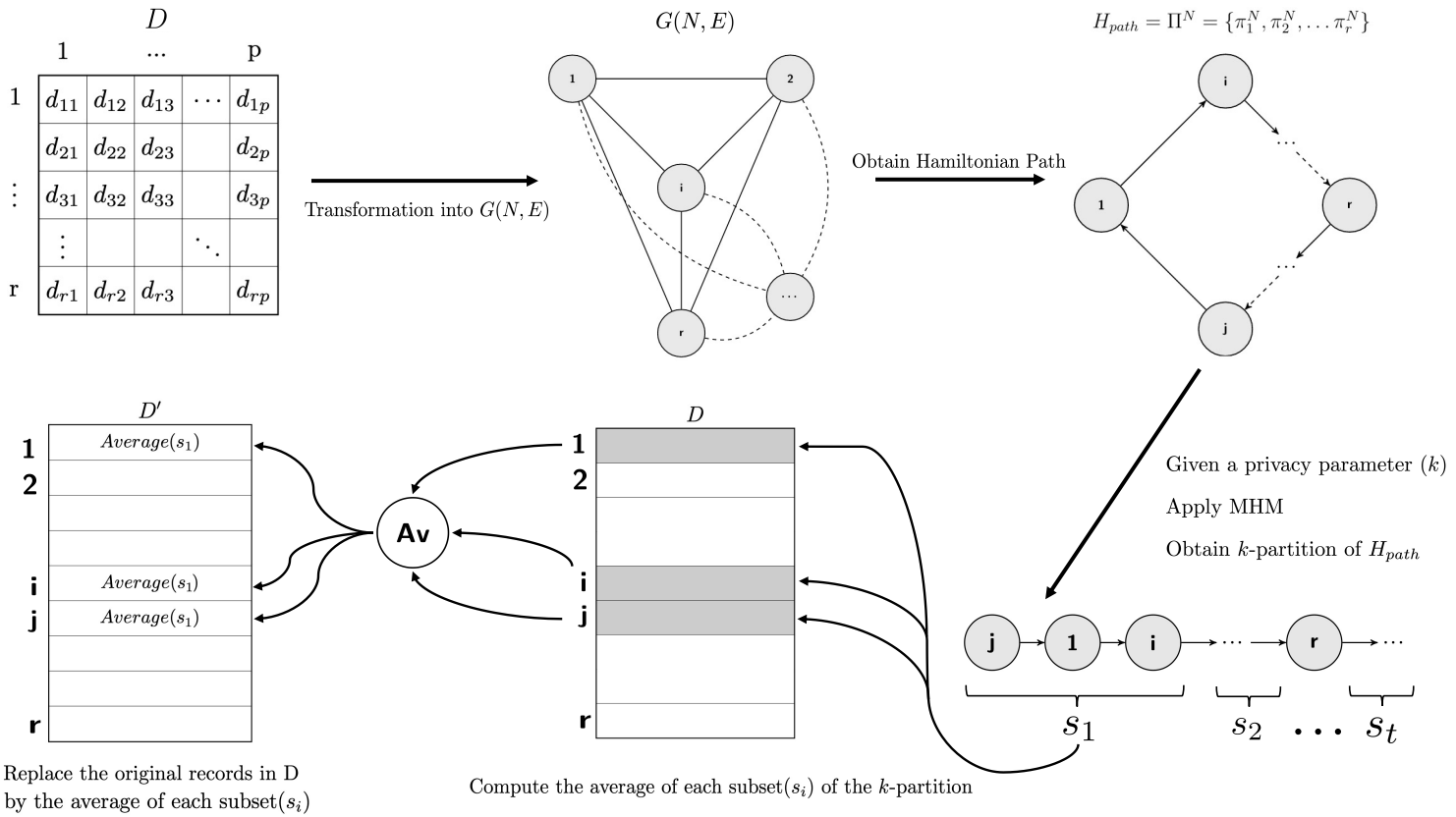


Figure 4.1: Given a microdata dataset, we use a tour construction heuristic to generate a Hamiltonian path, which will be used as the input of the MHM method to generate the groups.

building blocks from the multivariate and univariate worlds in a simple yet practical manner. Also, our solution is very flexible, since it allows the use of any heuristic H to create the Hamiltonian path H_{path}, and it allows for comprehensive studies such as the one we report in the next section.

Note that most TSP heuristics output a Hamiltonian cycle. However, since we need a Hamiltonian path we use the well-known solution of adding a dummy node in the graph (*i.e.* a theoretical node whose distance to all other nodes is zero) and we cut the cycle by eliminating this node, so as to obtain a Hamiltonian path.

For the sake of completeness, we summarise our proposal step-by-step in Algorithm 2, and we next comment on it. Our solution can be seen as a meta-heuristic to solve the multivariate microaggregation problem, since it can accommodate any Heuristic (H) able to create a Hamiltonian cycle from a complete graph (G), and it could deal with any privacy parameter (k). Thus, our algorithm receives as input a numerical multivariate microdata set D with p columns (attributes) and r rows, that have to be microaggregated, a Heuristic H, and a privacy parameter k (see Algorithm 2 : line 1). In order to avoid bias towards higher magnitude variables, the original dataset D (understood as a matrix) is standardised by subtracting to each element the average of its column and dividing it by the standard deviation of the column. The result is a standardised dataset D_{std} in which each column has zero mean and unitary standard deviation (see Algorithm 2 : line 2). Next the distance matrix M_{dist} is computed. Each element $m_{ij} \in M_{dist}$ contains the Euclidean distance between row i and row j in D_{std}, hence M_{dist} is a square matrix $(r \times r)$ (see Algorithm 2 : line 3). In order to be able to cut the Hamiltonian Cycle and obtain a Hamiltonian path, we add a dummy node to the dataset by adding a zero column and a zero row to M_{dist} and generate M_{dist}^{dum}, which is a square matrix $(r+1 \times r+1)$ (see Algorithm 2 : line 4). M_{dist}^{dum} is, in fact, a weighted adjacency matrix that defines a graph $G(N, E)$ with nodes $N = \{n_1, \ldots, n_{r+1}\}$ and edges $E = \{e_{11}, \ldots e_{i,j} \ldots e_{r+1,r+1}\} = \{M_{dist}^{dum}{}_{1,1}, \ldots M_{dist}^{dum}{}_{r+1,r+1}\}$. With this matrix as an input, we could compute a Hamiltonian Cycle H_{cycle} on G by applying a TSP heuristic H (see Algorithm 2 : line 5)[2]. After obtaining H_{cycle}, we cut it by removing the dummy node (see Algorithm 2 : line 6) and we obtain a Hamiltonian path H_{path} that defines a permutation ($\Pi^N = \{\pi_1^N, \pi_2^N, \ldots \pi_r^N\}$) of the nodes in N, and determines an order for the nodes that can be inputted to the MHM algorithm to obtain its optimal k-partition

[2]Notice that this Heuristic H could be anyone that gets as input a weighted graph and returns a Hamiltonian cycle. Some examples are: Concorde, Nearest Neighbour, Repetitive Nearest Neighbour, and Insertion Algorithms.

(S) (see Algorithm 2 : line 7). S is a set of disjoint subsets $S = \{s_1, s_2, \ldots s_t\}$ defining the clusters of nodes in N. Hence, with S and D we could create a microaggregated dataset D' by replacing each row in D by the average vector of the k-partition subset to which it belongs (see Algorithm 2 : line 8).

After applying the algorithm, we have transformed the original dataset D into a dataset D' that has been microaggregated so as to guarantee the privacy criteria established by k.

Algorithm 2 $(HM)^2$-Micro

1: **function** $(HM)^2$-MICRO(Microdata set D, TSP-Heuristic H, Privacy Parameter k)
2: D_{std} = StandardiseDataset(D)
3: M_{dist} = ComputeDistanceMatrix(D_{std})
4: M_{dist}^{dum} = InsertDummyNode(M_{dist})
5: H_{cycle} = CreateHamiltonianCycle(M_{dist}^{dum}, H)
6: H_{path} = CutDummyNode(H_{cycle})
7: S = MHM(H_{path}, D_{std}, k)
8: D' = BuildMicroaggregatedDataSet(D, S);
9: **return** D'
10: **end function**

4.3 Experiments

With the aim to practically validate the usefulness of our multivariate microaggregation proposal, we have thoroughly tested it on six datasets (described in Section 2.3) that serve as benchmarks. Also, we are interested in knowing (if and) to what extend our method outperforms the best performing microaggregation methods in the literature. Hence, we have compared our proposal with these methods (described in Section 4.3.1), and the results of all these tests are summarised in Section 4.3.2. Overall, considering four different values for the privacy parameter $k \in \{3, 4, 5, 6\}$, ten microaggregation algorithms, 50 repetitions per case and six datasets, we have run over 12.000 microaggregation tests, which allow us to provide a statistically solid set of results.

4.3.1 Compared methods

We have selected a representative set of well-known and state-of-the-art methods to assess the value of our approach. We have selected two classic microaggregation methods (*i.e.* **MDAV** and **V-MDAV**), as baselines[3]. Although some other newer methods might have achieved better results, they are still landmarks that deserve to be included in any microaggregation comparison.

For newer and more sophisticated methods, we have considered the work of Heaton and Mukherjee [14], in which they study a variety of microaggregation heuristics, including methods such as CBFS and MD. Thus, instead of comparing our proposal with all those methods, we have taken the method that Heaton and Mukherjee reported as the best performer, namely the **MDAV-LK-MHM** method. This method, which is based on MDAV, first creates a path using the microaggregation method MDAV, next improves the result of MDAV by applying the LK heuristic, and it finally applies MHM to obtain the resulting microaggregation (*cf.* [14] for further details on the algorithm).

Regarding our proposal (*i.e.* $(HM)^2$**-Micro**), as we already discussed, it can be understood as a meta-heuristic able to embody any heuristic H that returns a Hamiltonian Cycle. Hence, with the aim to determine the best heuristic, we have analysed seven alternatives, namely **Nearest Neighbour**, **Repetitive Nearest Neighbour**, **Nearest Insertion**, **Farther Insertion**, **Cheapest Insertion**, **Arbitrary Insertion**, and (our sugges-

[3]In the case of V-MDAV, the method was run for several values of $\gamma \in \{0, 2\}$ and the best result is reported.

Method		Cardinality	Computational Cost	Reference
MDAV		fixed	$O(n^2/2k)$	[8]
V-MDAV		variable	$O(n^2)$	[38]
MDAV-LK-MHM		variable	$O(n^2/2k)$	[14]
$(HM)^2$-Micro	**TSP Heuristic + MHM**			
	Nearest Neighbour	variable	$O(n^2)$	[31]
	Repetitive Nearest-Neighbour	variable	$O(n^2 \log n)$	[31]
	Nearest Insertion	variable	$O(n^2)$	[27]
	Farthest Insertion	variable	$O(n^2)$	[27]
	Cheapest Insertion	variable	$O(n^2)$	[27]
	Arbitrary Insertion	variable	$O(n^2)$	[27]
	Concorde	variable	$O(Mb^d)$	[24]

Table 4.1: Comparing methods and features. For Concorde, M is a bound on the time to explore subproblems, b is a branching factor, and d is a search depth.

tion) **Concorde**. Table 4.1 summarises some features of all selected methods, including the reference to the original article where the method was described. For our method, each reference points to the article describing the TSP heuristic.

The implementation of all these methods have used the *R* package *sdcMicro* [41], the TSP heuristics implemented in [12], and the LK heuristics implemented in [15]. LK has been configured so that the algorithm runs once at each iteration[4] until a local optimum is reached. This same criteria was followed for the other TSP heuristics. In this regard, the heuristics we used consider a random starting node at each run. Hence, each experiment has been repeated 50 times to guarantee statistically sound outcomes regardless of this random starting point.

4.3.2 Results overview

By using the datasets and methods described above, we have analysed the Information Loss (expressed in percentage), as a measure of data utility (*cf.*, Chapter 2 for details). It is assumed that given a privacy parameter k that guarantees that the microaggregated dataset is k-anonymous, the lower the Information Loss the better the result and performance of the microaggregation method. The results are reported in Tables 4.2-4.7 with the best (lowest) information loss highlighted in green.

Overall, it can be observed that, our method, $(HM)^2$-Micro, with the Concorde heuristic is the best performer in 79% of the experiments, and it is the second best in the remaining 21% (for which the MDAV-LK-MHM outperforms it by a narrow margin of less than 2%). Interestingly enough, although $(HM)^2$-Micro, with both Nearest Insertion and Farthest-Insertion, is not the best performer in any experiment, it outperforms MDAV-LK-MHM 50% of the times. The rest of the methods obtain less consistent results and highly depend on the dataset.

When we analyse the results more closely for each particular dataset, we observe that in the case of the "Census" dataset (*cf.,* Table 4.2), our method with Concorde outperforms all methods for all values of k. Also, despite the random nature of TSP-heuristics, the values of σ are very stable, denoting the robustness of all methods, yet slightly higher on average in the case of the methods with higher Information Loss. It is worth emphasising though, that in all runs, our method with Concorde and the MDAV-LK-MHM method obtained better results than MDAV and V-MDAV (*i.e.* the `max` values obtained in all runs are lower than the outcomes obtained by

[4] parameter `RUN`=1.

	Census																	
	k = 3				**k = 4**				**k = 5**				**k = 6**					
Method	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**
MDAV	5.6922	NA	NA	NA	7.4947	NA	NA	NA	9.0884	NA	NA	NA	10.3847	NA	NA	NA
V-MDAV	5.6619	NA	NA	NA	7.4947	NA	NA	NA	9.0070	NA	NA	NA	10.2666	NA	NA	NA
MDAV-LK-MHM	5.1085	0.0398	5.0256	5.1877	6.9131	0.0526	6.7774	7.0227	8.5199	0.0842	8.3100	8.7030	9.9752	0.1284	9.7675	10.2527
Nearest Insertion - MHM	5.6561	0.1369	5.3596	6.0695	7.4818	0.1579	7.1946	7.9318	8.9617	0.2539	8.5190	9.4727	10.3005	0.2927	9.7624	11.2086
Farthest Insertion - MHM	5.5638	0.0956	5.3300	5.8995	7.3485	0.0990	7.1723	7.5853	8.8234	0.1322	8.5784	9.1748	10.1250	0.1932	9.6970	10.7363
Cheapest Insertion - MHM	5.7044	0.0719	5.5669	5.8766	7.4625	0.1155	7.2674	7.8052	9.0340	0.1236	8.7212	9.3847	10.3787	0.1305	10.1706	10.9089
Arbitrary Insertion - MHM	5.5883	0.0976	5.4235	5.8763	7.3723	0.1438	7.1272	7.8250	8.8696	0.1788	8.5072	9.2867	10.2011	0.2475	9.7081	10.7794
Nearest Neighbour-MHM	6.9718	0.3508	6.1978	7.7291	9.2433	0.3702	8.6744	10.2246	11.3287	0.3854	10.5230	12.3958	13.1357	0.4053	12.4711	13.9421
Repetitive NN - MHM	6.2888	0.2192	5.8811	6.6841	8.6779	0.2799	7.9941	9.3345	10.7518	0.2472	10.3421	11.4554	12.5882	0.3143	11.9360	13.2915
Concorde - MHM	**5.0563**	0.0377	4.9917	5.1169	**6.8846**	0.0555	6.7895	7.0217	**8.4576**	0.0903	8.2372	8.6614	**9.8440**	0.1232	9.5542	10.2517

Table 4.2: Information Loss obtained on the Census dataset.

MDAV and V-MDAV).

For the "EIA" dataset (*cf.*, Table 4.3), MDAV-LK-MHM is the best performer for all values of k except $k = 5$, for which our proposal with Concorde performs better. In this case, the results obtained by these two methods are very close. Similarly to the results in "Census", the `max` values obtained by these two methods outperform MDAV and V-MDAV. In the case of "Tarragona", (*cf.,* Table 4.4), our method with Concorde outperforms all other methods. Surprisingly, both MDAV and V-MDAV obtain better results than MDAV-LK-MHM, which performs poorly in this dataset.

So, it can be concluded that the overall winner for the classical benchmarks (*i.e.* Census, EIA and Tarragona) is our method, $(HM)^2$-Micro, with the Concorde heuristic, that is only marginally outperformed by MDAV-LK-MHM in the EIA dataset.

Regarding the other three datasets containing GPS traces (*i.e.* Barcelona, Madrid and Tarraco) our method, $(HM)^2$-Micro, with the Concorde heuristic, is the best performer in 83% of the cases, and comes second best in the remaining 17%. For the Barcelona dataset (*cf.,* Table 4.5) MDAV-LK-MHM and $(HM)^2$-Micro, with the Concorde heuristic, perform very well and similarly. The methods with the worst Information Loss are MDAV and V-MDAV. Our method, $(HM)^2$-Micro, with the Insertion heuristics have a remarkable performance, obtaining values similar to those of MDAV-LK-MHM and Concorde. Nevertheless, it is worth noting that the max (worst) values obtained by MDAV-LK-MHM and Concorde are still better than the averages obtained by the other methods. In the case of the Madrid dataset (*cf.,* Table 4.6) our method, $(HM)^2$-Micro, with the Concorde heuristic achieves the minimum (best) value of Information Loss for all values of k. We can also observe that our method with Insertion heuristics offers higher performance than MDAV-LK-MHM. Finally, the results for the Tarraco dataset (*cf.,* Table 4.7) show that the minimum (best) Information Loss value is obtained by our method with the Concorde heuristic in all cases. In this case, MDAV-LK-MHM performs poorly and for $k = 3$ and $k = 4$, MDAV and V-MDAV are better.

4.3.3 Information Loss variability box plots

We have already discussed that all studied methods (with the exception of MDAV and V-MDAV) have a non-deterministic component emerging from the random selection of the initial node. This random selection affects the performance of the final microaggregation obtained. With the aim to analyse the effect of this non-deterministic behaviour we have studied the standard

	EIA																	
	k = 3				**k = 4**				**k = 5**				**k = 6**					
Method	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**		
MDAV	0.4829	NA	NA	NA	0.6713	NA	NA	NA	1.6667	NA	NA	NA	1.3078	NA	NA	NA		
V-MDAV	0.4829	NA	NA	NA	0.6713	NA	NA	NA	1.2771	NA	NA	NA	1.2320	NA	NA	NA		
MDAV-LK-MHM	**0.3741**	0.0075	0.3659	0.4097	**0.5251**	0.0116	0.5117	0.5693	0.7890	0.0336	0.7502	0.8932	**1.0430**	0.0289	1.0033	1.1113		
Nearest Insertion - MHM	0.4061	0.0114	0.3831	0.4238	0.5781	0.0241	0.5441	0.6179	0.8621	0.0456	0.8032	0.9760	1.1254	0.0837	0.9976	1.3334		
Farthest Insertion - MHM	0.4070	0.0119	0.3872	0.4207	0.5878	0.0251	0.5524	0.6277	0.8764	0.0522	0.8190	0.9747	1.1776	0.0359	1.1245	1.2484		
Cheapest Insertion - MHM	0.5254	0.0358	0.4692	0.5651	0.7321	0.0641	0.6322	0.8477	1.0868	0.0689	0.9910	1.2264	1.4061	0.1147	1.2605	1.6329		
Arbitrary Insertion - MHM	0.4281	0.0300	0.3921	0.4944	0.6092	0.0376	0.5566	0.6699	0.9048	0.0840	0.8194	1.0621	1.1928	0.1077	1.0652	1.3476		
Nearest Neighbour-MHM	0.9028	0.1455	0.5089	1.1023	1.1510	0.1675	0.7056	1.3776	1.4015	0.1788	0.9451	1.6767	1.6792	0.1107	1.4635	1.9139		
Repetitive NN - MHM	0.5110	0.0532	0.4725	0.6599	0.7192	0.0557	0.6646	0.8619	1.0072	0.0701	0.9274	1.1126	1.3101	0.1521	1.1561	1.4825		
Concorde - MHM	0.3889	0.0203	0.3673	0.4210	0.5288	0.0170	0.5087	0.5576	**0.7802**	0.0267	0.7581	0.8501	1.0476	0.0282	1.0009	1.0904		

Table 4.3: Information Loss obtained on the EIA dataset.

	Tarragona																	
	k = 3				k = 4				k = 5				k = 6					
Method	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**
MDAV	16.9326	NA	NA	NA	19.5460	NA	NA	NA	22.4619	NA	NA	NA	26.3252	NA	NA	NA
V-MDAV	16.6603	NA	NA	NA	19.5460	NA	NA	NA	22.4619	NA	NA	NA	26.3252	NA	NA	NA
MDAV-LK-MHM	18.7969	1.8738	15.0595	23.0830	22.8523	1.7576	19.1195	26.2806	26.2432	1.5066	23.0421	28.9522	28.5244	1.7742	25.1703	30.9656
Nearest Insertion - MHM	15.9687	0.8360	15.1107	20.1835	19.3677	1.3141	17.8032	24.5286	23.7323	1.4376	21.8365	28.9753	26.9018	1.5674	24.6538	33.0785
Farthest Insertion - MHM	15.7634	0.2062	15.4743	16.6623	19.0323	0.5521	18.1062	20.2105	22.8316	0.7636	21.3313	24.1988	25.7627	0.4496	24.9004	26.9613
Cheapest Insertion - MHM	16.3142	1.4861	15.2169	22.0271	19.7784	1.6060	18.3103	25.8916	23.9017	1.7155	22.3121	30.0828	27.5572	1.6611	25.2394	32.7082
Arbitrary Insertion - MHM	16.0918	0.7527	15.1310	18.9668	19.5461	1.3436	18.2072	25.8572	23.7685	1.3985	21.7333	29.1863	27.0419	1.6872	25.0093	33.2382
Nearest Neighbour-MHM	22.3019	0.8866	19.9620	23.5496	27.1002	1.2234	24.2527	29.5117	30.4478	1.5455	27.7026	33.3513	34.5445	1.2088	31.3302	37.5350
Repetitive NN - MHM	17.6981	1.2157	15.7435	20.9981	22.1232	1.9138	20.0839	28.7399	27.9089	1.7946	25.1434	32.5729	30.4085	1.9216	28.0648	35.2458
Concorde - MHM	**14.7677**	0.0858	14.6294	14.9633	**17.9957**	0.1241	17.7528	18.2211	**21.9895**	0.2164	21.6712	22.3479	**25.3459**	0.2061	24.8045	25.6564

Table 4.4: Information Loss obtained on the Tarragona dataset.

	Barcelona																	
	k = 3				**k = 4**				**k = 5**				**k = 6**					
Method	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**		
MDAV	2.5667	NA	NA	NA	3.5023	NA	NA	NA	4.2849	NA	NA	NA	5.1873	NA	NA	NA		
V-MDAV	2.5667	NA	NA	NA	3.3193	NA	NA	NA	4.2849	NA	NA	NA	5.1873	NA	NA	NA		
MDAV-LK-MHM	**1.6251**	0.0362	1.5637	1.7425	**2.1913**	0.0339	2.1170	2.2738	2.6798	0.0607	2.5156	2.8067	3.2120	0.0664	3.0731	3.3825		
Nearest Insertion - MHM	1.8022	0.0656	1.6857	1.9438	2.3526	0.0842	2.1754	2.5050	2.8405	0.1008	2.6417	3.0411	3.3316	0.1083	3.1093	3.5103		
Farthest Insertion - MHM	1.7838	0.0525	1.6967	1.8980	2.3575	0.0698	2.1919	2.4681	2.8386	0.0751	2.6654	2.9670	3.3189	0.1131	3.1112	3.6445		
Cheapest Insertion - MHM	1.8156	0.0565	1.6887	1.9293	2.3880	0.0912	2.2354	2.5473	2.8887	0.0792	2.7807	3.0405	3.4118	0.1238	3.1938	3.6247		
Arbitrary Insertion - MHM	1.8061	0.0635	1.6823	1.9469	2.3593	0.0749	2.1808	2.5414	2.8231	0.0911	2.6338	3.0251	3.3331	0.1085	3.1031	3.5584		
Nearest Neighbour-MHM	2.2019	0.1202	1.9165	2.4476	2.9274	0.1778	2.5276	3.3377	3.4733	0.2168	3.0611	3.9399	4.1053	0.2590	3.5159	4.6420		
Repetitive NN - MHM	2.0091	0.0563	1.8899	2.2547	2.7474	0.0611	2.6108	3.0130	3.2318	0.1001	3.1176	3.5701	3.8877	0.1220	3.7106	4.1982		
Concorde - MHM	1.6829	0.0375	1.6210	1.7848	2.2132	0.0534	2.1138	2.3426	**2.6786**	0.0627	2.4974	2.8268	**3.1075**	0.0718	2.9588	3.2348		

Table 4.5: Information Loss obtained on the Barcelona dataset.

	Madrid																	
	k = 3				k = 4				k = 5				k = 6					
Method	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**
MDAV	3.1876	NA	NA	NA	4.3353	NA	NA	NA	5.2883	NA	NA	NA	5.8235	NA	NA	NA
V-MDAV	3.1876	NA	NA	NA	4.3353	NA	NA	NA	5.2883	NA	NA	NA	5.8235	NA	NA	NA
MDAV-LK-MHM	2.9872	0.1285	2.7200	3.1946	4.0536	0.1398	3.6804	4.3314	4.8541	0.1664	4.4680	5.1856	5.5703	0.2163	5.0931	6.0088
Nearest Insertion - MHM	2.7511	0.0814	2.5782	2.9116	3.7039	0.1122	3.4304	3.9623	4.4522	0.1535	4.1533	4.8463	5.1544	0.1549	4.8661	5.5510
Farthest Insertion - MHM	2.6683	0.0558	2.5319	2.8280	3.6187	0.0742	3.4605	3.7755	4.3338	0.1131	4.1260	4.5668	5.0598	0.1172	4.8391	5.3372
Cheapest Insertion - MHM	2.7833	0.0749	2.6517	2.9789	3.7531	0.0804	3.5253	3.9830	4.4752	0.1140	4.3163	4.7356	5.2496	0.1345	5.0147	5.5609
Arbitrary Insertion - MHM	2.7476	0.0757	2.6009	2.9160	3.7156	0.0986	3.5213	3.9828	4.4149	0.1420	4.0583	4.7078	5.1070	0.1437	4.7687	5.3754
Nearest Neighbour-MHM	3.4257	0.1714	3.0816	3.9040	4.7553	0.2116	4.2823	5.3736	5.7671	0.2194	5.1807	6.3191	6.7615	0.2507	6.1871	7.4355
Repetitive NN - MHM	3.1236	0.1345	2.8799	3.5430	4.4141	0.1482	4.1254	5.0012	5.3911	0.2127	5.0894	6.1676	6.4865	0.2223	6.1764	7.3492
Concorde - MHM	**2.4845**	0.0336	2.4053	2.5728	**3.4302**	0.0466	3.3249	3.5664	**4.1124**	0.0774	3.9816	4.3228	**4.8066**	0.1065	4.6538	5.0534

Table 4.6: Information Loss obtained on the Madrid dataset.

	Tarraco																	
	k = 3				k = 4				k = 5				k = 6					
Method	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**	**average**	**σ**	**min**	**max**
MDAV	0.9988	NA	NA	NA	1.4180	NA	NA	NA	1.7683	NA	NA	NA	2.0260	NA	NA	NA
V-MDAV	0.9988	NA	NA	NA	1.3093	NA	NA	NA	1.7182	NA	NA	NA	2.0051	NA	NA	NA
MDAV-LK-MHM	1.1365	0.0154	1.0979	1.1465	1.4216	0.0203	1.4115	1.4723	1.7201	0.0401	1.6995	1.8257	2.0238	0.0404	2.0061	2.1247
Nearest Insertion - MHM	0.9113	0.0345	0.8490	1.0100	1.2634	0.0745	1.1052	1.4306	1.5988	0.1160	1.4220	1.8839	1.9105	0.1517	1.7018	2.2870
Farthest Insertion - MHM	0.9190	0.0368	0.8582	1.0268	1.2217	0.0490	1.1123	1.3755	1.5040	0.0581	1.3965	1.7118	1.8346	0.0612	1.7533	2.1299
Cheapest Insertion - MHM	0.9500	0.0406	0.8975	1.0962	1.2951	0.0557	1.2270	1.4637	1.6200	0.0870	1.5225	1.8677	1.9704	0.1094	1.8584	2.2471
Arbitrary Insertion - MHM	0.9258	0.0455	0.8589	1.0269	1.2530	0.0753	1.1419	1.4538	1.5695	0.0971	1.4454	1.8312	1.9051	0.1265	1.7475	2.3396
Nearest Neighbour-MHM	1.5080	0.1937	1.1624	2.0189	2.1341	0.2232	1.5881	2.6725	2.6499	0.2671	2.0802	3.2271	3.3041	0.4123	2.6557	4.3884
Repetitive NN - MHM	1.2177	0.1286	1.0276	1.5906	1.7806	0.1599	1.4244	2.1131	2.2545	0.1882	1.9146	2.7394	2.7384	0.2209	2.3073	3.4314
Concorde - MHM	**0.8482**	0.0179	0.8167	0.9005	**1.1031**	0.0324	1.0739	1.2348	**1.3805**	0.0556	1.3275	1.6813	**1.7280**	0.0652	1.6610	2.1308

Table 4.7: Information Loss obtained on the Tarraco dataset.

deviation of all methods for all values of k and for all datasets. Also, we have visually inspected the variability of the results by using box plot diagrams.

Since the results are quite similar and consistent across all datasets, for the sake of clarity we only reproduce here the box plots for the "Census" dataset (see Figures 4.2-4.5).

In Figures 4.2-4.5, we can observe that the Information Loss values increase with k but all methods have the same behaviour regardless of the value of k. Also, it is clear that the most stable methods are $(HM)^2$-Micro with Concorde, and MDAV-LK-MHM. The same behavior can be observed for the rest of dataset (see Figures 4.10-4.25).

Overall, we observe some expected differences depending on the datasets. However, the behaviour of the best performing methods is stable. Particularly, the datasets with GPS traces (*i.e.* Barcelona, Madrid, and Tarraco) show more stable results. In summary, the best method was our $(HM)^2$-Micro with Concorde, exhibiting the most stable results across all datasets.

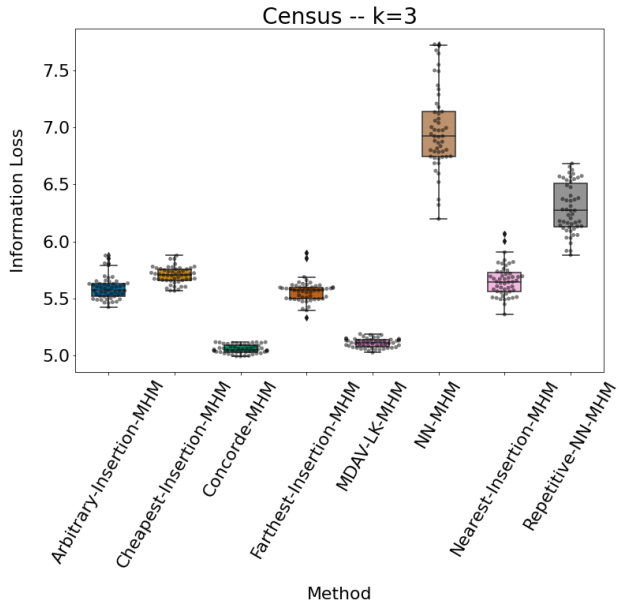


Figure 4.2: Information Loss variability for $k = 3$ over the Census dataset.

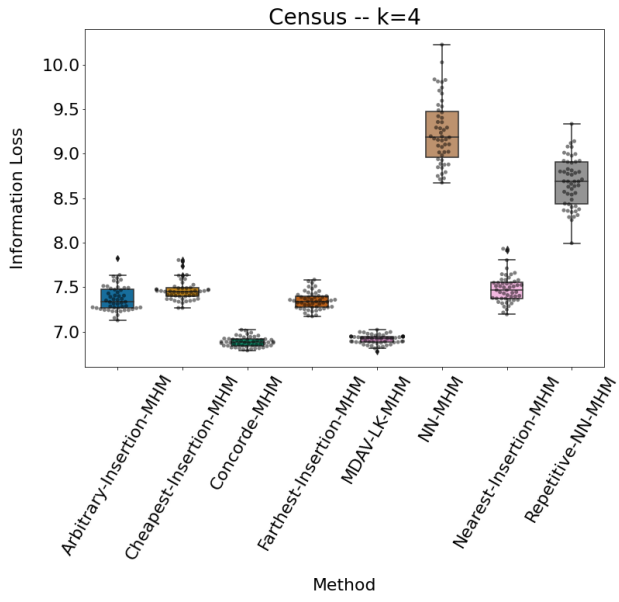


Figure 4.3: Information Loss variability for $k = 4$ over the Census dataset.

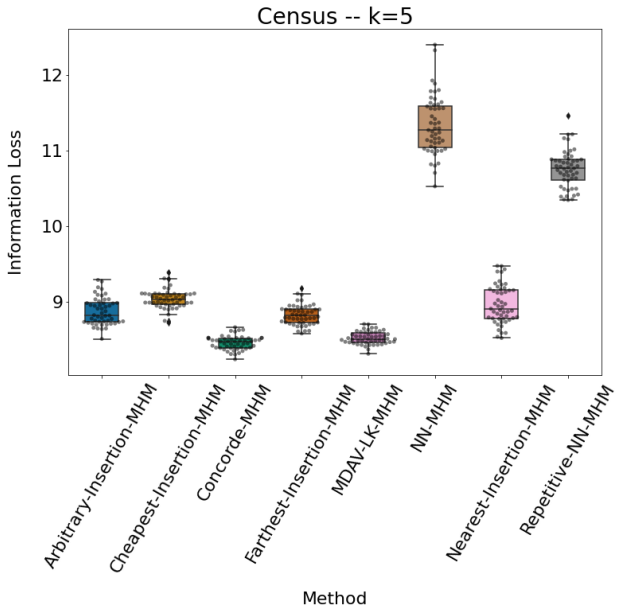


Figure 4.4: Information Loss variability for $k = 5$ over the Census dataset.

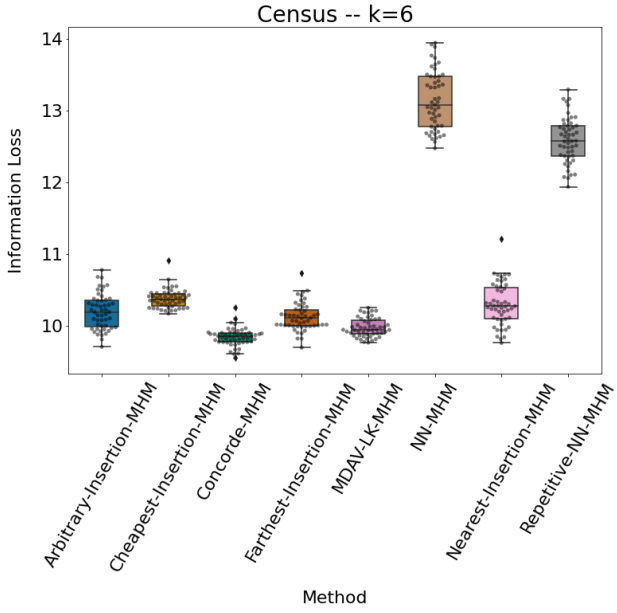


Figure 4.5: Information Loss variability for $k = 6$ over the Census dataset.

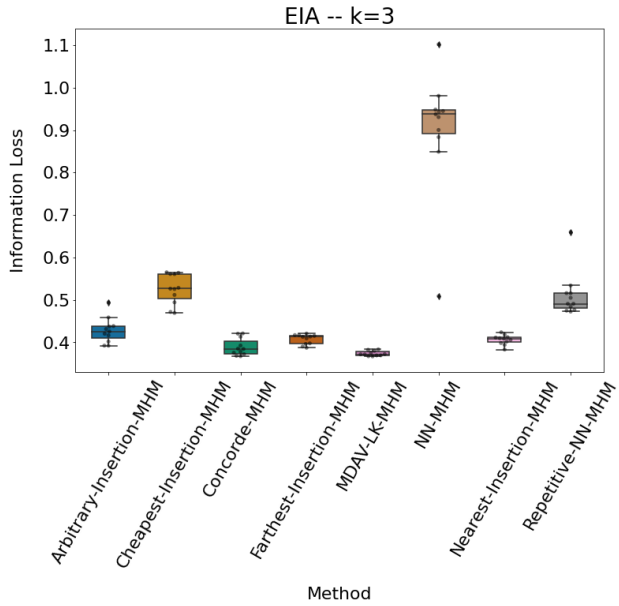


Figure 4.6: Information Loss variability for $k = 3$ over the EIA dataset.

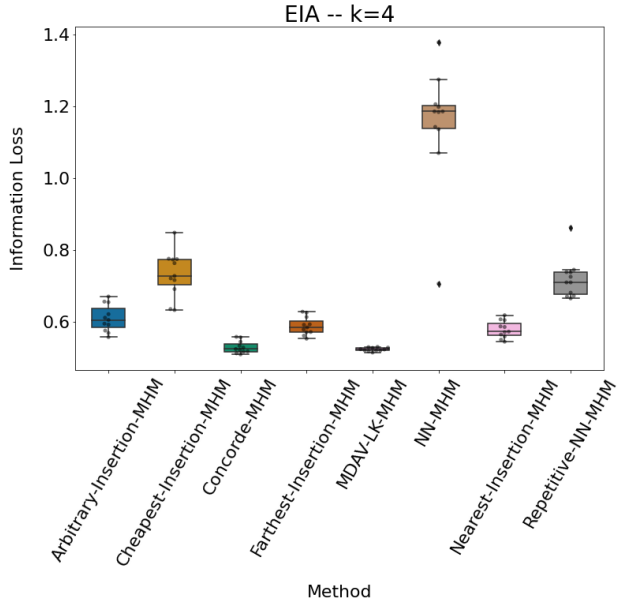


Figure 4.7: Information Loss variability for $k = 4$ over the EIA dataset.

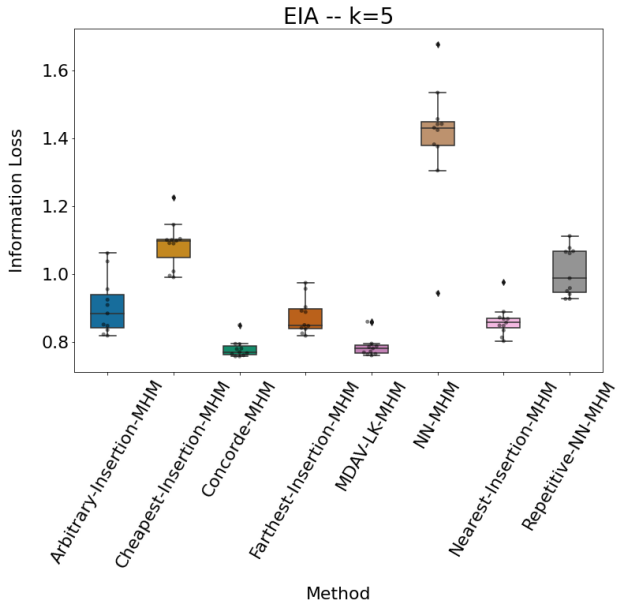


Figure 4.8: Information Loss variability for $k = 5$ over the EIA dataset.

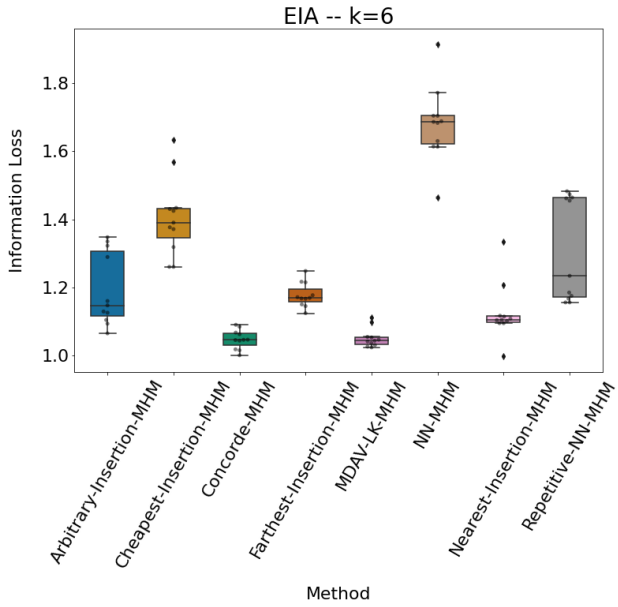


Figure 4.9: Information Loss variability for $k = 6$ over the EIA dataset.

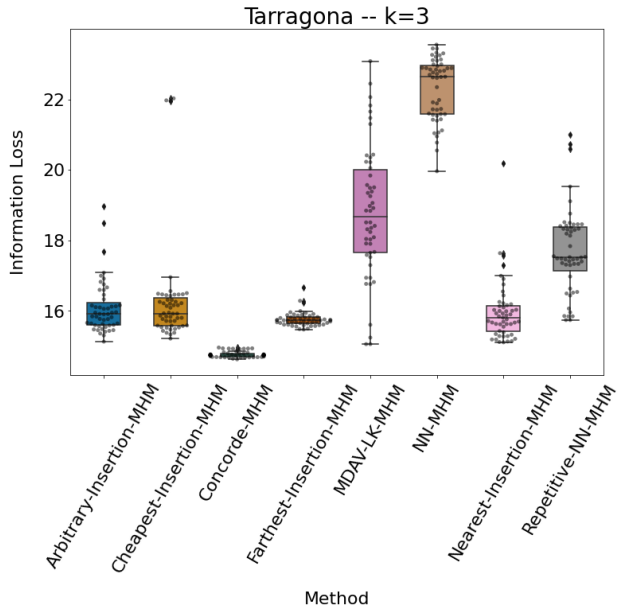


Figure 4.10: Information Loss variability for $k = 3$ over Tarragona dataset.

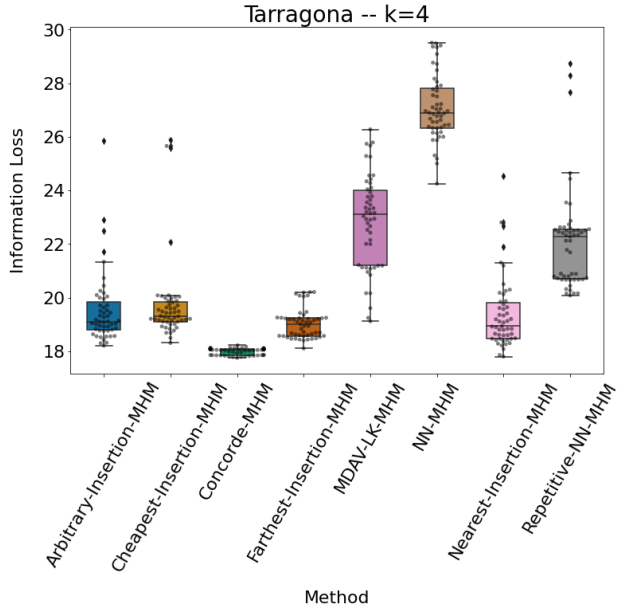


Figure 4.11: Information Loss variability for $k = 4$ over Tarragona dataset.

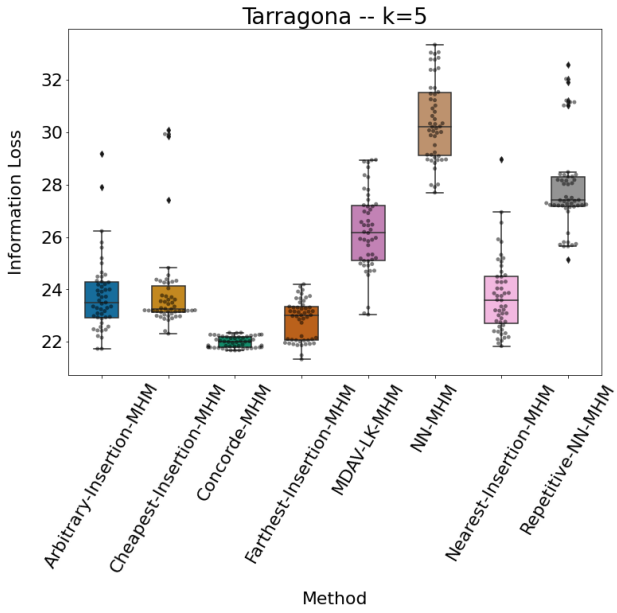


Figure 4.12: Information Loss variability for $k = 5$ over Tarragona dataset.

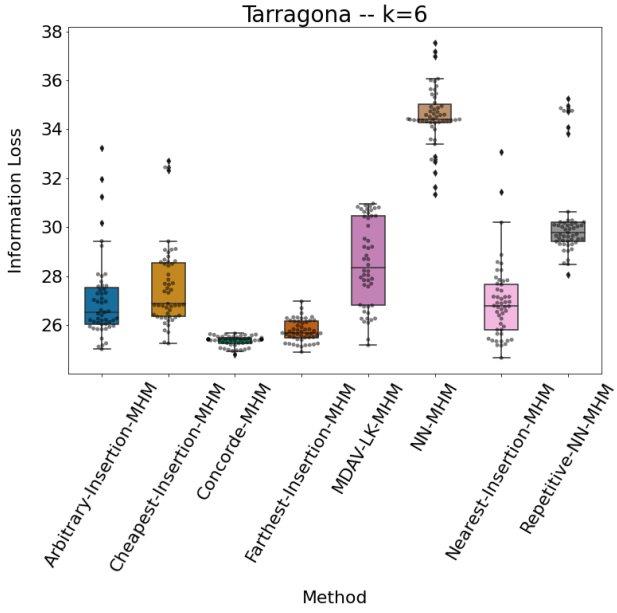


Figure 4.13: Information Loss variability for $k = 6$ over Tarragona dataset.

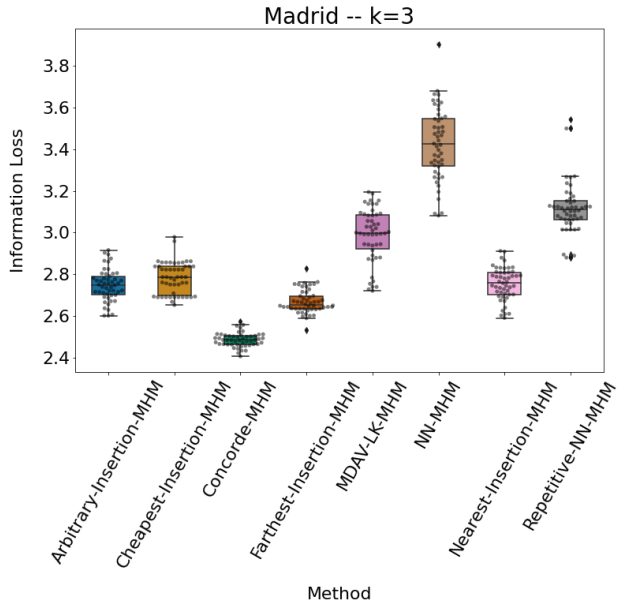


Figure 4.14: Information Loss variability for $k = 3$ over Madrid dataset.

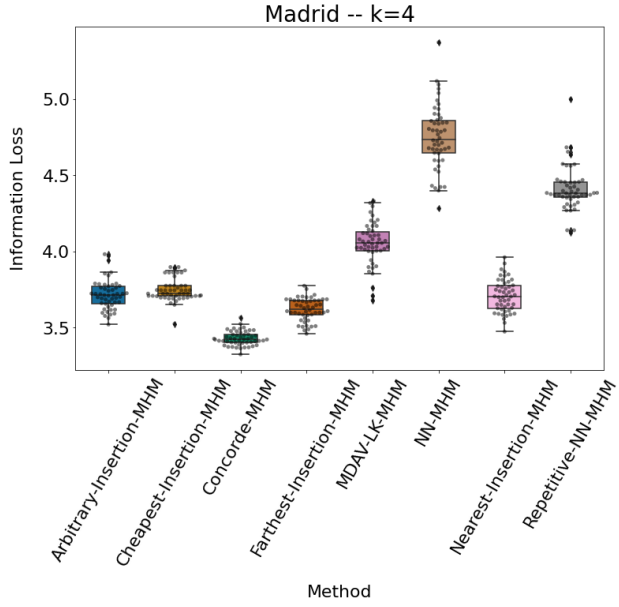


Figure 4.15: Information Loss variability for $k = 4$ over Madrid dataset.

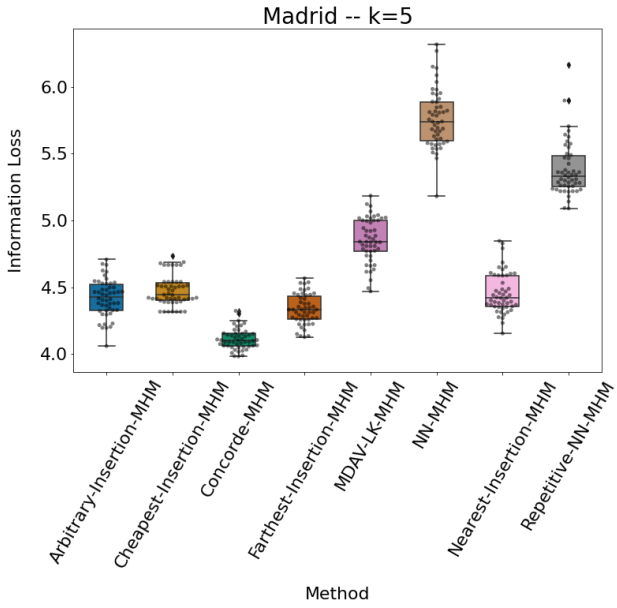


Figure 4.16: Information Loss variability for $k = 5$ over Madrid dataset.

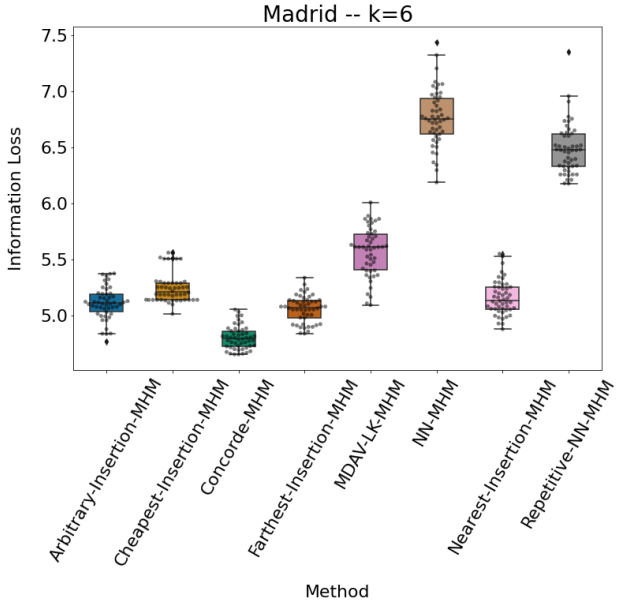


Figure 4.17: Information Loss variability for $k = 6$ over Madrid dataset.

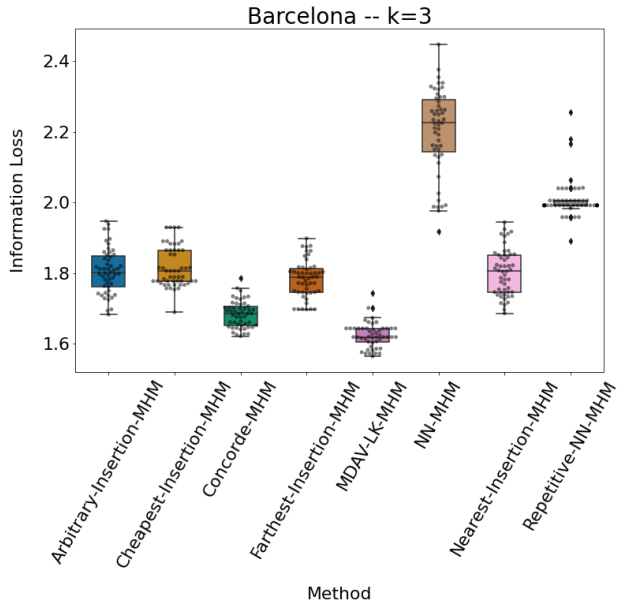


Figure 4.18: Information Loss variability for $k = 3$ over Barcelona dataset.

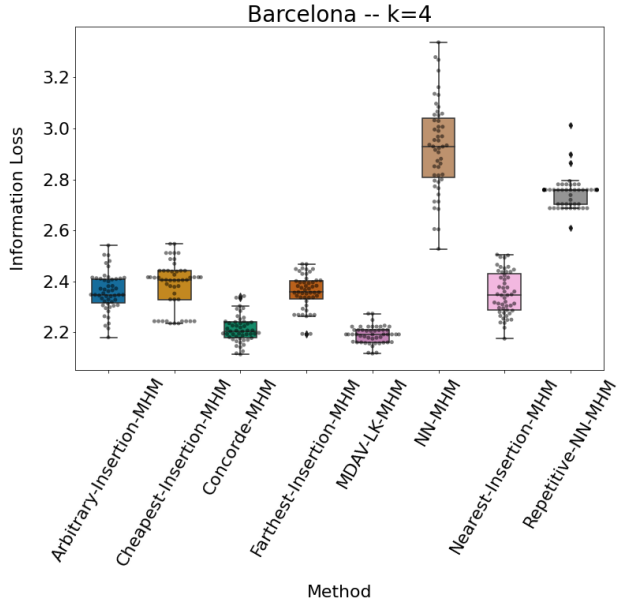


Figure 4.19: Information Loss variability for $k = 4$ over Barcelona dataset.

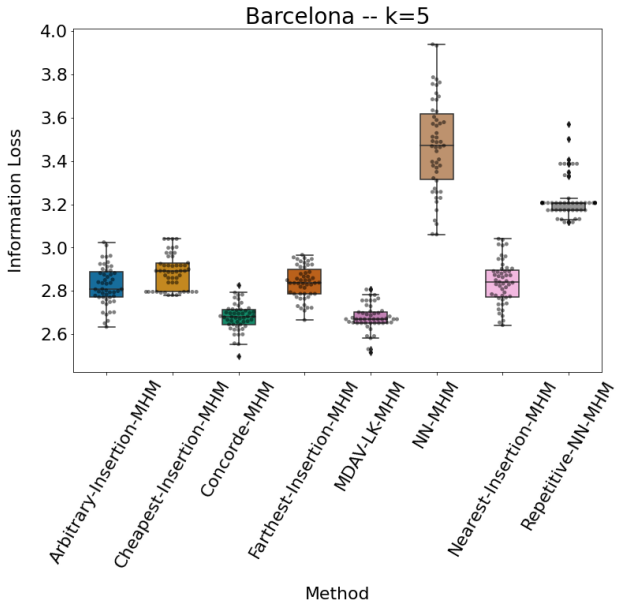


Figure 4.20: Information Loss variability for $k = 5$ over Barcelona dataset.

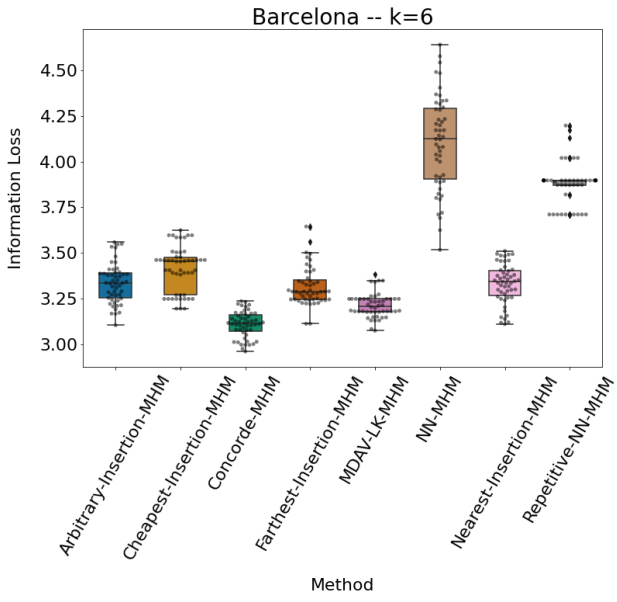


Figure 4.21: Information Loss variability for $k = 6$ over Barcelona dataset.

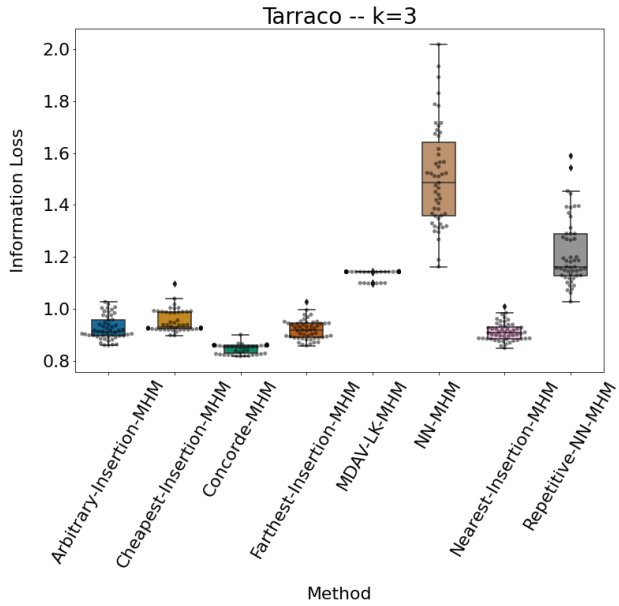


Figure 4.22: Information Loss variability for $k = 3$ over Tarraco dataset.

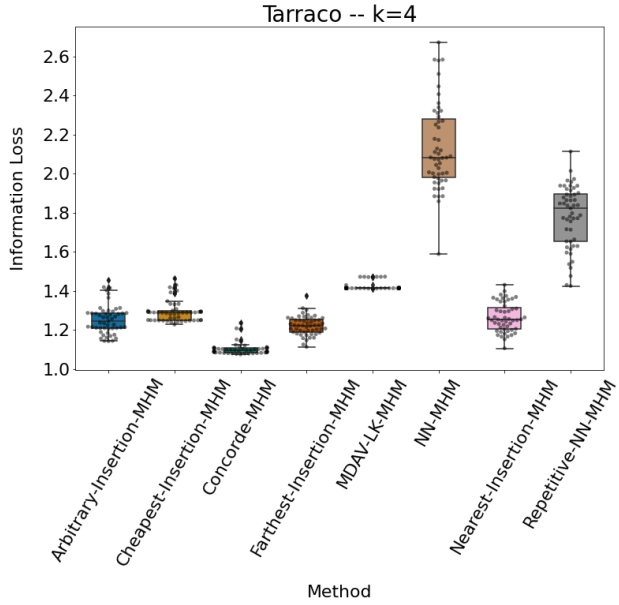


Figure 4.23: Information Loss variability for $k = 4$ over Tarraco dataset.

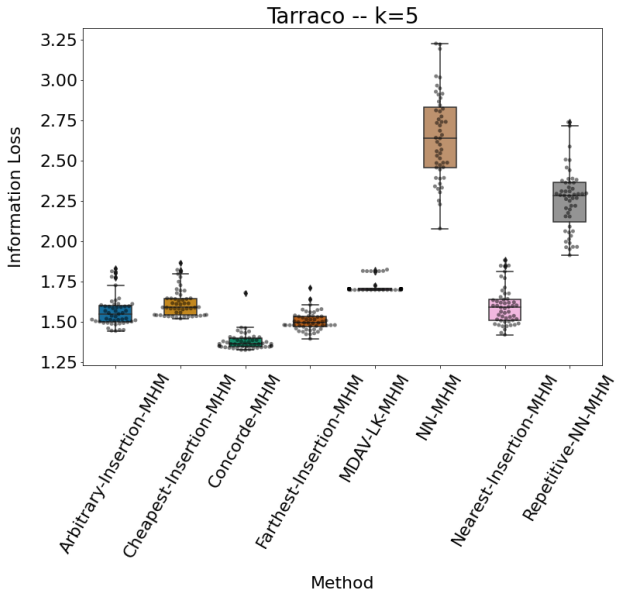


Figure 4.24: Information Loss variability for $k = 5$ over Tarraco dataset.

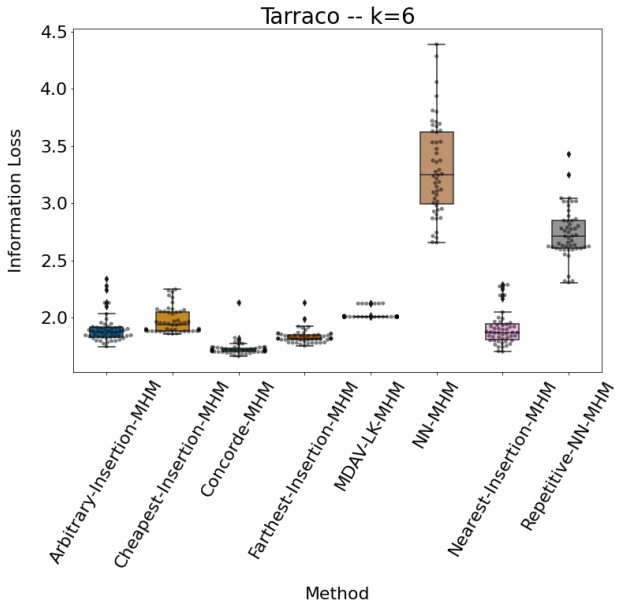


Figure 4.25: Information Loss variability for $k = 6$ over Tarraco dataset.

4.4 Discussion

Over the previous sections, we have presented our microaggregation method, $(HM)^2$-Micro, its rationale, and its performance against other classic and state-of-the-art methods on a variety of datasets. In the previous section, we have reported the main results and we will discuss them next by progressively answering the research questions that we posed in the Introduction of the Chapter.

> ***Q1:*** *How to create a suitable ordering for a univariate microaggregation algorithm, when the records are in* $\mathbb{R}^p$.

A main takeaway of this Chapter is that by using a combination of TSP tour construction heuristics (*e.g.* Concorde) and an optimal univariate microaggregation algorithm, we are properly ordering multivariate datasets in a univariate fashion that leads to excellent multivariate microaggregation solutions. Other approaches to order $\mathbb{R}^p$ points might consider projecting them over the principal component. However, the information loss associated with this approach makes it unsuitable. Also, other more promising approaches, like the one used in MDAV-LK-MHM, first create a k-partition and set an order based on maximum distance criteria. Although this approach might work well in some cases, we have clearly seen that Hamiltonian paths created by TSP-Heuristics like Concorde, outperform this approach. Hence, based on the experiments of Section 4.3 we can conclude that TSP-heuristics like Concorde provide an order for elements in $\mathbb{R}^p$ that is suitable for an optimal univariate microaggregation algorithm to output a consistent multivariate microaggregation solution with low Information Loss (*i.e.* high data utility). Moreover, from all analysed heuristics, it is clear that the best performer is Concorde, followed by insertion heuristics.

> ***Q2:*** *Are the length of the Hamiltonian path and the information loss of the microaggregation related?, or Do shorter Hamiltonian paths lead to microaggregation solutions with lower information loss?*

When we started this research, our intuition was that good heuristic solutions of the TSP (*i.e.* those with shorter path lengths) would provide a Hamiltonian path, that could be used as an ordered vector for the HM optimal univariate microaggregation algorithm, resulting in a good multivariate microaggregation solution. From this intuition, we assumed that shorter Hamiltonian paths would lead to lower Information Loss in microaggregated datasets.

In order to validate (or disproof) this intuition we have analysed the Pearson correlation between the Hamiltonian path length obtained by all studied heuristics (*i.e.* Nearest Neighbour, Repetitive Nearest Neighbour, Nearest Insertion, Farther Insertion, Cheapest Insertion, Arbitrary Insertion, and Concorde) and the SSE of the resulting microaggregation. We have done so for all studied datasets and k values. The results are summarised in Table 4.8, and all plots along with a trend line are available in Appendix B.

Dataset	$k = 3$	$k = 4$	$k = 5$	$k = 6$
Census	0.48	0.39	0.32	0.28
EIA	0.62	0.67	0.74	0.76
Tarragona	0.70	0.72	0.82	0.71
Barcelona	0.83	0.81	0.81	0.80
Madrid	0.84	0.81	0.80	0.78
Tarraco	0.80	0.82	0.82	0.80

Table 4.8: Summary of the Pearson correlation between Path Length and SSE

From the correlation analysis, it can be concluded that there is a positive correlation between the Hamiltonian path length and the SSE. This is, the shorter the path length the lower the SSE. This statement holds for all k and for all datasets (although Census exhibits a lower correlation). Hence, although this result is not a causality proof, it can be safely said that good solutions of the TSP problem lead to good solutions of the multivariate microaggregation problem. In fact, the best heuristic (*i.e.* Concorde) always results in the lowest (best) SSE.

Interested readers can find all plots in next section 4.4. However, for the sake of clarity, let us illustrate this result by discussing the case of the Madrid dataset with $k = 6$, depicted in Figure 4.26. In the figure, the positive correlation is apparent. Also, it is clear that heuristics tend to form clusters. In a nutshell, the best heuristic is Concorde, followed by the insertion family of methods (*i.e.* Nearest Insertion, Furthest Insertion, Cheapest Insertion and Arbitrary Insertion), followed by Repetitive Nearest Neighbour and Nearest Neighbour.

Although Figure 4.26 clearly illustrates the positive correlation between the path length and the SSE, it also shows that heuristics tend to cluster and might indicate that not only the path but the heuristic (*per se*) plays a role in the reduction of the SSE. This indication leads us to our next research question.

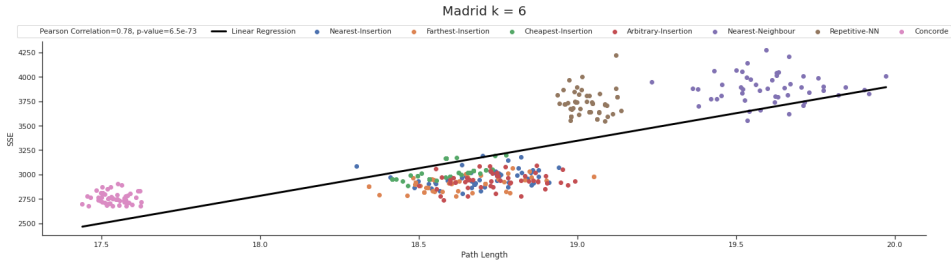


Figure 4.26: Relation between SSE and Path Length for Madrid and $k = 6$.

> ***Q3:*** *Is the length of the Hamiltonian path the only factor affecting information loss or does the particular construction of the path (regardless of the length) affect the information loss?*

In the previous question, we have found clear positive correlation between the path length and the SSE. However, we have also observed apparent clusters suggesting that the very heuristics could be responsible for the minimisation of the SSE. In other words, although the path length and SSE are positively correlated when all methods are analysed together, would this correlation hold when heuristics are analysed one at a time? In order to answer this question we have analysed the results of each heuristic individually and we have observed that, there is still positive correlation between path length and SSE, but it is very weak or almost non-existent (*i.e.*, very close to 0), as Figure 4.27 illustrates.

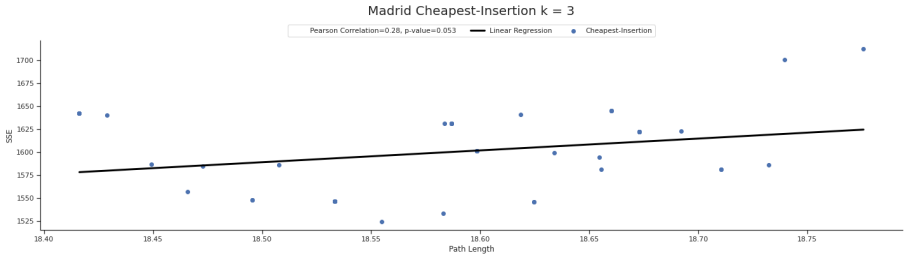


Figure 4.27: Correlation between path length and SSE for each individual method (from top to bottom: Cheapest Insertion, Concorde, and Nearest Neighbour) for $k = 3$ over the Madrid dataset.

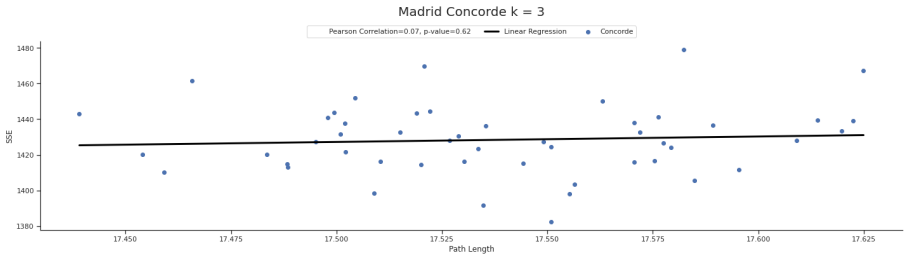


Figure 4.28: Correlation between path length and SSE for each individual method (from top to bottom: Cheapest Insertion, Concorde, and Nearest Neighbour) for $k = 3$ over the Madrid dataset.

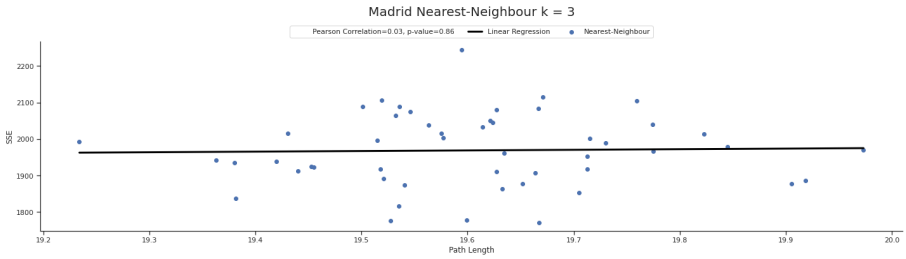


Figure 4.29: Correlation between path length and SSE for each individual method (from top to bottom: Cheapest Insertion, Concorde, and Nearest Neighbour) for $k = 3$ over the Madrid dataset.

The results shown in Figure 4.27 are only illustrative, and a deeper analysis that is out of the scope of this Chapter would be necessary. However, our initial results indicate that although there is positive correlation between path length and SSE globally, this correlation weakens significantly when analysed on each heuristic individually. This result suggests that it is not only the length of the path but the way in which this path is constructed what affects the SSE. This would explain why similar methods (*e.g.* those based on insertion) behave similarly in terms of SSE although their paths' length varies.

> ***Q4:*** *Does $(HM)^2$-Micro provide better solutions (in terms of information loss) than the best performing microaggregation methods in the literature?*

This question has been already answered in Section 4.3.2. However, for the sake of completeness we summarise it here: The results obtained after executing more than 12,000 tests suggest that our solution $(HM)^2$-Micro

obtains better results than classic microaggregation methods such as MDAV and V-MDAV. Moreover, when $(HM)^2$-Micro uses the Concorde heuristic to determine the Hamiltonian path, it outperforms the best state-of-the-art methods consistently. In our experiments, $(HM)^2$-Micro with Concorde was the best performer 79% of the times and was the second best in the remaining 21%.

> ***Q5:*** *Do TSP-based microaggregation methods perform better than current solutions on trajectories datasets?*

$(HM)^2$-Micro with Concorde is the best overall performer. Moreover, if we focus on those datasets with trajectory data (*i.e.* Barcelona, Madrid and Tarraco) the results are even better. It is the best performer in 83% of the tests and the second best in the remaining 17%. This good behaviour of the method could result from the very foundations of the TSP, however, there is still plenty of research to do in this line to reach more solid conclusions. Location privacy is a very complex topic that encompasses many nuances beyond k-anonymity models (such as the one followed in this article). However, this result is an invigorating first step towards the analysis of novel microaggregation methods applied to trajectory analysis and protection.

Correlation Analysis between "Path Length" and "SSE"

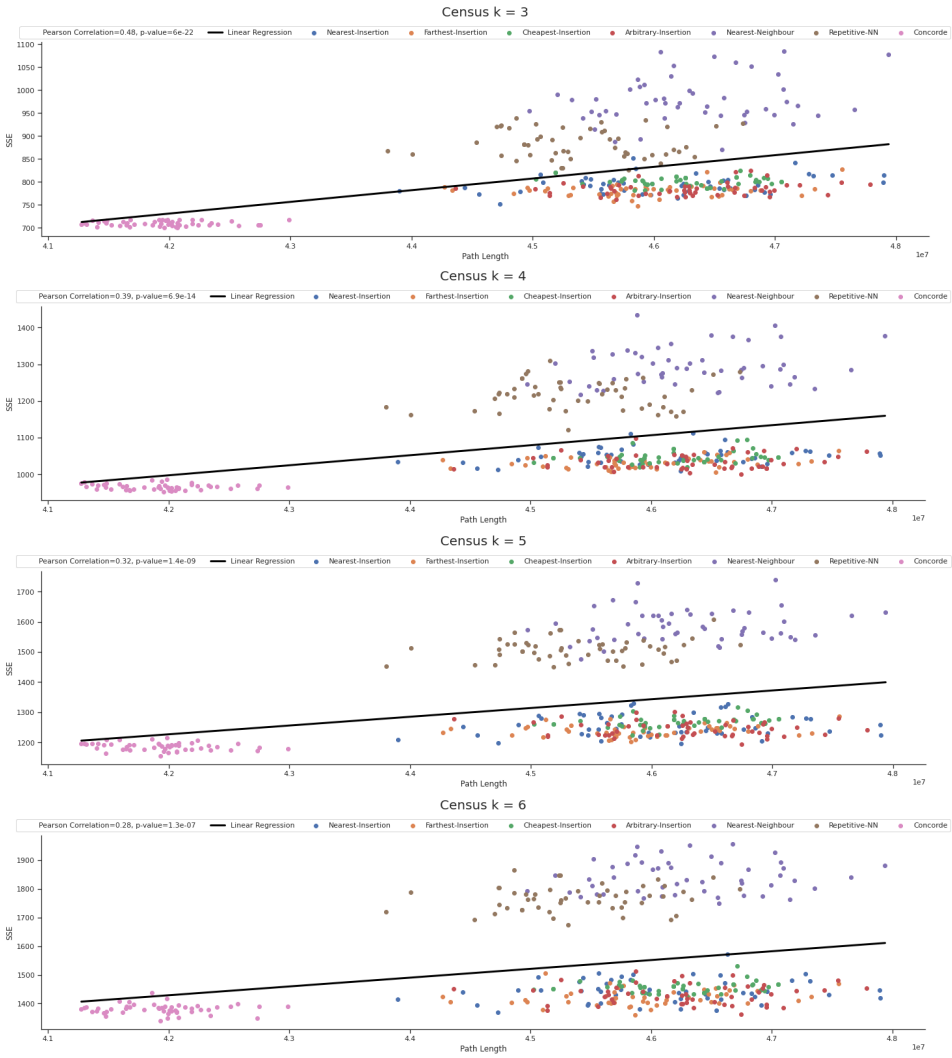


Figure 4.30: Relation between SSE and Path Length for Census and $k \in \{3, 4, 5, 6\}$.

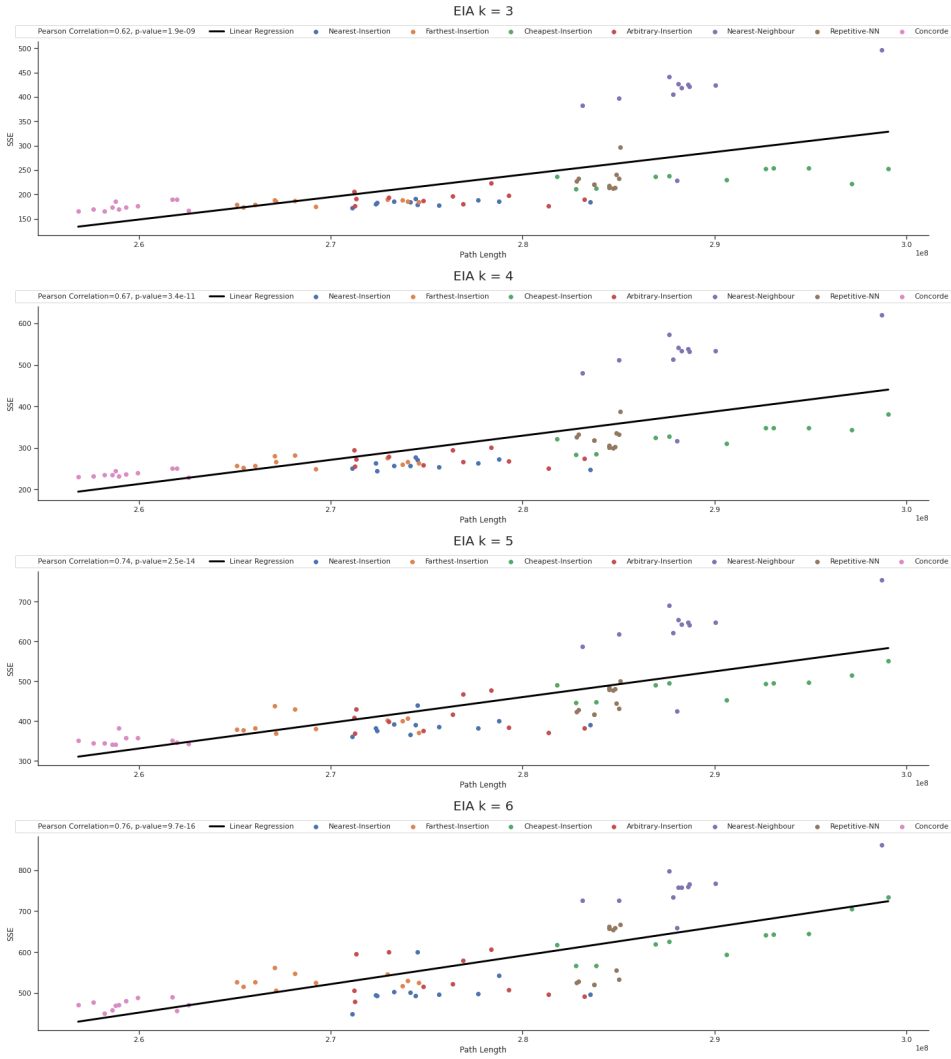


Figure 4.31: Relation between SSE and Path Length for EIA and $k \in \{3, 4, 5, 6\}$.

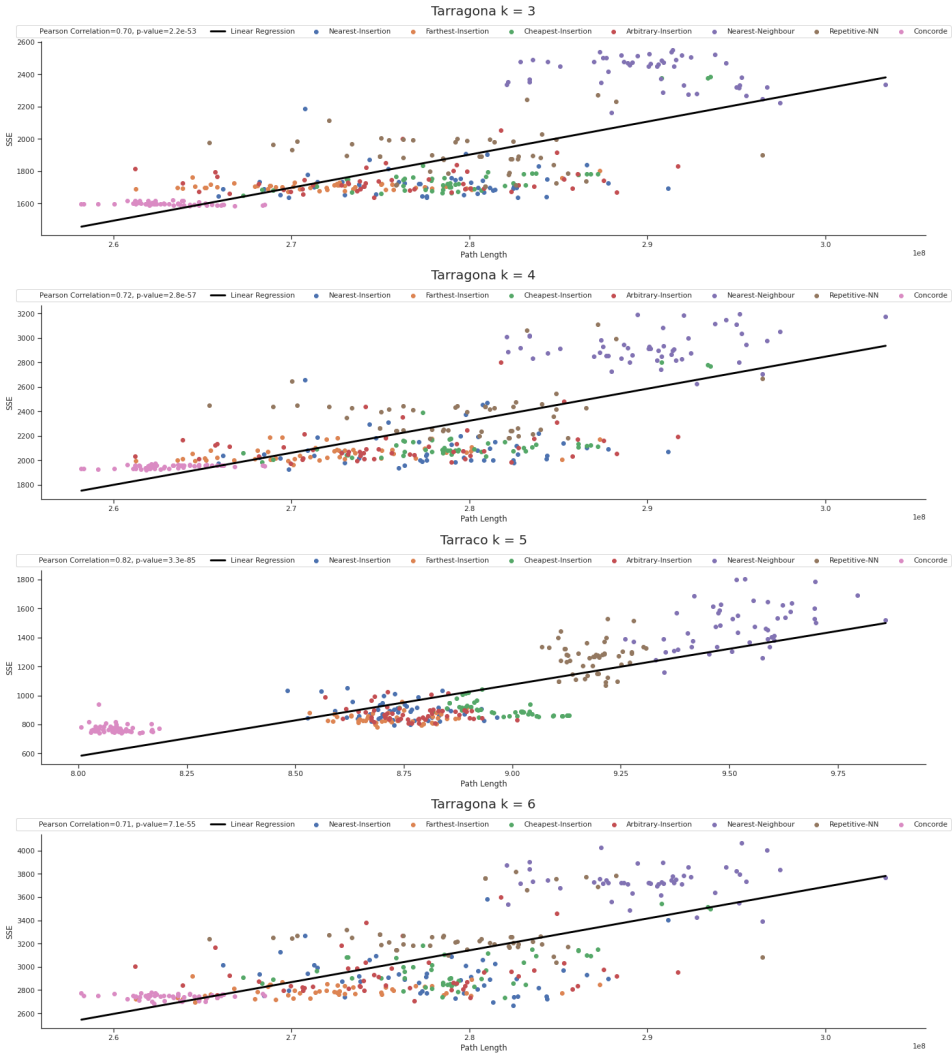


Figure 4.32: Relation between SSE and Path Length for Tarragona and $k \in \{3, 4, 5, 6\}$.

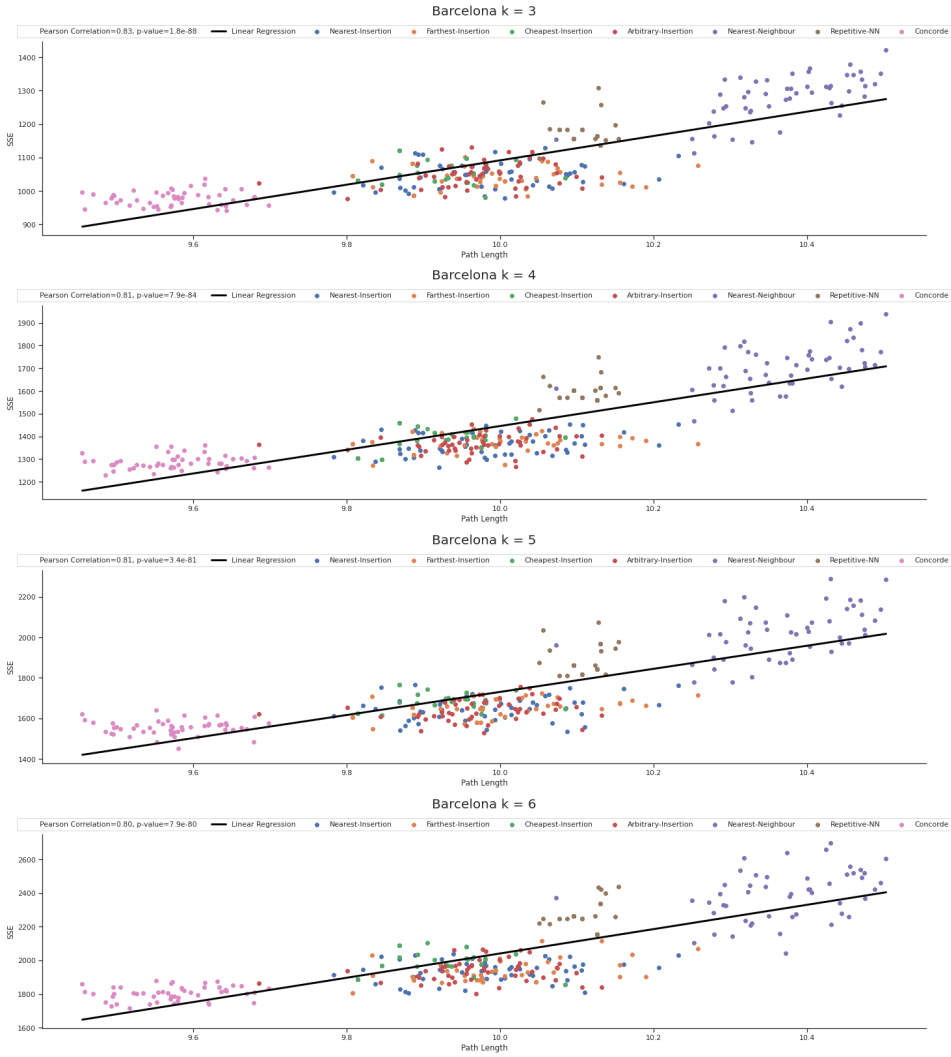


Figure 4.33: Relation between SSE and Path Length for Barcelona and $k \in \{3, 4, 5, 6\}$.

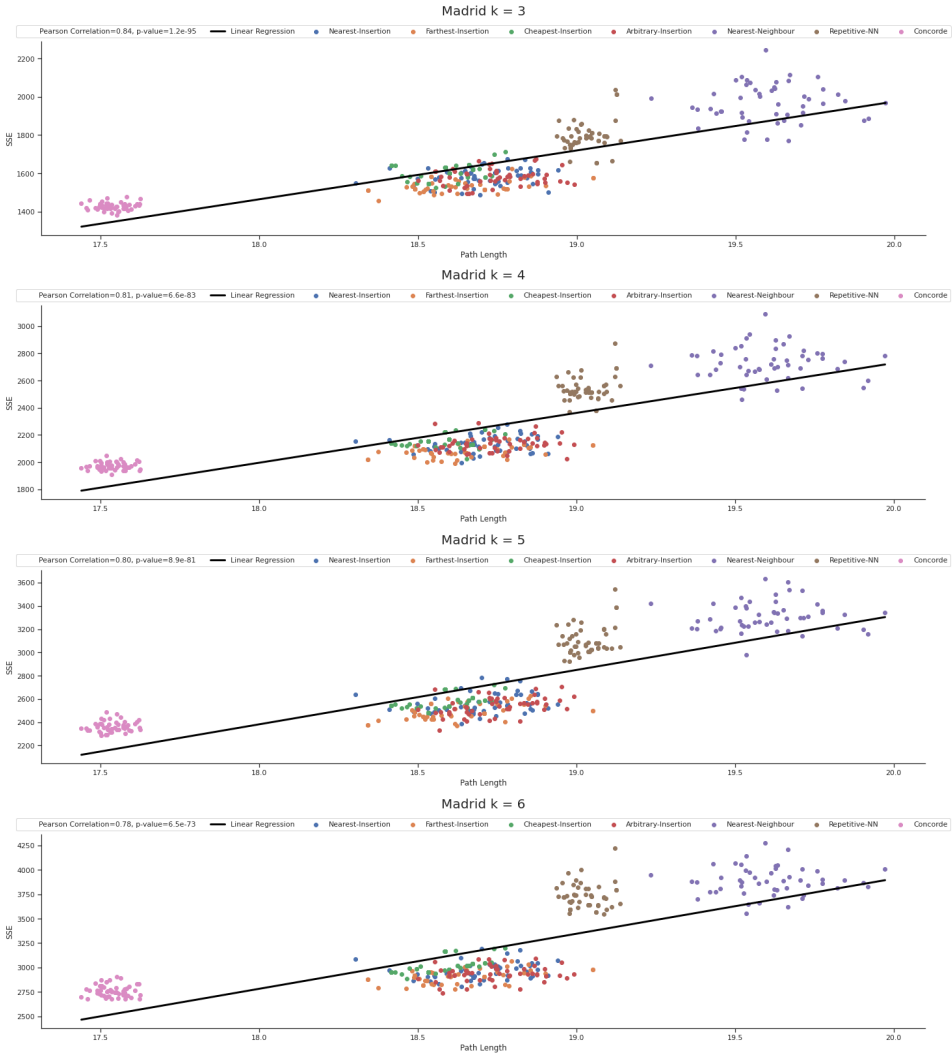


Figure 4.34: Relation between SSE and Path Length for Madrid and $k \in \{3, 4, 5, 6\}$.

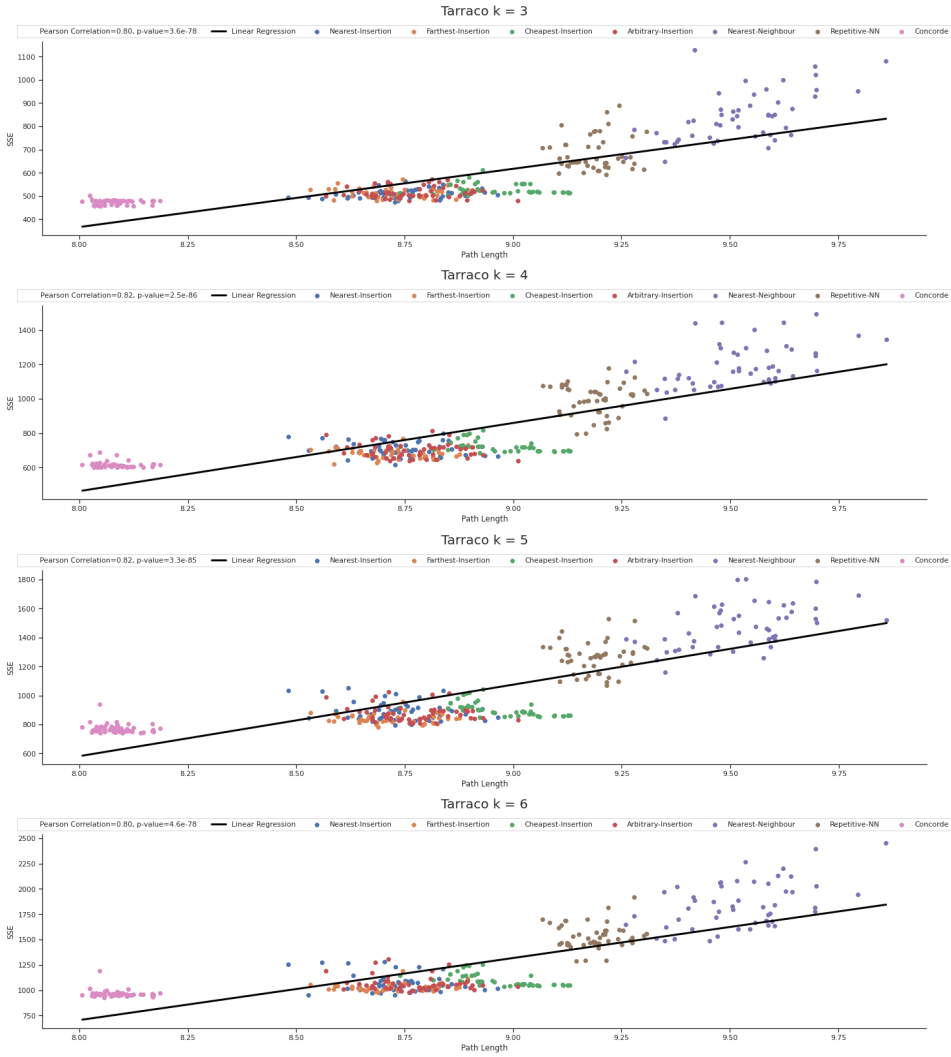


Figure 4.35: Relation between SSE and Path Length for Tarraco and $k \in \{3, 4, 5, 6\}$.

4.5 Conclusions

Although finding the optimal microaggregation is NP-Hard and a polynomial-time microaggregation algorithm has not been found, steady improvements over microaggregation heuristics have been made. Hence, after such a long research and polishing process, finding new solutions that improve the best methods is increasingly difficult. In this Chapter, we have presented $(HM)^2$-Micro, a meta-heuristic that leverages the advances in

TSP solvers and combines them with the optimal univariate microaggregation to create a flexible and robust multivariate microaggregation solution.

We have studied our method and thoroughly compared it to classic and state-of-the-art microaggregation algorithms over a variety of classic benchmarks and trajectories datasets. Overall, we have executed more than 12.000 tests, and we have shown that our solution embodying the Concorde heuristic outperforms the others. Hence, we have shown that our TSP-inspired method could be used to guarantee k-anonymity of trajectories datasets whilst reducing the Information Loss and hence increasing data utility. Furthermore, our proposal is very stable, this is, it does not change significantly its performance regardless of the random behaviour associated with initial nodes selection.

In addition to proposing $(HM)^2$-Micro, we have found clear correlations between the length of Hamiltonian Paths and the SSE introduced by microaggregation processes, and we have shown the importance of the Hamiltonian Cycle construction algorithms over the overall performance of microaggregation.

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO STATISTICAL DISCLOSURE CONTROL: ENHANCING MULTIVARIATE MICROAGGREGATION USING GRAPH THEORY
Armando Maya-López

Chapter 5

Contributions to Dataset Reduction Strategies

*NP-hard problems, like microaggregation, are decision problems that require a number of operations that grows exponentially with the size of the input, making it infeasible to solve for large instances. As a result, researchers often focus on developing efficient algorithms and heuristics to solve or approximate NP-hard problems. However, in many cases, these problems may still require significant computational resources and time to solve. Dataset reduction strategies are techniques used to reduce the size or complexity of a dataset while maintaining its integrity and usefulness for analysis. The choice of dataset reduction strategy will depend on the specific requirements of the analysis and the nature of the dataset. It is important to carefully consider each strategy's potential benefits and limitations before implementing them. In this Chapter, we present a method to compress a microdata dataset, which can be used to solve the Travelling Salesman Problem, allowing us to optimise its time cost. This strategy involves compressing the data to reduce its size while preserving the relevant information. To the best of our knowledge, this is the first time that an optimisation method for TSP-based microaggregation heuristics (*i.e. *optimising both the computational time and the quality of the groups) is presented in the literature.*

Contents

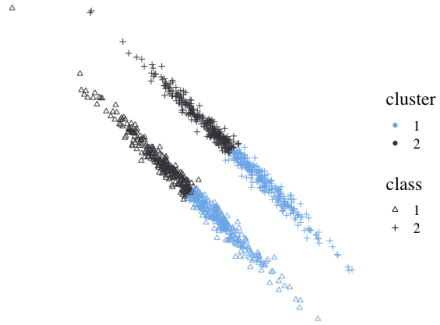
5.1 Dataset Reduction Strategies for Microaggregation

The computational cost of the heuristics grows in runtime with the size of the dataset on which we want to operate. Consequently, when dealing with large datasets or using computationally demanding methods, techniques like dataset splitting [33] or dimensionality reduction [18] must be contemplated.

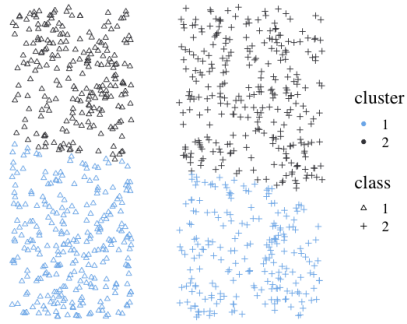
Notwithstanding, these approaches entail several shortcomings, such as the "noise" introduced by dimensionality reduction methods when dealing with high-dimensional data [5, 16], and the fact that splitting strategies (and hence the risk of grouping similar records into different subsets) may have a great impact on the usability of data and their statistical properties [4, 36].

A baseline strategy for dataset splitting is to use clustering algorithms to generate smaller subsets. Nevertheless, clustering algorithms such as K-means may create partitions that separate the records erroneously according to data distribution, thus introducing noise and dramatically hindering the quality of the groups, as seen in Figure 5.1.

Aiming at avoiding the split of natural clusters, in [36], authors applied MDAV in a two-step partitioning strategy. First, they used MDAV with a low value of k to ensure that close elements will not be separated; next, they generated partitions by using a higher value of k over the dataset created in the first step. While this strategy may be efficient for clustering-based algorithms, it is not efficient for TSP-based methods since they require all



(a) Non zero covariance dataset



(b) Uniform distribution dataset

Figure 5.1: Example of K-means inefficient partition strategies over two different datasets.

the dataset records to compute the optimal path by exploring all the possible connections.

5.2 Our Proposal

The TSP is a well-known NP-Hard problem in the literature, whose complexity depends on the amount of "cities" (in our case, the records in the dataset) considered to compute the path traversing all records (*i.e.* the Hamiltonian path). This is a common step of all TSP-based microaggregation heuristics, as extensively reported in [22].

In the words of the researchers who designed an implementation for solving the TSP, "At the current time, our notion of very large refers to problems having over 10,000 cities" [1]. Therefore, a strategy should be considered to calculate this type of dataset with more than 10,000 records. In this ar-

ticle, however, we show that optimisation strategies can be used in both small and large datasets according to the application scenario, contrary to the assumption stated above. Moreover, in addition to focusing only on the computational time perspective, our optimisation strategy can preserve, and sometimes improve, the quality of the clusters in a microaggregation setting. In what follows, we describe our proposal to leverage the performance of TSP-based microaggregation, and we analyse the benefits of the group protecting strategies in the context of TSP methods. Intuitively, methods that may reduce the computational time of TSP-based heuristics may incur a higher information loss. Nevertheless, in this Chapter, we propose a method that improves the performance of TSP-based heuristics and can be used in both small and large datasets effectively. Moreover, instead of focusing only on the computational time perspective, our method can preserve and sometimes reduce the information loss resulting from the microaggregation. Extensive experiments with the different benchmarks dataset described in section 2.3 show how our method is able to outperform the current state of the art, considering the trade-off between information loss and computational time.

The TSP [34] (*i.e.* finding the shortest hamiltonian path over a graph) has been extensively used in different types of applications [29, 46], yet it has been much less explored in the SDC field due to its computational cost [14, 22]. In this Chapter, we present a method to improve the efficiency of microaggregation (*i.e.* a family of SDC techniques) that utilises the TSP to create a path over the multivariate records. To the best of our knowledge, this is the first time that an optimisation method for TSP-based microaggregation heuristics (*i.e.* optimising both the computational time and the quality of the groups) is presented in the literature. The remainder of the Chapter is organised as follows: First, we describes our proposal, which is later thoroughly tested and compared with well-known classical and state-of-the-art microaggregation methods in Section 'Experiments'. Section 'Discussion' analyses the benefits and limitations of our approach. The Chapter concludes in Section 'Conclusions' with some final remarks.

5.2.1 A Compression Strategy for Efficient, TSP-based Microaggregation

Algorithm 3 describes our compressed, TSP-based microaggregation process, to microaggregate a dataset D with r records and p columns, with a privacy parameter k (*i.e.* to create a k-anonymous version of dataset D). Essentially, it is divided into four phases:

- *Phase 1. Compression.* A compressed version D_{comp} of the dataset is generated.
- *Phase 2. TSP Path Finding.* The TSP is performed to find a Hamiltonian path HP_{comp} over the compressed dataset, regarded as a graph.
- *Phase 3. Decompression.* It consists of using HP_{comp} to create a new Hamiltonian path HP_{dec} over the original dataset.
- *Phase 4. Microaggregation.* The Multivariate Hansen and Mukherjee technique is applied using the H_{dec} path to create the k-partition.

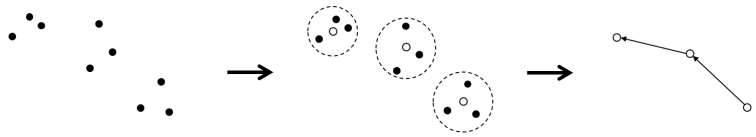
Next, we focus on some details of the algorithm. As an initial step and, to avoid bias towards higher magnitude variables, D is standardised (Algorithm 3: line 2). To this end, each value of the dataset is subtracted the average of its column and divided by the standard deviation of this column.

The first step of Phase 1 (Algorithm 3: line 3) consists in obtaining C_c, a c-partition of D_{std} using a microaggregation method m with a cardinality constraint parameter c (*i.e.* the compression ratio). Thus, $C_c = \{c_1, c_2, \dots c_{r/c}\}$ describes the clusters in the c-partition (*e.g.* $c_3 = \{7, 32, 94\}$ indicates that cluster number three is composed by rows 7, 32 and 94 of D).

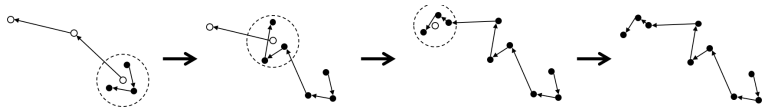
In the second step of Phase 1 (Algorithm 3: line 4), the compressed dataset D_c is built, using the set of r/c centroids C_c generated in the previous step. Following the previous example, the third row in D_c is the centroid of cluster c_3, *i.e.* the average of records 7, 32, and 94.

Phase 2 consists of finding a Hamiltonian path over D_c. First (Algorithm 3: line 5), we model D_c as a complete graph $G(N, E)$, where we assume that each row of D_c is represented by a node $n_i \in N$ and each edge $e_{ij} \in E$ represents the Euclidean distance between n_i and n_j. Thus, we have a set of nodes $N = \{n_1, n_2, \dots n_{r/c}\}$ each representing rows of the compressed microdata set in a multivariate space $\mathbb{R}^p$. Next (Algorithm 3: line 6), we apply a TSP path construction heuristic over G to create a Hamiltonian path HP_C, *i.e.* a permutation ($\Pi^N = \{\pi_1^N, \pi_2^N, \dots \pi_{r/c}^N\}$) of the nodes in N, which, *de facto* determines a specific order.

Phase 3, aims at creating a new Hamiltonian path HP_D, but in this case considering all the records/nodes of D_{std}, whilst preserving the specific order determined in HP_C. This process consists of iterating all the nodes of HP_C and, for each node $HP_c(i)$, insert as nodes in HP_D all the records its corresponding cluster in the c-partition C_c (Algorithm 3: line 9). The order of insertion of each cluster's records is determined by the distance to dataset's centroid according to D_{std} (Algorithm 3: line 8).



(a) Original records (left), clusters and their corresponding centroids (center), and resulting Hamiltonian Path (right).



(b) From left to right, decompression of each centroid by replacing it with the records of the original cluster. Inner nodes are ordered according to their distance to the dataset's corresponding centroid.

Figure 5.2: Illustration of the compression (a) and the decompression (b) processes.

After this decompression process, the result is a new HP_D that includes the permutation of all elements existing in the original dataset.

Finally, in Phase 4 the original dataset is microaggregated. HP_D is used as input to the MHM method, together with the privacy parameter k, to create the k-partition (Algorithm 3: line 11). It returns the optimal univariate k-partition of D_{std}, which is used to build the microaggregated dataset, *i.e.* each record in the group is substituted by the group's centroid (Algorithm 3: line 12).

For the sake of clarity, we provide an overview of the compression and the decompression steps in Figure 5.2

Algorithm 3 Function Compress-TSP-MHM that microaggregates the dataset D.

function COMPRESS-TSP-MHM(microdata set **D**, microaggregation method **m**, TSP method **t**, privacy parameter **k**, compression ratio **c**)
    $D_{std} \leftarrow standardizeDataset(D)$

    // Phase 1: Compression
    $C_c \leftarrow$ microaggregation(D_{std}, m, c)
    $D_{comp} \leftarrow$ getCentroids(C_c, D_{std})

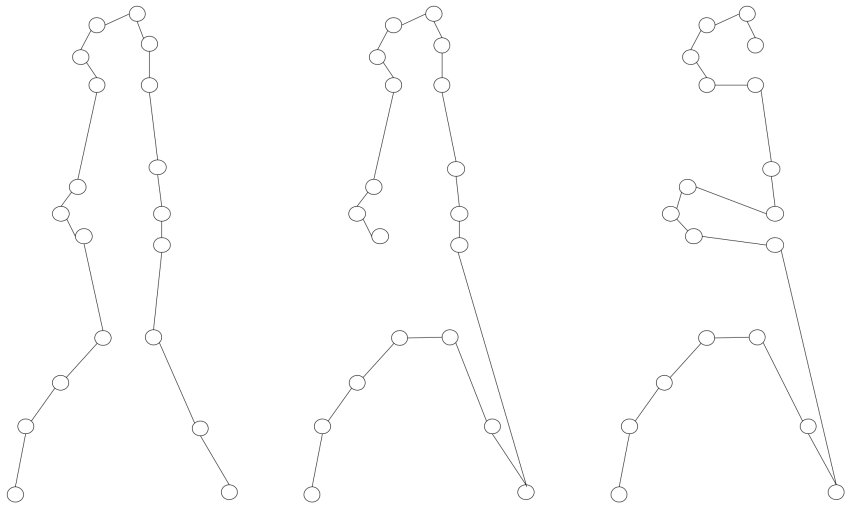
    // Phase 2: TSP Path Finding
    $G \leftarrow$ createGraph(D_{comp})
    $HP_{comp} \leftarrow$ computeTSP(G)

    // Phase 3: Decompression
    for $i = 1$ to $length(HP_{comp})$ **do**
        $c \leftarrow sortRecords(C_c(HP_{comp}(i)), D_{std})$
        $HP_{dec} \leftarrow addRecords(c)$
    end for

    // Phase 4: Microaggregation
    $C_k \leftarrow$MHM(HP_{dec}, D_{std}, k)
    $M \leftarrow$ buildMicroaggregatedDataSet(C_k, D)
     return M
end function

5.2.2 Path Length and Microaggregation

One of the primary keys supporting the feasibility of our method is the fact that a shorter path length does not guarantee the creation of better grouping strategies than using longer paths, as observed in the literature [7]. The latter becomes more evident when applying an optimisation method such as MHM, as noted later in Section 'Microaggregation Results'. In fact, given a strategy that protects the groups according to some parameters, the resulting path may generate a more appropriate grouping strategy for microaggregation. An example of the latter is illustrated in Figure 5.3. As it can be observed, paths b and c are longer than path a, mainly due to the long diagonal connection emerging from their rightmost node. Despite that, we can observe that path b can protect groups, especially for a cardinality value $k > 4$, in a more efficient way than path a since the latter would



(a) Path a (b) Path b (c) Path c

Figure 5.3: Example of different grouping protection strategies. Note that paths b and c may generate better microaggregation strategies despite describing longer paths than path a.

generate groups with more separated elements. A further group protection strategy can be seen in path c, in which groups of $k > 4$ will be generated more efficiently than with path a, thus improving the microaggregation outcomes. Note that different protection strategies may exhibit different outcomes according to each dataset and its inherent data distribution.

5.3 Experiments

In this section, we provide different experiments to showcase the efficacy of our approach by testing it on three datasets that serve as benchmarks. More concretely, we first analyse different aspects related to the computational time of our method in section 'Computational Time and Data Distribution'. Next, by using the benchmark datasets, we analyse the I_{loss} (expressed in percentage), as a measure of data utility (*cf.*, Section 2.3 for details). Note that given a privacy parameter k that guarantees that the microaggregated dataset is k-anonymous, the lower the I_{loss} the better the result and performance of the microaggregation method. Therefore, we compare our proposal to current state-of-the-art methods, and the results of all these tests are summarised in Section 'Microaggregation Results'. Finally, we analyse the trade-off between I_{loss} and computational time of our method in Section

'Trade-off Analysis of TSP-based Methods'.

We used three datasets as benchmarks for our experiments, defined in section 2.3, "Census", "EIA" and "Tarragona". These SDC microdata sets have been used for years as benchmarks in the literature [9, 41].

In the implementation, we selected MDAV due to its efficiency (*i.e.* the complexity of MDAV is quadratic with respect to the number of records in the dataset [9]). For solving the TSP we used the Concorde approach [1], which is currently one of the best approaches, as seen in Chapter 2.

For the sake of brevity, we refer to the application of TSP without the compression/decompression phases (hence, performing the TSP over the original, r-records dataset) as C-MHM. Accordingly, the application of our compression proposal, *i.e.* executing compression/decompression phases of Algorithm 3, is labelled as Cc-C-MHM, where c denotes the compression factor. For comparison purposes, we have used MDAV [9] and V-MDAV [38].

In the experiments related to running time, we have used an implementation in R with RStudio IDE version 1.3.1093 and the packages TSP version 1.1 and sdcMicro version 5.5.1, running on a computer with 4 x 2.2GHz Intel Core i7 CPU and 16GB of RAM.

5.3.1 Computational Time and Data Distribution

As defined in [24], the computational cost of Concorde is $O(Mb^d)$, where M is a bound on the time to explore subproblems and is tied to the number of nodes, b is a branching factor, and d is a search depth. To study the impact of a dataset's number of records and the distribution of its elements on the Concorde method, we created two sets of datasets in $\mathbb{R}^2$ with $[100, 500, 1000, 5000, 10000]$ records, namely normal set and uniform set, where:

- In the normal set, the *rnorm()* function from *R-Project* was used, which generates random elements with a normal distribution $N(0, 1)$.
- In the uniform set, the *runif()* function from *R-Project* was used, which generates random elements with a uniform distribution $U(0, 1)$.

Next, we applied the C-MHM method to each dataset and depicted the corresponding times in Figure 5.4. As observed, there is a relationship between the number of elements of a dataset and the computational time of the method, which follows an exponential growth path. Moreover, note there are no distinguishable differences between both data distributions (cf. Figure 5.4). Notably, in datasets with a low number of elements, the difference is almost negligible since most of the time corresponds to system overhead.

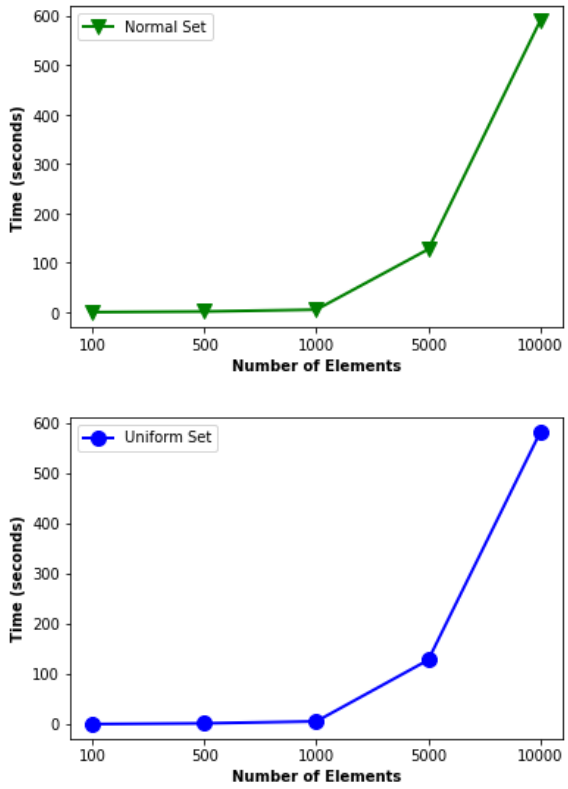


Figure 5.4: Time (in seconds) required by our method to create a TSP path according to different data distributions.

The next experiment analyses the efficiency due to compression/decompression compared to the original Concorde-MHM method. Table 5.1 shows the time required to compute the TSP path according to each strategy and benchmark dataset. As observed, the time required by the original Concorde approach is always higher than the one required when including our compression proposal. Moreover, the higher the value of c, the lower the time, which decreases exponentially as seen in Section 'Computational Time and Data Distribution'.

5.3.2 Microaggregation Results

Compression aims to reduce the computational time required by the TSP methods to compute a path while minimising the I_{loss}. In this regard, the outcomes of the experiments denote that applying the compression succeeds at fulfilling that objective with more or less efficacy according to their input

Table 5.1: Computational times (in seconds) of each combination of compression parameter and benchmark.

	Methods				
Dataset	**C-MHM**	**C2-C-MHM**	**C3-C-MHM**	**C4-C-MHM**	**C5-C-MHM**
Census	6.58	1.81	0.9	0.56	0.39
Tarragona	4.75	1.71	0.89	0.6	0.44
EIA	75.26	18.28	8.05	4.97	3.19

parameter and the characteristics of the dataset under evaluation. In all cases, compression increases the path length compared to the original Concorde approach. However, they achieve lower, and thus better I_{loss} outcomes in some cases, following the observations described in Section 'Path Length and Microaggregation'. In the case of Census (cf Table 5.2) we can observe that the C2-C-MHM strategy obtains similar values as C-MHM, especially for $k > 3$, and outperforms it for $k > 5$. Similarly, the rest of the compression ratios obtain better values as k increases. C5-C-MHM is the strategy that obtains the best outcomes for $k = 10$, despite the notable increase in path length.

The outcomes obtained in the case of Tarragona are depicted in Table 5.3. The behaviour of applying compression in Tarragona is not as efficient as in Census in terms of I_{loss}. For $k > 5$, we achieve values close to these obtained by the C-MHM method, outperforming them in the case of $k = 8$ with C2-C-MHM, and C3-C-MHM with $k = 10$.

Finally, Table 5.4 shows the outcomes obtained in the case of EIA. Again, the outcomes between C-MHM and the application of compression are closer the higher the value of k is, especially for $k > 6$. In the particular case of $k = 8$, C3-C-MHM outperforms the original C-MHM. We can also observe that in the case of C5-C-MHM, the I_{loss} obtained is remarkably worse, indicating that such compression is hindering the inherent group distribution of the dataset. More concretely, due to the particular data distribution of EIA's records, we can observe that some values of k enforce the creation of groups that break such natural disposition. The latter, however, can be remarkably overcome by using variable-sized heuristics.

5.3.3 Trade-off Analysis of TSP-based Methods

To better illustrate the efficacy of our proposal in the context of TSP-based heuristics, we created an additional experiment analysing the trade-off between the I_{loss} and the computational time, as we aimed to enhance both.

Table 5.2: Percentage of I_{loss} obtained on the Census dataset. Highlighted values denote the best outcome for each k.

	Census								
Method	**Path Length**	**k = 3**	**k = 4**	**k = 5**	**k = 6**	**k = 7**	**k = 8**	**k = 9**	**k = 10**
MDAV	N/A	5.6922	7.4947	9.0884	10.3847	11.6688	12.3916	13.3368	14.1559
V-MDAV	N/A	5.6619	7.4947	9.0070	10.2666	11.5999	12.2985	13.3368	14.0730
C-MHM	1173.23	5.0321	6.9691	8.4681	9.7646	11.0744	12.4255	13.7720	14.9954
C2-C-MHM	1308.05	5.2541	6.9869	8.5187	9.4894	10.7869	11.9158	12.8477	13.8309
C3-C-MHM	1531.64	5.6821	7.8676	8.9504	9.5073	11.1161	12.118	12.7336	13.7258
C4-C-MHM	1607.55	6.9922	7.4893	9.7413	10.8816	11.7034	12.096	13.4572	14.1274
C5-C-MHM	1689.13	7.8868	8.8086	9.0884	11.4911	12.2091	12.9724	13.4921	13.6916

Table 5.3: Percentage of I_{loss} obtained on the Tarragona dataset. Highlighted values denote the best outcome for each k.

	Tarragona								
Method	**Path Length**	**k = 3**	**k = 4**	**k = 5**	**k = 6**	**k = 7**	**k = 8**	**k = 9**	**k = 10**
MDAV	N/A	16.9326	19.5460	22.4619	26.3252	27.5184	29.6929	31.2146	33.1929
V-MDAV	N/A	16.6603	19.5460	22.4619	26.3252	27.5184	29.6929	31.2146	33.1929
C-MHM	772.62	14.8835	18.0649	21.6954	25.0690	27.6827	29.5296	30.7770	32.3488
C2-C-MHM	885.02	15.7625	18.1403	22.5123	25.5584	28.1538	29.4089	31.9773	33.7297
C3-C-MHM	990.96	16.9326	19.6913	22.2274	25.8644	29.1273	30.8186	31.5190	32.3006
C4-C-MHM	1020.69	18.7037	19.5460	23.7335	26.8053	29.9042	30.1385	31.7141	32.7920
C5-C-MHM	1068.95	20.8067	22.3369	22.4616	28.1803	30.9707	32.2069	32.7667	33.0026

Table 5.4: Percentage of I_{loss} obtained on the EIA dataset. Highlighted values denote the best outcome for each k.

	EIA								
Method	**Path Length**	**k = 3**	**k = 4**	**k = 5**	**k = 6**	**k = 7**	**k = 8**	**k = 9**	**k = 10**
MDAV	N/A	0.4829	0.6713	1.6667	1.3078	2.2141	2.9910	3.4086	3.5474
V-MDAV	N/A	0.4829	0.6713	1.2771	1.2320	2.2038	2.9193	3.2835	2.7478
C-MHM	740.69	0.3704	0.5166	0.7606	1.0554	1.6652	1.8448	1.9348	2.1129
C2-C-MHM	928.97	0.4159	0.6309	0.9423	1.1263	1.6821	1.8587	2.0378	2.1869
C3-C-MHM	992.1	0.4645	0.8101	0.9318	1.0674	1.6948	1.8359	2.0205	2.1734
C4-C-MHM	1049.04	0.5554	0.6443	1.1502	1.2983	1.7335	1.8831	2.0982	2.1756
C5-C-MHM	1271.74	1.3066	1.4934	1.6076	2.2107	2.4945	2.7590	2.9181	2.9567

Figure 5.5 shows, for each dataset, the trade-off analysis of each TSP-based method. Overall, all the outcomes remain close in the x-axis (*i.e.* denoting a similar range of I_{loss} values, with a clear exception in the case of C5-C-MHM and EIA). In contrast, all compression strategies require less time to be computed. In the case of Census, we can observe that the outcomes of C2-C-MHM and C3-C-MHM are almost aligned in the x-axis with these obtained by the C-MHM method (*i.e.* the circular markers denoting the values for each value of k are almost vertically aligned). Moreover, applying compression we always obtain lower I_{loss} values the higher the value of k, which is particularly obvious for $k = 10$. In the latter case, the original C-MHM obtains a value that is isolated from the rest, both in terms of I_{loss} and time. The outcomes of Tarragona resemble those obtained by Census, yet in this case, only C2-C-MHM seems to obtain similar values to C-MHM

in terms of I_{loss}. However, as seen in Census, these values get closer the higher the value of k is. Finally, the outcomes of the EIA dataset show that almost all strategies but C5-C-MHM obtain similar values to those achieved by C-MHM, yet with much more efficiency.

In summary, these outcomes justify the applicability of our compression method both in small and big datasets, despite being the latter the ones that reflect the trade-off gain in a more evident manner. However, different datasets may exhibit different behaviours and thus require careful analysis to select the most appropriate strategy according to the trade-off optimisation criteria.

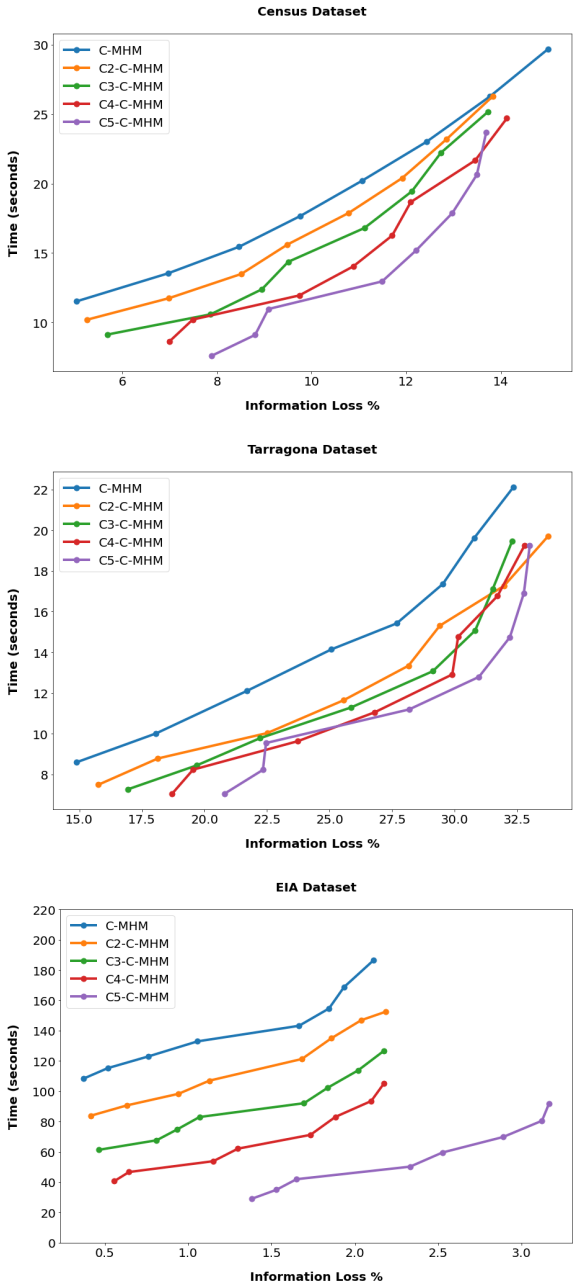


Figure 5.5: Trade-off between I_{loss} and computational time of the TSP-based methods. The circular markers of the series denote the values obtained from $k = 3$ to $k = 10$ (*i.e.* left to right).

5.4 Discussion

The search for optimal microaggregation is NP-hard. Hence, researchers have devoted extensive efforts to find good but suboptimal solutions [47]. As a result, it is becoming increasingly difficult to find new solutions that improve the state of the art, not to mention that we want to improve both the I_{loss} and the efficiency of such solutions.

Despite providing the lowest I_{loss} values, TSP-based microaggregation methods suffer from scalability issues [22]. Therefore, developing strategies to reduce the computational time of TSP heuristics is crucial. Going a step beyond, we aimed to develop a strategy that can be applied regardless of the dataset's size and improve the outcomes provided by classical TSP approaches.

As observed in Section 'Computational Time and Data Distribution', all compression strategies succeed in reducing the computational time of C-MHM. Moreover, as discussed in Section 'Microaggregation Results', and especially for high values of k, compression strategies enhance the I_{loss} values, which sometimes are close or even better than the ones achieved by the original C-MHM. In this regard, different compression values are combined with different k to study the impact of different configurations according to different types of datasets. Overall, as seen in Section 'Trade-off Analysis of TSP-based Methods', the trade-off between I_{loss} and computational time is always beneficial when applying Compress strategies, especially those with lower parameter values. Moreover, the larger the number of dataset records, the more significant the compression's impact on reducing the computational time.

Therefore, our approach enables the application of different configurations to satisfy an optimisation criteria (i.e. balancing I_{loss} and computational time according to each application context). However, the capability of our method to improve both the I_{loss} and the computational time depends on the particularities of each dataset.

5.5 Conclusion

Microaggregation heuristics have been extensively explored in the past, often with the aim to improve their data utility. However, considering the advent of Big Data and more accurate yet costly methods such as TSP, heuristics aiming to reduce the computational time of microaggregation procedures are crucial. In this Chapter, we proposed an efficient Compression-based TSP heuristic for microaggregation, which outperforms the state of the art

in terms of trade-off between I_{loss} and computational time, according to extensive experiments and comparisons. Moreover, in some cases, our Compression heuristic is able to preserve the natural distribution of data more efficiently than the original method and thus, generate groups that increase the utility while reducing the computational time. Notably, the performance of the Compress method could be extended to other areas further than microaggregation.

CHAPTER 6

Contributions to Microaggregation Optimisation

Optimisation problems involve finding the best solution among a set of possible solutions to a problem. The goal is to maximise or minimise a certain objective function while satisfying a set of constraints. In many cases, optimisation problems can be solved using various optimisation algorithms, such as gradient descent, simulated annealing, and genetic algorithms. One common issue that can arise when solving optimisation problems is falling into a local minimum. A local minimum is a solution that appears to be the best solution in the immediate vicinity of the current solution but is not the overall best solution for the problem. This can occur when the optimisation algorithm is stuck in a particular search space region rather than exploring the entire space of possible solutions. When an optimisation algorithm falls into a local minimum, recovering and finding the true global minimum may be difficult. One common way to address this issue is to use techniques such as random restarts, which involve starting the optimisation algorithm multiple times from different initial points. Another approach is to use more advanced optimisation techniques that are less likely to get stuck in local minima, such as particle swarm optimisation, differential evolution, or Bayesian optimisation. Falling into a local minimum is a major obstacle in solving optimisation problems, and it is crucial to recognise this issue and utilise effective strategies to mitigate it. A novel post-processing technique is presented in this Chapter, which aims to enhance the results of MDAV and prevent local minimums. The proposed methodology involves two steps: firstly, microaggregation will be performed using MDAV on the dataset; secondly, a post-processing approach will be implemented to refine the microaggregation outcomes.

Contents

6.1 Microaggregation Optimisation through Random Cluster Shuffling

Datasets, *per se*, are not useful, unless they are analysed using techniques like data mining (*e.g.*, to determine behaviour of attributes, to identify patterns, etc.), process mining (*e.g.* to discover processes, to check the conformance between existing process models and those reflected in the data, etc.) and, ultimately, feeding machine learning systems. Companies can analyse *their* data on their own, but they can also delegate (*i.e.* release) datasets to third parties. Note that sensitive information can be inferred from records and values in the dataset: consumer habits, location tracking, health issues, etc. This Chapter presents *Random Cluster Shuffling* (RCS), a new post-processing technique aiming at improving MDAV's results. Hence, in a first step, the dataset will be microaggregated using MDAV; in a second step, RCS will be applied to improve microaggregation. Section 6.2 describes the post-processing method. Preliminary results are shown in Section 6.3. Finally, Section 6.4 concludes the Chapter.

6.2 Our Proposal

Cluster optimisation is a topic widely explored in the literature; specifically, its application to the microaggregation problem (where cluster size is constrained), has been explored in [39], in which the Step 1 of our proposal is inspired. The main disadvantage of cluster post-processing techniques is their fast convergence into local minima, a fact related to their greedy nature. RCS modifies the basic algorithm to overcome such disadvantage by creating an event that, with a certain probability of occurrence, modifies the clusters created during the initial post-processing step to minimise the SSE. RCS, described in Algorithm 4, consists of the following steps:

1. For each element of the k-partition, the algorithm evaluates if extracting it from its current cluster C and assigning it to the nearest one improves the SSE. According to C's cardinality, two situations can occur: if $card(C) = k$, we dissect the whole cluster and perform the previous evaluation for each of its records; if $card(C) > k$, we will only evaluate the current record[1].

2. After the evaluation of each record, and given a shuffling probability and a maximum number of events, a shuffling event may occur. If such event occurs, one of the clusters created is randomly selected (C_i).

3. The cluster whose centroid is nearest to C_i is selected. Both clusters will be merged into a single cluster, in which the records will be sorted with respect to the centroid of the new cluster.

4. The new cluster can contain between $2k$ and $4k$-2 records and, thus, the cluster will be divided into new clusters, so as to satisfy the cardinality constraints of optimal microaggregation. The resulting formed clusters replace the merged ones.

5. The stopping condition of the algorithm evaluates the SSE improvement at each iteration and, if it is below 0.0001, the algorithm finishes.

[1] Our method differs from [39] in the order in which clusters are evaluated. In the former, clusters are selected sequentially, while in the latter, the next cluster is the one with the highest SSE.

Algorithm 4 Random Cluster Shuffling

D : Dataset with n *p-dimensional* data points
k : Minimum cardinality constraint
C : Matrix with clusters from microaggregated dataset
S : Shuffling event probability
N : Shuffling event maximum number
repeat
  for record in D **do**
    $C_i(record) \leftarrow$ SearchRecordCluster$(C, record)$
    if $card(C_i(record)) < k$ **then**
      DeleteRecord$(C, record)$
      $cent \leftarrow$ ComputeCentroid(C)
      $C \leftarrow$ AddRecordToNearestCentroid$(C, cent, record)$
    else
      $c_i \leftarrow$ BreakCluster$(C_i(record))$
      $cent \leftarrow$ ComputeCentroids(C)
      $C' \leftarrow$ AssignElementsToNearestCentroids$(C, cent, c_i)$
      if $SSE(C) \geq SSE(C')$ **then**
        $C \leftarrow C'$
      end if
    end if
    if RandomEvent(S) And $EventCount \leq N$ **then**
      $C_i \leftarrow$ SelectRandomCluster(C)
      $C_j \leftarrow$ SelectNearestCluster(C, C_i)
      $C_m \leftarrow$ MergeClusters(C_i, C_j)
      $C_m \leftarrow$ SortElementsFromCentroid(C_m)
      $C_s \leftarrow$ SplitIntoClustersOfSizeCloseToK(C_m, k)
      $C \leftarrow$ ReplaceByNewCluster$([C_i, C_j], C_s)$
      $EventCount++$
    end if
  end for
until NoSignificantImprovement
$D' \leftarrow$ ReplaceClustersByCentroid(D, C) **return** (D')

6.3 Experiments

We used three benchmark datasets in our experiments, see in section 2.3. The Census dataset and two datasets composed of OpenStreetMap GPS traces collected from two different cities, Barcelona and Madrid. More details on the datasets can be obtained in [41] and [22].

The R package sdcMicro [41] has been used for the MDAV implementation. Since our optimisation algorithm randomly selects the clusters which will shuffle elements, the experiments have been repeated 5 times. The outcomes of our post-processing approach are reported in Table 6.1.

From the results in Table 6.1 we can affirm that post-processing always improves the solution offered by the traditional MDAV method, since it allows the creation of clusters of variable size that are able to better capture the structure of the data. For $k = 3$, we observe that random events can negatively affect the configuration of the clusters, influencing the local minimum

Table 6.1: Outcomes of the different settings of RCS applied after the MDAV clustering. The "average" column denotes the percentage of I_{loss} (the best results for each k and dataset are highlighted in green). Gray rows correspond to the outcomes of the original MDAV method.

		Shuffling	Max	k = 3		k = 5		k = 10	
Dataset	**Method**	**probability**	**events**	**Average**	**σ**	**Average**	**σ**	**Average**	**σ**
Census	**MDAV**	NA		5.692	NA	9.088	NA	14.156	NA
	MDAV - RCS	0	0	5.483	NA	8.450	NA	12.774	NA
		1/1000	10	5.538	0.051	8.299	0.032	12.446	0.028
		1/1000	20	5.617	0.066	8.401	0.068	12.918	0.064
		10/1000	10	5.559	0.050	8.513	0.052	12.775	0.050
		10/1000	20	5.485	0.057	8.328	0.055	12.708	0.056
		100/1000	10	5.530	0.056	8.512	0.020	12.750	0.064
		100/1000	20	5.691	0.043	8.555	0.057	12.885	0.052
Barcelona	**MDAV**	NA		2.567	NA	4.285	NA	7.699	NA
	MDAV - RCS	0	0	1.682	NA	2.723	NA	4.849	NA
		1/1000	10	1.697	0.025	2.725	0.067	4.779	0.034
		1/1000	20	1.690	0.053	2.718	0.020	4.745	0.031
		10/1000	10	1.701	0.051	2.714	0.021	4.824	0.053
		10/1000	20	1.705	0.042	2.734	0.042	4.957	0.054
		100/1000	10	1.723	0.025	2.749	0.055	4.755	0.072
		100/1000	20	1.752	0.049	2.753	0.028	4.969	0.066
Madrid	**MDAV**	NA		3.188	NA	5.288	NA	8.611	NA
	MDAV - RCS	0	0	2.634	NA	4.218	NA	7.591	NA
		1/1000	10	2.700	0.077	4.154	0.036	6.690	0.065
		1/1000	20	2.663	0.039	4.251	0.060	7.323	0.046
		10/1000	10	2.642	0.011	4.294	0.060	7.233	0.034
		10/1000	20	2.672	0.023	4.215	0.027	7.260	0.033
		100/1000	10	2.668	0.055	4.248	0.045	7.135	0.024
		100/1000	20	2.585	0.028	4.274	0.060	7.006	0.065

where the solution fall. For $k = 5$ and $k = 10$, the cluster's configuration allows the impact of these modifications to be diminished. Apparently, the algorithm improves the results of the base post-processing method in configurations with higher cardinality, although this preliminary results should be extended by using different microaggregation methods to analyze if this effect is also related to the dataset distribution of records (*e.g.* if data are clustered or scattered), as seen in [39].

In general, we can observe that, while the SSE of Census and Madrid was improved substantially, the effect of the RCS post-processing method in the Barcelona dataset was outstanding, with improvements of approximately 40% over the original MDAV. Overall, we can conclude that a low probability in the occurrence of the shuffling events allows the algorithm to continue improving the SSE values without falling into a local minimum. If these

events occur with a high probability, the number of cluster modifications in the early stages of post-processing is drastically increased, resulting in a potentially unrecoverable SSE increase.

6.4 Conclusion

In this chapter, we have proposed a novel post-processing method that applies a heuristic to reduce the SSE of a clustered dataset. Given a shuffling probability and a maximum number of events, our approach exchanges elements between a randomly selected cluster and its closest one to find an alternative k-partition with lower SSE. As observed in our preliminary experiments, the outcomes support the potential of our approach.

CHAPTER 7

Conclusions

This chapter summarises the contributions of this thesis. It also outlines some lines for future work, either resulting from partially achieved goals or expected improvements.

Contents

The topics covered in this dissertation focus on the problem of microaggregation. First, we provide an extensive background on Statistical Disclosure Control. In addition, the basics of the Travelling Salesman Problem are presented. Second, we investigated various dimensions that affect this problem, such as pre- and post-processing of the dataset, and propose two methods for solving the fixed- and variable-size microaggregation problem. Moreover, extensive experiments show that our methods achieve stable results and outperform the current literature.

7.1 Publications

The main publications that support the content of this work are the following:

Journals

2021 | Armando Maya-López, Fran Casino, and Agusti Solanas, **"Improving Multivariate Microaggregation through Hamiltonian Paths and Optimal Univariate Microaggregation"**, Symmetry 13, no.6: 916. https://doi.org/10.3390/sym13060916

2022 | Armando Maya-López, Fran Casino, and Antoni Martínez-Ballesté, **"A Compression Strategy for an Efficient TSP-based Microaggregation"**, Expert Systems With Applications. https://doi.org/10.1016/j.eswa.2022.118980

Conferences

2020 Maya López, A., Solanas, A., **"Multivariate Microaggregation with Fixed Group Size Based on the Travelling Salesman Problem."**, In: Saponara, S., De Gloria, A. (eds) Applications in Electronics Pervading Industry, Environment and Society. ApplePies 2019. Lecture Notes in Electrical Engineering, vol 627. Springer, Cham.

2021 Maya-López, A., Casino, F., Solanas, A., Martínez-Ballesté, A., **"Microaggregation Optimisation Through Random Cluster Shuffling."**, In: Saponara, S., De Gloria, A. (eds) Applications in Electronics Pervading Industry, Environment and Society. ApplePies 2021. Lecture Notes in Electrical Engineering, vol 866. Springer, Cham.

7.2 Future Work

The Internet of Things facilitates the collection of large amounts of data: sensors, smartphones, and even home appliances, generate a data deluge about individuals, their context and the events in their daily life. Providers can analyse these data to extract patterns and increase knowledge about their services independently or by transferring datasets to third parties. To mitigate the Big Brother effect, *i.e.* to preserve the individuals' right to privacy, techniques in the scope of Statistical Disclosure Control must be applied. Microaggregation is a powerful technique for preserving data analysis privacy while maintaining data utility. By using clustering algorithms, microaggregation can effectively mask sensitive information in datasets by grouping data points together, thus protecting the privacy of individuals while still allowing for meaningful analysis. Additionally, various modifications and extensions of microaggregation have been developed to enhance its effectiveness and address specific privacy concerns.

However, there are still some limitations to microaggregation that need to be considered. For example, data quality can be compromised by the loss of granularity that results from grouping data points. In addition, the security of the method may be compromised by attacks from sophisticated attackers who can reverse engineer the original data from the masked dataset.

This thesis has explored the intersection of microaggregation and the travelling salesman problem (TSP), demonstrating how these two techniques can be combined to solve problems of privacy preservation. Through extensive experimentation, this study has shown that multivariate microaggregation through Hamiltonian paths and optimal univariate microaggregation

can provide a higher level of data privacy than traditional microaggregation methods. The combination of these two approaches allows for the creation a new dataset that preserves the statistical characteristics of the original data while ensuring that individual-level information is obscured.

Furthermore, this work has investigated the development of a compression strategy for efficient TSP -based microaggregation. The proposed method provides a novel approach to anonymising data that preserves privacy while reducing the size of the dataset so that it can be processed more efficiently. One of the main advantages of this approach is its scalability, as the method can be applied to large datasets without compromising the accuracy of the anonymised data.

The findings of this thesis have important implications for data privacy in various fields, including healthcare, finance, and social sciences. The following paragraphs summarise the main contributions of this thesis to microaggregation:

1. In Chapter 3, we have described our algorithm and empirically shown that it performs better than off-the-shelf, well-known microaggregation methods for low cardinalities over benchmark datasets frequently used in the literature. Our proposal represents the first step towards creating a more solid TSP-based microaggregation algorithm that would outperform current methods, not only for small cardinalities but for any k as well, and it opens the door to a fruitful research line in the field of SDC. As further work, we plan to improve our clustering algorithm over Hamiltonian paths permutations and test alternative TSP heuristics.

2. In Chapter 4, despite these relevant results, there is still much to be done in the study of microaggregation and data protection. Future work will focus on scaling up$(HM)^2$-Micro to high-dimensional and very- large datasets. Considering the continuous growth of Big Data and cloud computing, adapting our solution to a distributed computation environment is paramount. Moreover, splitting-based approaches and modifying such TSP heuristics to leverage more lightweight microaggregation-based approaches are interesting research paths to follow. Even though the privacy parameter k is typically low (*i.e.* 3,4,5,6), we plan to investigate the impact of larger values of k on our solution. Since microaggregation is essentially a data-, we will explore how our solution can be adapted to data structures from specific domains such as healthcare, transportation, energy, etc. With $(HM)^2$-Micro, we have set the ground for the study of multivariate

microaggregation meta-heuristics from a new perspective that might continue in the future.

3. Chapter 5 has presented a heuristic to reduce the SSE of a clustered dataset. Given a shuffling probability and a maximum number of events, our approach exchanges elements between a randomly selected cluster and its closest one to find an alternative k-partition with a lower SSE. As observed in our preliminary experiments, the outcomes support the potential of our approach. Future work will focus on studying the impact of a more extensive set of parameter configurations and using other benchmark datasets.

4. In Chapter 5, we presented an efficient Compression-based TSP heuristic for microaggregation, which outperforms the state of the art regarding the trade-off between IL and computational time, according to extensive experiments and comparisons. Moreover, in some cases, our Compression heuristic can preserve the natural distribution of data more efficiently than the original method and, thus, generate groups that increase utility while reducing the computational time. It is noteworthy that the performance of the Compress method could be extended beyond microaggregation to other areas. The latter, along with the analysis of other group protection heuristics, is part of our future work.

Bibliography

[1] David L Applegate, Robert E Bixby, Vasek Chvatal, and William J Cook. *The traveling salesman problem: a computational study*. Princeton university press, 2006.

[2] Michael Barbaro, Tom Zeller, and Saul Hansell. A face is exposed for aol searcher no. 4417749. *New York Times*, 2006.

[3] Edgar Batista and Agusti Solanas. Process Mining in Healthcare: A Systematic Review. In *9th Int. Con. on Information, Intelligence, Systems and Appl.*, pages 1–6. IEEE, 2018.

[4] Fran Casino, Josep Domingo-Ferrer, Konstantinos Patsakis, Domenec Puig, and Agusti Solanas. A k-anonymous approach to privacy preserving collaborative filtering. *Journal of Computer and System Sciences*, 81(6):1000–1011, 2015.

[5] Fran Casino, Constantinos Patsakis, and Agusti Solanas. Privacy-preserving collaborative filtering: A new approach based on variable-group-size microaggregation. *Electronic Commerce Research and Applications*, 38:100895, 2019.

[6] Chin-Chen Chang, Yu-Chiang Li, and Wen-Hung Huang. Tfrp: An efficient microaggregation algorithm for statistical disclosure control. *Journal of Systems and Software*, 80(11):1866–1878, 2007.

[7] Sharlee Climer, Weixiong Zhang, and Thorsten Joachims. Rearrangement clustering: Pitfalls, remedies, and applications. *Journal of Machine Learning Research*, 7(6), 2006.

[8] J. Domingo-Ferrer, Antoni Martinez-Ballesté, J.M. Mateo-Sanz, and Francesc Sebé. Efficient multivariate data-oriented microaggregation. *The VLDB Journal*, 15:355–369, 2006.

[9] Josep Domingo-Ferrer and Josep Maria Mateo-Sanz. Practical data-oriented microaggregation for statistical disclosure control. *Knowledge and Data Engineering, IEEE Transactions on*, 14(1):189–201, 2002.

[10] Josep Domingo-Ferrer, Francesc Sebé, and Agusti Solanas. A polynomial-time approximation to optimal multivariate microaggregation. *Computers and Mathematics with Applications*, 55(4):714 – 732, 2008.

[11] Ebaa Fayyoumi and B John Oommen. A survey on statistical disclosure control and micro-aggregation techniques for secure statistical databases. *Software: Practice and Experience*, 40(12):1161–1188, 2010.

[12] Michael Hahsler and Kurt Hornik. TSP - infrastructure for the traveling salesperson problem. *Journal of Statistical Software*, 23(2):1–21, 12 2007.

[13] S.L. Hansen and S. Mukherjee. A polynomial algorithm for optimal univariate microaggregation. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):1043–1044, 2003.

[14] Brook Heaton and Sumitra Mukherjee. Record ordering heuristics for disclosure control through microaggregation. In *Proceedings of the International Conference on Advances in Communication and Information Technology,*, 2011.

[15] Keld Helsgaun. An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, 2000.

[16] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, 1998.

[17] Michael Laszlo and Sumitra Mukherjee. Minimum spanning tree partitioning algorithm for microaggregation. *IEEE Transactions on Knowledge and Data Engineering*, 17(7):902–911, 2005.

[18] Chee Sun Liew, Assad Abbas, Prem Prakash Jayaraman, Teh Ying Wah, Samee U Khan, et al. Big data reduction methods: a survey. *Data Science and Engineering*, 1(4):265–284, 2016.

[19] Jun-Lin Lin, Tsung-Hsien Wen, Jui-Chien Hsieh, and Pei-Chann Chang. Density-based microaggregation for statistical disclosure control. *Expert Systems with Applications*, 37(4):3256–3263, 2010.

[20] Shen Lin and Brian W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516, 1973.

[21] Antoni Martinez-Balleste, Pablo Alejandro Perez-Martines, and Agusti Solanas. The pursuit of citizens' privacy: a privacy-aware smart city is possible. *IEEE Communications Magazine*, 51(6):136–141, 2013.

[22] Armando Maya-López, Fran Casino, and Agusti Solanas. Improving multivariate microaggregation through hamiltonian paths and optimal univariate microaggregation. *Symmetry*, 13(6), 2021.

[23] Abid Mehmood, Iynkaran Natgunanathan, Yong Xiang, Guang Hua, and Song Guo. Protection of big data privacy. *IEEE access*, 4:1821–1834, 2016.

[24] David R. Morrison, Sheldon H. Jacobson, Jason J. Sauppe, and Edward C. Sewell. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19:79–102, 2016.

[25] Reza Mortazavi and Saeed Jalili. Fast data-oriented microaggregation algorithm for large numerical datasets. *Knowledge-Based Systems*, 67:195 – 205, 2014.

[26] Reza Mortazavi, Saeed Jalili, and Hojjat Gohargazi. Multivariate microaggregation by iterative optimization. *Applied intelligence*, 39(3):529–544, 2013.

[27] Christian Nilsson. Heuristics for the traveling salesman problem. Technical report, Linköping University, Sweden, `http://www.ida.liu.se/~TDDB19/reports_2003/htsp.pdf`, 2003.

[28] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

[29] Eneko Osaba, Xin-She Yang, and Javier Del Ser. Traveling salesman problem: a perspective review of recent research and new results with bio-inspired metaheuristics. *Nature-Inspired Computation and Swarm Intelligence*, pages 135–164, 2020.

[30] Costas Panagiotakis and Georgios Tziritas. Successive group selection for microaggregation. *IEEE Transactions on Knowledge and Data Engineering*, 25(5):1191–1195, 2011.

[31] Daniel J Rosenkrantz, Richard E Stearns, and Philip M Lewis, II. An analysis of several heuristics for the traveling salesman problem. *SIAM journal on computing*, 6(3):563–581, 1977.

[32] Pierangela Samarati. Protecting respondents identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, Nov 2001.

[33] Ali Seyed Shirkhorshidi, Saeed Aghabozorgi, Teh Ying Wah, and Tutut Herawan. Big data clustering: a review. In *International conference on computational science and its applications*, pages 707–720. Springer, 2014.

[34] David Bernard Shmoys, JK Lenstra, AHG Rinnooy Kan, and Eugène L Lawler. *The traveling salesman problem*, volume 12. John Wiley & Sons, Incorporated, 1985.

[35] Agusti Solanas, Fran Casino, Edgar Batista, and Robert Rallo. Trends and Challenges in Smart Healthcare Research: A Journey from Data to Wisdom. In *3rd IEEE Int. Forum on Research and Technologies for Society and Industry*, pages 1–6, Modena, Italy, 2017.

[36] Agusti Solanas, Ursula González-Nicolás, and Antoni Martínez-Ballesté. A variable-mdav-based partitioning strategy to continuous multivariate microaggregation with genetic algorithms. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, 2010.

[37] Agusti Solanas, Antoni Martinez-Balleste, and Josep Maria Mateo-Sanz. Distributed architecture with double-phase microaggregation for the private sharing of biomedical data in mobile health. *IEEE Transactions on Information Forensics and Security*, 8(6):091–910, 2013.

[38] Agusti Solanas and Antoni Martínez-Ballesté. VMDAV: A multivariate microaggregation with variable group size. *17th COMPSTAT Symposium of the IASC, Rome*, pages 917–925, 2006.

[39] Agusti Solanas, Gloria Pujol, Antoni Martinez-Balleste, and Josep M Mateo-Sanz. A post-processing method to lessen k-anonymity dissimilarities. In *2008 Third International Conference on Availability, Reliability and Security*, pages 1060–1066. IEEE, 2008.

[40] Marc Solé, Victor Muntés-Mulero, and Jordi Nin. Efficient microaggregation techniques for large numerical data volumes. *International Journal of Information Security*, 11(4):253–267, 2012.

[41] Matthias Templ. Statistical disclosure control for microdata using the r-package sdcmicro. *Transactions on Data Privacy*, 1(2):67–85, 2008.

[42] Vicenç Torra. Microaggregation for categorical variables: A median based approach. In *Privacy in statistical databases*, pages 162–174. Springer, 2004.

[43] Dinusha Vatsalan, Ziad Sehili, Peter Christen, and Erhard Rahm. *Privacy-Preserving Record Linkage for Big Data: Current Approaches and Research Challenges*, pages 851–895. Springer International Publishing, Cham, 2017.

[44] Gaoming Yang, Xinxin Ye, Xianjin Fang, Rongshi Wu, and Li Wang. Associated attribute-aware differentially private data publishing via microaggregation. *IEEE Access*, 8:79158–79168, 2020.

[45] H. Ye, X. Cheng, M. Yuan, L. Xu, J. Gao, and C. Cheng. A survey of security and privacy in big data. In *2016 16th International Symposium on Communications and Information Technologies (ISCIT)*, pages 268–272, Sep. 2016.

[46] Taskeen Zaidi. Travelling salesman problem and its applications. *International Journal of Mathematics, Game Theory, and Algebra*, 29(2/3):73–80, 2020.

[47] A. Zigomitros, F. Casino, A. Solanas, and C. Patsakis. A survey on privacy properties for data publishing of relational data. *IEEE Access*, 8:51071–51099, 2020.

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO STATISTICAL DISCLOSURE CONTROL: ENHANCING MULTIVARIATE MICROAGGREGATION USING GRAPH THEORY
Armando Maya-López

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO STATISTICAL DISCLOSURE CONTROL: ENHANCING MULTIVARIATE MICROAGGREGATION USING GRAPH THEORY
Armando Maya-López

UNIVERSITAT
ROVIRA i VIRGILI