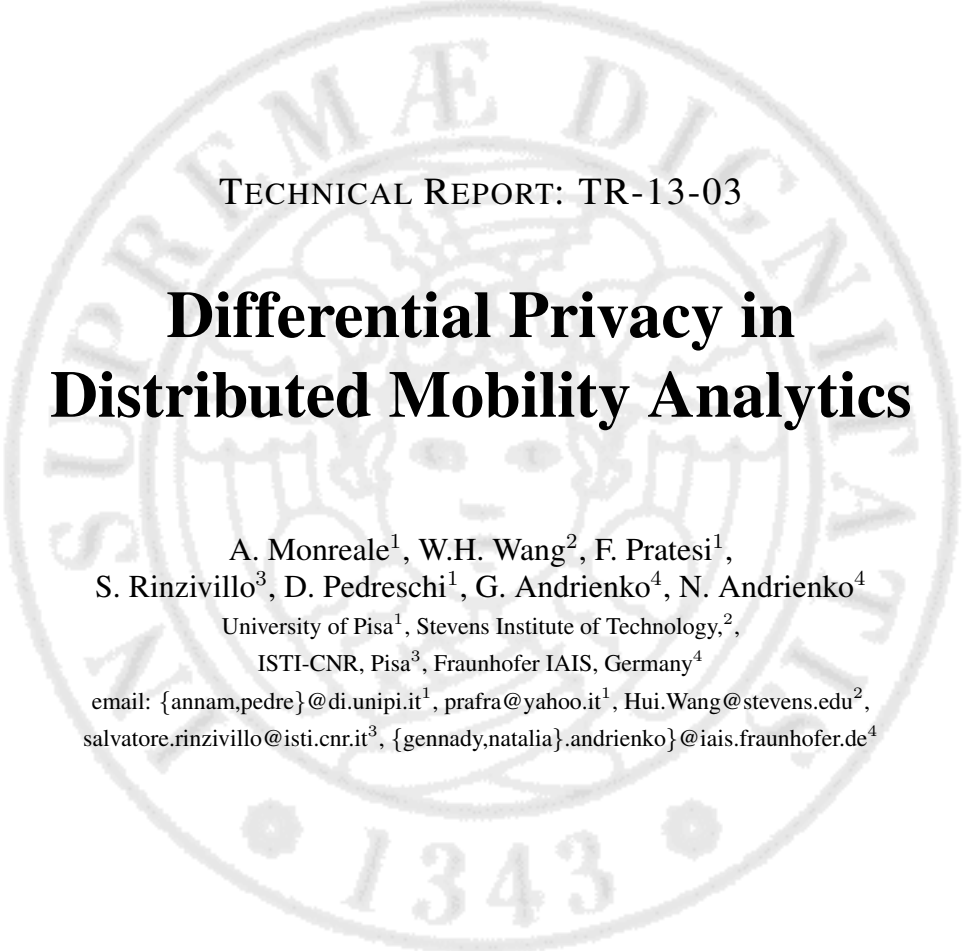
UNIVERSITÀ DI PISA
DIPARTIMENTO DI INFORMATICA



TECHNICAL REPORT: TR-13-03

Differential Privacy in Distributed Mobility Analytics

A. Monreale[1], W.H. Wang[2], F. Pratesi[1],
S. Rinzivillo[3], D. Pedreschi[1], G. Andrienko[4], N. Andrienko[4]
University of Pisa[1], Stevens Institute of Technology,[2],
ISTI-CNR, Pisa[3], Fraunhofer IAIS, Germany[4]
email: {annam,pedre}@di.unipi.it[1], prafra@yahoo.it[1], Hui.Wang@stevens.edu[2],
salvatore.rinzivillo@isti.cnr.it[3], {gennady,natalia}.andrienko}@iais.fraunhofer.de[4]

31 January 2013

ADDRESS: Largo B. Pontecorvo 3, 56127 Pisa, Italy. TEL: +39 050 2212700 FAX: +39 050 2212726

Differential Privacy in Distributed Mobility Analytics

A. Monreale[1], W.H. Wang[2], F. Pratesi[1],
S. Rinzivillo[3], D. Pedreschi[1], G. Andrienko[4], N. Andrienko[4]
University of Pisa[1], Stevens Institute of Technology,[2],
ISTI-CNR, Pisa[3], Fraunhofer IAIS, Germany[4]
email: {annam,pedre}@di.unipi.it[1], prafra@yahoo.it[1], Hui.Wang@stevens.edu[2],
salvatore.rinzivillo@isti.cnr.it[3], {gennady,natalia}.andrienko}@iais.fraunhofer.de[4]

31 January 2013

Abstract

Movement data are sensitive, because people's whereabouts may allow re-identification of individuals in a de-identified database and thus can potentially reveal intimate personal traits, such as religious or sexual preferences. In this paper, we focus on a distributed setting in which movement data from individual vehicles are collected and aggregated by a centralized station. We propose a novel approach to privacy-preserving analytical processing within such a distributed setting, and tackle the problem of obtaining aggregated traffic information while preventing privacy leakage from data collection and aggregation. We study and analyze three different solutions based on the differential privacy model and on sketching techniques for efficient data compression. Each solution achieves different trade-off between privacy protection and utility of the transformed data. Using real-life data, we demonstrate the effectiveness of our approaches in terms of data utility preserved by the data transformation, thus bringing empirical evidence to the fact that the "privacy-by-design" paradigm in big data analytics has the potential of delivering high data protection combined with high quality even in massively distributed techno-social systems.

1 Introduction

The widespread availability of low cost GPS devices enables the collection of data about movements of people and objects at a large scale. Understanding of the human mobility behavior in a city is important for improving the use of city space and accessibility of various places and utilities, managing the traffic network, and reducing traffic jams. Generalization and aggregation of individual movement data can provide an overall description of traffic flows in a given time interval and their variation over time. Intuitively, movement data of multiple individual devices can be collected and aggregated by a central station. However, this centralized setting entails two important

problems: a) the amount of information to be collected and processed may exceed the capacity of the storage and computational resources; and b) the raw data describes the mobility behavior of the individuals with great detail that could enable the inference of very sensitive information related to the personal private sphere.

Some recent work [29, 20, 2] have investigated how to aggregate distributed mobility data efficiently. For instance, Andrienko et al. [2] propose a method for generalization and aggregation of movement data that requires all individual moving trajectories be transformed into aggregate flows between areas. Though these work consider releasing statistic information instead of raw trajectories to the central station, there still may exist privacy leakage. For instance, the analysis of low-density aggregate traffic flows (e.g., in rural areas) may still reveal the identity of the vehicles involved in these flows.

In order to solve these problems, we propose a privacy-preserving *distributed* analytical processing framework for the aggregation of movement data. We assume that on-board location devices in vehicles continuously trace the positions of the vehicles and periodically send statistical information about their movements to a central station. The central station, which we call *coordinator*, will store the received statistical information and compute a summary of the traffic conditions of the whole territory, based on the information collected from individual vehicles. Since the coordinator can be untrusted, we design privacy-preserving methods for each individual vehicle participant that provides formal privacy guarantee, meaning that the statistic information revealed to the coordinator will not be swayed too much by whether or not a specific individual participates. The basic idea behind our approach is that even radical forms of data randomization, capable of yielding strong protection of personal mobility data for each participating vehicle, can be adopted in our setting while still allowing a correct reconstruction of aggregated traffic information at the coordinator side. Leveraging on the differential privacy model on one side, and on large real-life movement data on the other side, we are able to show how the theoretical and empirical outcomes of our study achieves a high-level trade-off between data privacy and analytical utility. We believe that the research reported here brings evidence to the fact that the "privacy-by-design" paradigm in big data analytics has the potential of delivering high data protection combined with high quality even in massively distributed techno-social systems. When we have a clear analytical goal to realize, e.g., the continuous monitoring of traffic flows, it is possible to design a privacy-preserving process that, as in our study, solves the problem delivering results with a bounded (small) quality-loss within a framework where the risk of privacy leakage is also bounded (and very small).

We have the following contributions. First, to protect individual privacy, we propose three data transformation methods based on the well-known differential privacy model. Each solution is characterized by a different trade-off between privacy and data utility. Second, to further reduce the amount of information that each vehicle communicates to the central station, we propose to apply sketch techniques to the differentially private data to obtain a compressed representation. The central station is able to reconstruct the movement data represented by the sketched data that, although transformed for guaranteeing privacy, preserve some important properties of the original data that make them useful for mobility analysis. We validate the robustness and efficiency of our privacy-preserving data aggregation methods by extensive experiments on large,

real GPS data.

The remainder of the paper is organized as follows. Section 2 discusses the related work. Section 3 introduces background information and definitions. Section 4 describes the system architecture and states the problem. Sections 5&6 present our proposals for providing a privacy-preserving framework; in particular, the first one describes the details of the node computation and the second one the computation of the controller. Experimental results from applying our methods to real-world data are presented and discussed in Section 7. Lastly, Section 8 concludes the paper.

2 Related Work

The existing methods of privacy-aware publishing of trajectories can be categorized into two classes: (1) generalization/suppression based data perturbation, and (2) differential privacy.

Generalization/suppression based data perturbation techniques. There have been some recent works on privacy-preserving publishing of spatio-temporal moving points by using the generalization/suppression techniques. The mostly widely used privacy model of these work is adapted from what so called k-anonymity [34, 33], which requires that an individual should not be identifiable from a group of size smaller than k based on their quasi-identifies (QIDs), i.e., a set of attributes that can be used to uniquely identify the individuals. [1] proposes the (k, δ)-anonymity model that exploits the inherent uncertainty of the moving object's whereabouts, where δ represents possible location imprecision. Terrovitis and Mamoulis [36] assume that different adversaries own different, disjoint parts of the trajectories. Their anonymization technique is based on *suppression* of the dangerous observations from each trajectory. Yarovoy et al. [39] consider timestamps as the quasi-identifiers, and define a method based on *k-anonymity* to defend against an attack called *attack graphs*. Monreale et al. [28] propose a spatial generalization approach to achieve k-anonymity. A general problem of these k-anonymity based privacy preserving techniques is that these techniques assume a certain level of background knowledge of the attackers, which may not be available to the data owner in practice.

Differential privacy. The recently proposed concept of *differential privacy* (DP) [16] addresses the above issue. There are two popular mechanisms to achieve differential privacy, *Laplace* mechanism that supports queries whose outputs are numerical [16] and *exponential mechanism* that works for any queries whose output spaces are discrete [25]. The basic idea of the Laplace mechanism is to add noise to aggregate queries (e.g., counts) or queries that can be reduced to simple aggregates. The Laplace mechanism has been widely adopted in many existing work for various data applications. For instance, [37, 11] present methods for minimizing the worst-case error of count queries; [4, 14] consider the publication of data cubes; [21, 38] focus on publishing histograms; and [26, 23] propose the methods of releasing data in a differential private way for data mining. On the other hand, for the analysis whose outputs are not real or make no sense after adding noise, the exponential mechanism selects an output from the output domain, $r \in R$, by taking into consideration its score of a given utility function q in a differentially private manner. It has been applied for the publication of audition results

[25], coresets [18], frequent patterns [6] and decision trees [19].

Recently some attention is paid to distributed private data analysis. In this setting, n parties each holding some sensitive data wish to compute some aggregate statistics over all parties' data with or without a centralized coordinator. [5, 17] prove that when computing the sum of all parties' inputs without a central coordinator, any differentially-private multi-party protocol with a small number of rounds and small number of messages must have large error. To the best of our knowledge, Rastogi et al. [31] and Chan et al. [35] were the first ones to consider the problem of privately aggregating sums over multiple time periods. Both of them consider untrusted coordinator, in particular, malicious coordinator, and use both encryption and differential privacy for the design of privacy-preserving data aggregation methods. Compared with their work, we focus on semi-honest coordinator, with the aim as designing privacy-preserving techniques by adding meaningful noises to improve data utility, which is an issue that is rarely discussed in both [31, 35]. We agree that our methods can be further enforced to against the malicious coordinator by applying the encryption methods in [31, 35]. In our previous work [27], we considered the same distributed aggregation framework as in this paper, and proposed to apply ϵ-differential privacy model to the framework. However, we rarely addressed the utility issue in [27]. This paper improves [27] significantly in terms of the design of alternative approaches that address the trade-off between privacy and data utility.

There are some work on publishing differentially private spatial data. Chen et al. [7] propose to release a prefix tree of trajectories with injected Laplace noise. Each node in the prefix tree contains a doublet in the form of $< tr(v), c(v) >$, where $tr(v)$ is the set of trajectories of the prefix v, and $c(v)$ is a version of $|tr(v)|$ with Laplace noise. Compared with our work, the prefix tree in [7] is *data-dependent*, i.e., it should have a different structure when the underlying database changes. In our work, the frequency vector is *data-independent*. Cormode et al. present a solution to publish differentially private spatial index (e.g., quadtrees and kd-trees) to provide a private description of the data distribution [11]. Its main utility concern is the accuracy of multi-dimensional range queries (e.g., how many individuals fall within a given region). Therefore, the spatial index only stores the count of a specific spatial decomposition. It does not store the movement information (e.g., how many individuals move from location i to location j) as in our work. In another paper, Cormode et al. [12] proposes to publish a contingency table of trajectory data. The contingency table can be indexed by specific locations so that each cell in the table contains the number of people who commute from the given source to the given destination. The contingency table is very similar to our frequency vector structure. However, [12] has a different focus from ours: we investigate how to publish the frequency vector in a differential privacy way, while [12] address the sparsity issue of the contingency table and presents a method of releasing a compact summary of the contingency table with Laplace noise.

3 Preliminaries

In this section, we introduce the preliminaries of our paper.

3.1 Movement Data Representation

Given a 2-D space territory $\mathbf{R}^2$, we define a *trajectory* as a sequence of triples $T = \{\langle l_1, t_1\rangle, \ldots, \langle l_n, t_n\rangle\}$, where t_i (with $i = 1 \ldots n$) denotes a timestamp such that $\forall_{1 \leq i < n}\ t_i < t_{i+1}$ and $l_i = (x_i, y_i)$ are points in $\mathbf{R}^2$. Intuitively, each pair $\langle l_i, t_i\rangle$ indicates that the object is in the position $l_i = \langle x_i, y_i\rangle$ at time t_i. In a time interval τ, each moving object can have multiple trajectories. We do not require that each trajectory is *complete*, i.e., locations may be missing at some timestamps. Moreover, each trajectory may contain repeated sub-trajectories (i.e., the object may move between locations l_i and l_j back and forth for multiple times). For example, a vehicle can have two trajectories: $T_1 = \{\langle a, t_1\rangle, \langle b, t_2\rangle, \langle c, t_3\rangle, \langle a, t_4\rangle\}$ and $T_2 = \{\langle a, t_5\rangle, \langle b, t_6\rangle, \langle a, t_7\rangle, \langle b, t_8\rangle\}$. We assume that the territory $\mathbf{R}^2$ is subdivided into cells $C = \{c_1, c_2, \ldots, c_p\}$ which compose a partition of the territory. For this partition we can use existing division of the territory (e.g., census sectors, road segments, etc.) or we can determine a data-driven partition as will be discussed in Section 7.2. During travel a user may move from one cell to another. We use g to denote the function that applies the spatial generalization to a trajectory. Given a trajectory T this function generates the generalized trajectory $g(T)$, i.e. a sequence of *moves* with temporal annotations, where a *move* is an pair (l_{c_i}, l_{c_j}) indicating that the moving object moves from the cell c_i to the *adjacent* cell c_j. Note that l_{c_i} denotes the pair of spatial coordinates representing the centroid of the cell c_i; in other words $l_{c_i} = \langle x_{c_i}, y_{c_i}\rangle$. The *temporal annotated move* is the quadruple $(l_{c_i}, l_{c_j}, t_{c_i}, t_{c_j})$ where l_{c_i} is the location of the origin, l_{c_j} is the location of the destination and the t_{c_i}, t_{c_j} are the time information associate to l_{c_i} and l_{c_j}. As a consequence, we define a generalized trajectory as follows.

Definition 3.1 [Generalized Trajectory] *Given a trajectory $T = \langle l_1, t_1\rangle, \ldots, \langle l_n, t_n\rangle$. Let $C = \{c_1, c_2, \ldots, c_p\}$ be the territory partition. A generalized version of T is a sequence of temporal annotated moves $T_g = \{(l_{c_1}, l_{c_2}, t_{c_1}, t_{c_2})\ (l_{c_2}, l_{c_3}, t_{c_2}, t_{c_3})\ \ldots\ (l_{c_{m-1}}, l_{c_m}, t_{c_{m-1}}, t_{c_m})\}$ with $m <= n$.*

Now, we show how to construct *frequency distribution vectors* of generalized trajectories. First, we define the function *Move Frequency* (MF) to compute how many times the move appears in a generalized trajectory T_g within a given time interval. More formally, we define the *move frequency* function as follows.

Definition 3.2 [Move Frequency] *Let T_g be a generalized trajectory and let (l_{c_i}, l_{c_j}) be a move. Let τ be a temporal interval. The* move frequency *function is defined as:*

$$MF(T_g, (l_{c_i}, l_{c_j}), \tau) = |\{(l_{c_i}, l_{c_j}, t_i, t_j) \in T_g | t_i \in \tau \wedge t_j \in \tau\}|.$$

For any move (l_{c_i}, l_{c_j}), the value of $MF(T_g, (l_{c_i}, l_{c_j}), \tau)$ can be any non-negative integer. For instance, consider the trajectory $T_2 = \{\langle a, t_5\rangle, \langle b, t_6\rangle, \langle a, t_7\rangle, \langle b, t_8\rangle\}$. Assume location a and b are located in the cells c_1 and c_2, respectively. Then, the generalized version of T_2 is $T_g = \{(l_{c_1}, l_{c_2}, t_5, t_6), (l_{c_2}, l_{c_1}, t_6, t_7)(l_{c_1}, l_{c_2}, t_7, t_8)\}$ and $MF(T_g, (l_{c_1}, l_{c_2}), [t_5, t_8]) = 2$. This function can be easily extended for taking into consideration a set of generalized trajectories $\mathcal{T}^{\mathcal{G}}$. In this case, the information computed by the function represents the total number of movements from the cell c_i to the cell c_j in a time interval in the set of trajectories.

Definition 3.3 [Global Move Frequency] *Let $\mathcal{T}^{\mathcal{G}}$ be a set of generalized trajectories and let (l_{c_i}, l_{c_j}) be a move. Let τ be a time interval. The* global move frequency *function is defined as:*

$$GMF(\mathcal{T}^{\mathcal{G}}, (l_{c_i}, l_{c_j}), \tau) = \sum_{\forall T_g \in \mathcal{T}^{\mathcal{G}}} MF(T_g, (l_{c_i}, l_{c_j}), \tau).$$

The number of movements between two cells computed by either the function MF or GMF describes the amount of traffic flow between the two cells in a specific time interval. This information can be represented by a frequency vector. To define the frequency vector, we first define *vector of moves*.

Definition 3.4 [Vector of Moves] *Let $C = \{c_1, c_2, \ldots, c_p\}$ be the set of the cells composing the territory partition. The* vector of moves *M is a vector of size $s = |\{(c_i, c_j) | c_i \ is \ adjacent \ to \ c_j\}|$, in which each element $M[k] = (l_{c_i}, l_{c_j})$, where $1 \leq k \leq s$, is the* move *from the cell c_i to the adjacent cell c_j.*

Now we are ready to define the frequency vector.

Definition 3.5 [Frequency Vector] *Let $C = \{c_1, c_2, \ldots, c_p\}$ be the cells that compose the territory partition and let M be its vector of moves. Given a set of generalized trajectories $\mathcal{T}^{\mathcal{G}}$ in a time interval τ, its* frequency vector *f is a vector of size $s = |\{(c_i, c_j) | c_i \ is \ adjacent \ to \ c_j\}|$, in which each element $f[k] = GMF(\mathcal{T}^{\mathcal{G}}, M[k], \tau)$.*

3.2 Differential Privacy

Differential privacy implies that adding or deleting a single record does not significantly affect the result of any analysis. Intuitively, differential privacy can be understood as follows. Let a database D include a private data record d_i about an individual *i*. By querying the database, it is possible to obtain certain information I(D) about all data and information I(D-d_i) about the data without the record d_i. The difference between I(D) and I(D-d_i) may enable inferring some private information about the individual *i*. Hence, it must be guaranteed that I(D) and I(D-d_i) do not significantly differ for any individual *i*.

The formal definition of differential privacy [16] is the following. Here the parameter, ϵ, specifies the level of guaranteed privacy.

Definition 3.6 [ϵ-differential privacy] [16] *A privacy mechanism A gives ϵ-differential privacy if for any dataset D_1 and D_2 differing on at most one record, and for any possible output D' of A we have $Pr[A(D_1) = D'] \leq e^{\epsilon} \times Pr[A(D_2) = D']$ where the probability is taken over the randomness of A.*

Two principal techniques for achieving differential privacy have appeared in the literature, one for numerical outputs [16] and the other for outputs of arbitrary types [25]. A fundamental concept of both techniques is the global sensitivity of a function mapping underlying datasets to (vectors of) real numbers.

Definition 3.7 [Global Sensitivity] [15] *For any function $f : D \rightarrow R^d$, its sensitivity is $\Delta f = max_{D_1, D_2} ||f(D_1) - f(D_2)||_1$ for all D_1, D_2 differing in at most one record.*

In particular, when $d = 1$, the sensitivity of f is the maximum difference in the values that the function f may take on a pair of databases that differ in only one element.

For the analysis whose outputs are numerical, a standard mechanism to achieve differential privacy is to add Laplace noise to the true output of a function. Dwork et al. [16] propose the Laplace mechanism which takes as inputs a dataset D, a function f, and the privacy parameter ϵ. The magnitude of the noise added conforms to a Laplace distribution with the probability density function $p(x|\lambda) = \frac{1}{2\lambda}e^{-|x|/\lambda}$, where $\lambda = \Delta f/\epsilon$.

Theorem 3.1 *[16] For any function $f : D \rightarrow R^d$ over an arbitrary domain D, the mechanism A $A(D) = f(D) + Lap((\Delta f/\epsilon)$ gives ϵ-differential privacy.*

A relaxed version of differential privacy, named (ϵ, δ)-differential privacy [16], allows a small amount of privacy loss due to a variation in the output distribution for the privacy mechanism A.

Definition 3.8 [(ϵ, δ)-differential privacy] [16] *A privacy mechanism A gives (ϵ, δ)-differential privacy if for any dataset D_1 and D_2 differing on at most one record, and for any possible output D' of A we have $Pr[A(D_1) = D'] \leq e^{\epsilon} \times Pr[A(D_2) = D'] + \delta$ where the probability is taken over the randomness of A.*

Note that when $\delta = 0$, (ϵ, δ)-differential privacy is equivalent to ϵ-differential privacy. In the remaining of this paper we will study how the trade-off between privacy and data utility changes in our specific problem setting when we use differential private methods based on ϵ-differential privacy (Definition 3.6) and (ϵ, δ)-differential privacy (Definition 3.8).

4 Problem Definition

4.1 System Architecture

We consider a system architecture similar to the one described in [9]. In particular, we assume a distributed-computing environment comprising a collection of k (trusted) remote nodes (e.g., individual vehicles) and a designated coordinator site. Streams of traffic data updates arrive continuously at remote nodes, while the coordinator site is responsible for generating approximate answers to periodic user queries posed over the aggregates of remotely-observed streams across all nodes. Each remote node exchanges messages only with the coordinator, providing it with state information on its (locally observed) streams. There is no communication between remote nodes.

In our scenario, the coordinator is responsible for computing the aggregation of movement data on a territory by combining the information received by each node. In order to obtain the aggregation of the movement data in the centralized setting, we need to generalize all the trajectories by using the cells of a partition of the territory. In our distributed setting we assume that the partition of the territory, i.e., the set of cells $C = \{c_1, \ldots, c_m\}$ used for the generalization, is known by both all the nodes and the coordinator. Each node, that represents a vehicle that moves in this territory, in a given

time interval observes a set of trajectories, that are sequence of spatio-temporal points, generalize them and contributes to the computation of the *global vector* representing the movement data aggregation.

Formally, each remote node $j \in \{1, \ldots, k\}$ observes local update streams that incrementally render a distinct frequency distribution vector f^j over data elements; that is, $f^j[v]$ denotes the frequency of element v observed locally at remote node j. The coordinator for each computes the *global frequency distribution vector* $F = \sum_{j=1}^{k} f^j$.

4.2 Privacy Model

We consider sensitive information as any information from which the typical mobility behavior of a user may be inferred. This information is considered sensitive for two main reasons: 1) typical movements can be used to identify the drivers who drive specific vehicles even when a simple de-identification of the individual in the system is applied; and 2) the places visited by a driver could identify specific sensitive areas such as clinics, hospitals and routine locations such as user's home and work.

In our setting, we assume that each node in our system is honest; in other words we do not consider attacks at the node level. We also assume that the coordinator is untrusted. There are two types of untrusted coordinators: (i) *semi-honest* coordinator who will try to infer the sensitive mobility information from the inputs of nodes, but otherwise follows the protocol correctly, and (ii) *malicious* coordinator who may have arbitrary auxiliary information to help break the protocol. For example, the coordinator may be able to obtain real mobility statistics information from other sources, such as from public datasets on the web, or through personal knowledge about a specific participant [35]. In this paper, we focus on designing privacy-preserving technique to defend against a semi-honest coordinator. With weakened assumption of the strength of coordinator, we aim at designing privacy-preserving techniques that can provide meaningful data utility.

Unfortunately, releasing frequency of moves instead of raw trajectory data to the coordinator is not privacy-preserving, as the intruder may still infer the sensitive typical movement information of the driver. As an example, the attacker could learn the driver's most frequent move; this information can be very sensitive because such move usually correspond to user's transportation between home and work place. Therefore, our goal is to compute a distributed aggregation of movement data for a comprehensive exploration of them while preserving privacy. In particular, we aim to find effective privacy mechanisms to protect the frequency information associated to each move. Our problem can be defined formally as the following.

Definition 4.1 [Problem Definition] *Given a set of cells* $C = \{c_1, \ldots, c_p\}$ *and a set* $V = \{V_1, \ldots V_k\}$ *of vehicles, the* privacy-aware distributed mobility data analytics problem *consists in computing, in a specific time interval* τ*, the* $F^\tau(V) = [f_1, f_2, \ldots, f_s]$*, where* $f_i = GMF(\mathcal{T}^\mathcal{G}, M[i], \tau)$ *and* $s = |\{(c_i, c_j)|c_i \; is \; adjacent \; to \; c_j\}|$*, while preserving privacy. Here,* $\mathcal{T}^\mathcal{G}$ *is the set of generalized trajectories related to the* k *vehicles* V *in the time interval* τ *and* M *is the vector of moves defined on the set of cells* C*.*

4.3 Approach Overview

In this paper, we propose different privacy-preserving solutions based on differential privacy, which is a strong privacy model independent on the background knowledge of an adversary. Each of our solutions is characterized by different trade-off between privacy and data utility. In the following, we describe the key ideas of these three solutions, including the computation by each node and by the coordinator respectively. The node computation mainly involves transforming data to achieve desired privacy guarantee. We present three privacy-preserving data transformation approaches. The first one, named $UniversalNoise$, is based on the classical ϵ-differential privacy. It can provide strong privacy guarantee but high loss of data utility, due to the generation of negative flows and noise of very high magnitude. These two issues are managed in the second solution, named $BoundedNoise$, by relaxing the privacy guarantee to (ϵ, δ)-differential privacy, where δ measures the privacy loss. We will show that: (1) the $BoundedNoise$ approach can improve data utility significantly, and (2) in some cases, the $BoundedNoise$ approach may provide low level of guaranteed privacy in practice. Indeed we can show that sometimes the privacy loss can be high. As a consequence, we propose a third solution named $BalancedNoise$ that tries to maintain the balance between privacy and utility under control by setting appropriate values of ϵ and δ. The mechanism allows the nodes to specify the level of privacy ϵ and the maximum privacy loss δ and find the best solution that is capable to minimize the noise magnitude and the possible negative flows, so that it can achieve good utility. Besides the design of the privacy-preserving data transformation methods, we also design sketch approaches to reduce the communication between nodes and the coordinator. We will validate our theoretical analysis with an extensive set of experiments on large, real data.

5 Privacy-Aware Node Computation

We assume that each node represents a vehicle that moves in a specific territory. Each vehicle in a given time interval observes sequences of spatio-temporal points (trajectories) and computes the corresponding frequency vector that is to be sent to the coordinator. The node computation is composed of two main steps described in Algorithm 1: (a) the computation of a privacy-preserving frequency vector and (b) the vector sketching that compresses the information to be communicated with the coordinator. The first step, described in detail in Algorithm 2, is the challenging step because it has to transform data to achieve privacy without destroying too much the data utility. It is composed of three phases: (1) trajectory generalization; (2) frequency vector construction; and (3) frequency vector transformation for achieving differential privacy. We describe the details of these three phases in Section 5.1 - 5.3 respectively, and discuss the details of the vector sketching step in Section 5.4.

5.1 Trajectory Generalization

Given a specific division of the territory, a trajectory is generalized in the following way. We apply place-based division of the trajectory into segments. The area c_1 con-

Algorithm 1 NODECOMPUTATION(ϵ, τ, M, T^G, w, d)

Inputs: A privacy budget ϵ, a time interval τ, the vector of the moves M, a set of trajectories T^G, the sketch w of dimension d.
Output: The sketch vector representing the privacy-preserving frequency vector $sk(\tilde{f}^{V_j})$.
//Privacy-Preserving Computation (Sec. 5.1-5.3)
$\tilde{f}^{V_j} = PrivacyTransformation(\epsilon, M, T^G, \tau)$
//Data Compression (Sec. 5.4)
$sk(\tilde{f}^{V_j}) = Count(\tilde{f}^{V_j}, w, d)$
return $sk(\tilde{f}^{V_j})$

Algorithm 2 PRIVACYTRANSFORMATION(ϵ, M, T^G, τ)

Inputs: A privacy budget ϵ, a time interval τ, the vector of the moves M, a set of trajectories T^G.
Output: The privacy-preserving frequency vector $\tilde{f}^{V_j}$.
for all observed trajectory $T \in T^G$ **do**
  //Trajectory Generalization (Sec. 5.1)
  $T_g = TrajectoryGeneralization(M, T)$
  //Update of the Frequency Vector f^{V_j} (Sec. 5.2)
  for all move $(l_{c_i}, l_{c_j}) \in T_g$ **do**
    $n = MF(T_g, (l_{c_i}, l_{c_j}), \tau)$
    $f^{V_j}[(l_{c_i}, l_{c_j})] += n$
//Transformation for achieving DP (Sec. 5.3)
$\tilde{f}^{V_j} = AchievingDP(f^{V_j}, \epsilon, T^G)$
return $\tilde{f}^{V_j}$

taining its first point l_1 is found. Then, the second and following points of the trajectory are checked for being inside c_1 until finding a point l_i not contained in c_1. For this point l_i, the containing area c_2 is found.

The trajectory segment from the first point to the i-th point is represented by the vector (c_1, c_2). Then, the procedure is repeated: the points starting from l_{i+1} are checked for containment in c_2 until finding a point l_k outside c_2, the area c_3 containing l_k is found, and so forth up to the last point of the trajectory.

In the result, the trajectory is represented by the sequence of moves $(c_1, c_2, t_1, t_2)(c_2, c_3, t_2, t_3)$ $\ldots (c_{m-1}, c_m, t_{m-1}, t_m)$. Here, in a specific quadruple t_i is the time moment of the last position in c_i and t_j is the time moment of the last position in c_j. There may be also cases when all points of a trajectory are contained in one and the same area c_1. Then, the whole trajectory is represented by the sequence $\{c_1\}$. Since, globally we want to compute aggregation of moves we discard this kind of trajectories. Moreover, as most of the methods for analysis of trajectories are suited to work with positions specified as points, the areas $\{c_1, c_2, \ldots, c_m\}$ are replaced, for practical purposes, by the sequence $l_{c_1}, l_{c_2}, \ldots, l_{c_m}$ consisting of the centroids of the areas $\{c_1, c_2, \ldots, c_m\}$.

5.2 Frequency Vector Construction

After the generalization of a trajectory, the node computes the *Move Frequency* function for each move (l_{c_i}, l_{c_j}) in that trajectory and updates its frequency vector f^{V_j} associated to the current time interval τ. Intuitively, the vehicle populates the frequency vector f^{V_j} according to the generalized trajectory observed. Therefore, at the end of the time interval τ, the element $f^{V_j}[i]$ contains the number of times that the vehicle V_j moved from m to n in the given time interval τ, if $M[i] = (m, n)$.

5.3 Privacy-preserving Vector Transformation

As we stated in Section 4.2, if a node sends the original frequency vector without any data transformation to the coordinator, the intruder may still be able to infer the sensitive typical movements of the vehicle represented by the node. Clearly, the generalization step can help protect the privacy of drivers but it depends on the density of the area. Specifically, if the area is not so dense, the attacker could identify few candidates of locations that the driver has been to. In this case, the privacy is at high risk to be bleached. An attacker could also infer if during a trip a user went from a location a to a location b and how many times. The questions are, *how can we hide the event that the user moved from a location a to a location b during a trip in the time interval τ? And how can we hide the real count of a move in that time window?* To answer these questions, we propose three solutions based on a rigorous privacy model named ϵ-differential privacy (Section 3.2). Each solution provides a different balance between privacy and data utility. The key point of this model is the definition of the sensitivity. We argue that the sensitivity of a move frequency count depends on the occurrence of that move in a trajectory. In a time interval τ for a given vehicle (node) we can have different trips or trajectories. We have a trajectory when the user starts from a location and stops at another. Recall that in our setting each trajectory is transformed into a generalized one and a vehicle can go from cell a to cell b more than once during a trajectory. Therefore, the frequency count of each move can be any arbitrary non-negative integer number. We recall that the frequency count of move (l_{c_a}, l_{c_b}) by node n_j is equal to

$$f = \sum_{\forall T_{g_i}} MF(T_{g_i}, (l_{c_a}, l_{c_b}), \tau),$$

where T_{g_i} is one of the generalized trajectories of n_j in the time interval τ, l_{c_a} and l_{c_b} denote the pair of spatial coordinates representing the centroids of the cells that l_a and l_b that locate in respectively. We argue that adding or deleting one trajectory of n_j can affect the count of move (l_a, l_b) by at most $max_{i=1,\ldots,r}(MF(T_{g_i}, (l_{c_a}, l_{c_b}), \tau))$. Therefore, the sensitivity

$$\Delta f = max_{i=1,\ldots,r}(MF(T_{g_i}, (l_{c_a}, l_{c_b}), \tau)). \quad (1)$$

Note that the frequency count f of move (l_{c_a}, l_{c_b}) always satisfies that $f \geq \Delta f$, as $f = \sum_{i=1,\ldots,r}(MF(T_{g_i}, (l_{c_a}, l_{c_b}), \tau))$. Given the sensitivity of the count of a move we can define a differential private mechanism in various ways. In the following, we present three solutions, each one corresponding to a different implementation of the *AchievingDP* function in Algorithm 2.

5.3.1 $UniversalNoise$ Approach

Our first approach, named $UniversalNoise$, is based on the classic ϵ-differential privacy model. In particular, at the end of the time interval τ, before sending the frequency vector to the coordinator, each node adds the Laplace noise $Lap(\frac{\Delta f}{\epsilon})$, where Δf is defined in Equation 1, to each element in the frequency vector the value in that position of the vector. At the end of this step the node transforms f^{V_j} into $\tilde{f}^{V_j}$. This process is described in Algorithm 3.

Algorithm 3 $UniversalNoise(f^{V_j}, \epsilon, T^G,)$

1: Inputs: A frequency vector f^{V_j}, a privacy budget ϵ, a set of trajectories T^G.
2: Output: The privacy-preserving frequency vector $\tilde{f}^{V_j}$.
3: **for all** vector element $f^{V_j}[k]$ **do**
4: $\Delta f = max_{\forall T_{gi} \in T^G}(MF(T_{gi}, M[k], \tau))$, where M is the vector of moves of f^{V_j}.
5: $noise = Lap(\frac{\Delta f}{\epsilon})$
6: $\tilde{f}^{V_j}[k] = f^{V_j}[k] + noise$
7: **return** $\tilde{f}^{V_j}$

Privacy Analysis. We are ready to show that Algorithm 2 with the privacy transformation presented just now satisfies ϵ-differential privacy.

Theorem 5.1 *Given the total privacy budget ϵ, for each frequency value x, $UniversalNoise$ approach ensures ϵ-differential privacy.*

The correctness of Theorem 5.1 is straightforward due to how the noise is added according to the Laplace mechanism [16].

5.3.2 $BoundedNoise$ Approach

The $UniversalNoise$ approach has a few weakness. First, it could lead to large amounts of noise, due to the fact that Δf, although with small probability, can be arbitrarily large. Second, adding noise drawn from the Laplace distribution could generate negative frequency counts of moves, which does not make sense in our setting. To fix these two problems, we propose the second approach, named $BoundedNoise$ approach, that bounds the noise drawn from the Laplace distribution. In particular, for each value x of the vector f^{V_j}, we draw the noise from $Lap(\frac{\Delta f}{\epsilon})$ bounded to the interval $[-x, x]$. In other words, for any original frequency $f^{V_j}[i] = x$, its perturbed version after adding noise should be in the interval $[0, 2x]$. By doing this, we reduce the amounts of utility loss due to adding noise. Algorithm 4 describes the $BoundedNoise$ approach.

We are aware that using a truncated version of the Laplace distribution may lead to privacy leakage. In the following we show that the $BoundedNoise$ approach satisfies (ϵ, δ)-differential privacy, where δ measures the privacy loss.

Privacy Analysis. As pointed out in [22], differential privacy must be applied with caution. The privacy protection provided by differential privacy relates to the data

Algorithm 4 $BoundedNoise(f^{V_j}, \epsilon, T^G)$

Inputs: A frequency vector f^{V_j}, a privacy budget ϵ, a set of trajectories T^G.
Output: The privacy-preserving frequency vector $\tilde{f}^{V_j}$.
for all vector element $f^{V_j}[k]$ **do**
    $\Delta f = max_{\forall T_{gi} \in T^G}(MF(T_{gi}, M[k], \tau))$, where M is the vector of moves of f^{V_j}.
    $noise = Lap(\frac{\Delta f}{\epsilon})$
    while $(noise > f^{V_j}[k])$ *or* $(noise < -f^{V_j}[k])$ **do**
        $noise = Lap(\frac{\Delta f}{\epsilon})$
    $\tilde{f}^{V_j}[k] = f^{V_j}[k] + noise$
return $\tilde{f}^{V_j}$

generating mechanism and deterministic aggregate level background knowledge. Let F and F' be the frequency distribution before and after adding Laplace noise. We observe that bounding the Laplace noise will lead to some privacy leakage on some values. For instance, from the noisy frequency values that are large, the attacker can infer that these values should not be transformed from small ones. To analyze the privacy leakage of our bound-noise approach, we first explain the concept of *statistical distance* [3]. Formally, given two distributions X and Y, the *statistical distance* between X and Y over a set U is defined as $d(X, Y) = max_{S \in U}(Pr[X \in S] - Pr[Y \in S])$.

[3] also shows the relationship between (ϵ, δ)-differential privacy and the statistical distance.

Lemma 5.1 *[3] Given two probabilistic functions F and G with the same input domain, where F is (ϵ, δ_1)-differentially private. If for all possible inputs x we have that the statistical distance on the output distributions of F and G is: $d(F(x), G(x)) \leq \delta_2$, then G is $(\epsilon, \delta_1 + (e^\epsilon + 1)\delta_2)$-differentially private.*

Let F and F' be the frequency distribution before and after adding Laplace noise. We can show that the statistical distance between F and F' can be bounded as follows:

Lemma 5.2 *[3] Given an (ϵ, δ)-differentially private function F with $F(x) = f(x) + R$ for a deterministic function f and a random variable R. Then for all x, the statistical distance between F and its throughput-respecting variant F' with the bound b on R is at most $d(F(x) - F'(x)) \leq Pr[|R| > b]$.*

[3] has the following lemma to bound the probability $Pr[|R| > b]$.

Lemma 5.3 *[3] Given a function F with $F(x) = f(x) + Lap(\frac{\Delta f}{\epsilon})$ for a deterministic function f, the probability that the Laplacian noise $Lap(\frac{\Delta f}{\epsilon})$ applied to f is larger than b is bounded by: $Pr(|Lap(\frac{\Delta f}{\epsilon})| > b) \leq \frac{2(\Delta f)^2}{b^2 \epsilon^2}$.*

We stress that this upper bound is not tight. For instance, when $\Delta f = 1$, $b = 1$, and $\epsilon = 1$, the bound $\frac{2(\Delta f)^2}{x^2 \epsilon^2} = 2$. Therefore, we improve the bound by the following theorem.

Lemma 5.4 *Given a function F with $F(x) = f(x) + Lap(\frac{\Delta f}{\epsilon})$ for a deterministic function f, the probability that the Laplacian noise $Lap(\frac{\Delta f}{\epsilon})$ applied to f is larger than b is bounded by:*

$$Pr(|Lap(\frac{\Delta f}{\epsilon})| > b) \leq e^{\frac{-b\epsilon}{\Delta f}}.$$

Proof. *Let $\lambda = \frac{\Delta f}{\epsilon}$. The probability density function is $p(x) = \frac{1}{2\lambda}e^{(-|x|/\lambda)}$ and the cumulative distribution function is*

$$D(x) = (1/2)(1 + sgn(x)(1 - exp(|x|/\lambda))).$$

Therefore,

$$\begin{aligned} Pr(Lap(\frac{\Delta f}{\epsilon}) > b) &= \int_b^{\infty} \frac{1}{2\lambda} e^{(-|x|/\lambda)}\,\mathrm{d}x \\ &= \frac{1}{2\lambda}(\int_0^{\infty} e^{(-|x|/\lambda)}\,\mathrm{d}x - \int_0^{b} e^{(-|x|/\lambda)}\,\mathrm{d}x) \\ &= D(\infty) - D(b) \\ &= e^{(-b/\lambda)}. \end{aligned} \tag{2}$$

Our analysis shows that $e^{\frac{-b\epsilon}{\Delta f}} \leq \frac{2(\Delta f)^2}{b^2\epsilon^2}$, i.e., our bound is tighter than that in [3]. We stress that in our approach, the bound b of each frequency value x is not fixed. Indeed, $b = x$. Therefore, each frequency value x has different amounts of privacy leakage. Our approach thus achieves different degree of (ϵ, δ)-differentially privacy guarantee on each frequency value x. Theorem 5.2 shows more details.

Theorem 5.2 *Given the privacy budget ϵ, for each frequency value x, $BoundedNoise$ approach ensures $(\epsilon, (e^{\epsilon}+1)e^{\frac{-x\epsilon}{\Delta f}})$-differentially privacy, where Δf is defined in Equation 1.*

The correctness of Theorem 5.2 can be easily proven by Lemma 5.1 and Lemma 5.4. Note that the frequency vectors with Laplace noise (without truncation) satisfies $(\epsilon, 0)$-differentially privacy. It is easy to verify that the privacy loss, measured as $\delta = (e^{\epsilon}+1)e^{\frac{-x\epsilon}{\Delta f}}$, can be high. More details are as following. Recall that for any frequency count x, $x \geq \Delta f$ always holds. Next we discuss by cases that $x = \Delta f$ and $x > \Delta f$. For the former case that $x = \Delta f$, $\delta = (1 + e^{-\epsilon}) > 1$, i.e., the privacy loss is always grater than 1. For the latter case that $x > \Delta f$, $\delta = e^{(1-\frac{x}{\Delta f})\epsilon} + e^{\frac{-x\epsilon}{\Delta f}}$. In this case, $\delta > 1$ holds when $x < \frac{ln(e^{\epsilon}+1)\Delta f}{\epsilon}$. In other words, smaller frequency counts have higher probability to get larger amounts of privacy loss.

5.3.3 $BalancedNoise$ Approach

As discussed above, the $UniversalNoise$ approach may provide very strong privacy guarantee but poor data utility, while the $BoundedNoise$ approach can improve data utility but with possible high privacy loss. Our third approach, named $BalancedNoise$,

tries to address the trade-off issue between privacy and data utility. This approach, described in Algorithm 5, allows the user to set the desirable values for the two parameters, the privacy budget threshold ϵ and the privacy loss threshold δ. Next, for each value x of the vector f^{V_j}, the algorithm finds the smallest interval $[-b, b]$ such that the following inequality holds: $(e^{\epsilon} + 1)e^{\frac{-b\epsilon}{\Delta f}} \leq \delta$. Note that $e^{\frac{-b\epsilon}{\Delta f}}$ is the privacy loss we found in Lemma 5.4. This implies that $b \geq \frac{-\Delta f}{\epsilon} ln \frac{\delta}{e^{\epsilon}+1}$.

After finding the interval, for each value x of the frequency vector, the node draws the noise from $Lap(\frac{\Delta f}{\epsilon})$ bounding the noise value to the interval $[-b, b]$, where $b = max(0, \frac{-\Delta f}{\epsilon} ln \frac{\delta}{e^{\epsilon}+1})$. Note that this solution limits as much as possible the generation of noise with values of too high magnitude while not completely solves the problem of the negative flows. Clearly, the possibility to compute the minimum interval that better fits the user privacy requirements also helps to limit the negative flows. Similar to the $BoundedNoise$ approach, the $BalancedNoise$ approach satisfies (ϵ, δ)-differential privacy.

Algorithm 5 $BalancedNoise(f^{V_j}, \epsilon, T^G, \delta)$

1: Inputs: A frequency vector f^{V_j}, a privacy budget ϵ, the privacy loss δ, a set of trajectories T^G.
2: Output: The privacy-preserving frequency vector $\tilde{f}^{V_j}$.
3: Compute $b = \frac{-\Delta f}{\epsilon} ln \frac{\delta}{e^{\epsilon}+1}$.
4: **for all** vector element $f^{V_j}[k]$ **do**
5: $\Delta f = max_{\forall T_{gi} \in T^G}(MF(T_{gi}, M[k], \tau))$, where M is the vector of moves of f^{V_j}.
6: $noise = Lap(\frac{\Delta f}{\epsilon})$
7: **while** $(noise > b)$ *or* $(noise < -b)$ **do**
8: $noise = Lap(\frac{\Delta f}{\epsilon})$
9: $\tilde{f}^{V_j}[k] = f^{V_j}[k] + noise$
10: **return** $\tilde{f}^{V_j}$

5.4 Compact Communications

In a distributed system an important issue to be considered is the amount of data to be communicated. In fact, real life systems usually involve thousands vehicles (nodes) that are located in any place of the territory. Each vehicle has to send to the coordinator the information contained in its frequency vector that has a size depending on the number of cells that represent the partitions of the territory. The number of cells in a territory can be very huge and this can make large frequency vectors. As an example, in the dataset of real-life trajectories used in our experiments, there are approximately $4,200$ vehicles and we use a territory tessellation of about $2,400$ cells. So, considering as possible moves only pairs of adjacent cells we obtain frequency vectors containing about $15,900$ positions (moves). Therefore, the system has to be able to handle not only a very large number of nodes but also huge amounts of the information to be communicated. These considerations make the optimization of communicated information

necessary. To address this problem it is possible to compress the transmitted data by sketching algorithms [10]. Here, we propose the application of *Count* sketch algorithm [13]. This algorithm maps a frequency vector f onto a more compressed vector. The sketch consists of an array C of $d \times w$ counters. For each of d rows, there are two hash functions: h_j that maps items, that our case are moves, onto one of the elements of the $j-th$ row, and g_j that maps each item onto $\{-1, 1\}$. For each item i, it will be mapped onto d entries in the array by adding the value $f[i] \times g_j(i)$ on the entry $C[j, h_j(i)]$ in row j, for $1 \leq j \leq d$.

Given a sketch representation of a vector we can estimate the original value of each component of the vector by the following function $\hat{f}[i] = median_{1 \leq j \leq d} g_j(i) C[j, h_j(i)]$. Setting $d = \log \frac{1}{\gamma}$ and $w = log(\frac{4}{\alpha^2})$ this sketch ensures that the estimation of $f[i]$ has error at most αn with probability at least $1 - \gamma$. Therefore, here α indicates the accuracy (i.e. the approximation error), and γ represents the probability of exceeding the accuracy bounds.

In [27] we proposed the application of *Count-Min* sketch algorithm; it is suitable for compressing non-negative values while does not work well in case of presence of negative values. Since with $UniversalNoise$ and $BalancedNoise$ we can obtain negative flows then we choose the *Count* sketch algorithm that is not sensible to negative values.

Adding this data summarization step (the last step in Algorithm 1) does not change the privacy guarantee provided by the above methods. This is due to the fact that the *Count* sketching function only accesses a differentially private frequency vector, not the underlying database. As proven by Hay et al. [21], a post-processing of differentially private results remains differentially private. Therefore, also the whole Algorithm 1 with the sketching step maintains the same privacy guarantee of Algorithm 2.

6 Coordinator Computation

The computation of the coordinator is composed of two main phases: 1) computation of the set of moves and 2) computation of the aggregation of global movements.

Move Vector Computation. The coordinator in an initial setup phase has to send to the nodes the *vector of moves* (Definition 3.4). The computation of this vector depends on the set of cells that represent the partition of the territory. This partition can be a simple grid or a more sophisticated territory subdivision such as Voronoi tessellation. The sharing of vector of moves is a requirement of the whole process because each node has to use the same data structure for allowing the coordinator the correct computation of the global flows.

Global Flow Computation. The coordinator has to compute the global vector that corresponds to the global aggregation of movement data in a given time interval τ by composing all the local frequency vectors. It receives the sketch vector $sk(\tilde{f}^{V_j})$ from each node; then it reconstructs each frequency vector from the sketch vector, by using the estimation described in Section 5.4. Finally, the coordinator computes the global frequency vector by summing the estimate vectors component by component. Clearly the estimate global vector is an approximated version of the global vector obtained by summing the local frequency vectors after the only privacy transformation.

7 Experiments

7.1 Dataset

For our experiments we used a large dataset of GPS vehicles traces, collected in a period from 1st May to 31st May 2011. The GPS traces were collected in the geographical areas around Pisa, in central Italy, and it counts for around $4,200$ vehicles, generating around $15,700$ trips. In our simulation, the coordinator collects the Frequency Vectors (FV) from all the vehicles to determine the Global Frequency Vector (GFV), i.e. the sum of all the trajectories crossing any link, at the end of each day, so we defined a series of time intervals τ_i, where each τ_i spans over a single day. In the following we show the resulting GFV for the 25th May 2011, but similar accuracy is observed also for the other days. Note that we conducted experiments on data by considering different sizes of time interval τ: 4 hours, one day and 2 days. The results we found in terms of data utility are very similar, therefore for lack of space, in the following we only report the results concerning τ equal to *one day*.

7.2 Space Tessellation

The generalization and aggregation of movement data is based on space partitioning. Arbitrary territory divisions, such as administrative districts or regular grids, do not reflect the spatial distribution of the data. The resulting aggregations may not convey the essential spatial and quantitative properties of the traffic flows over the territory. Our method for territory partitioning extends the data-driven method suggested in paper [2]. Using a given sample of points (which may be, for example, randomly selected from a historical set of movement data), the original method finds spatial clusters of points that can be enclosed by circles with a user-chosen radius. The centroids of the clusters are then taken as generating seeds for Voronoi tessellation of the territory. We have modified the method so that dense point clusters can be subdivided into smaller clusters, so that the sizes of the resulting Voronoi polygons vary depending on the point density: large polygons in data-sparse areas and small polygons in data-dense areas. The method requires the user to set 3 parameters: maximal radius R, minimal radius r, and minimal number of points N allowing a cluster to be subdivided. In our experiments, we used a tessellation with $2,681$ polygons obtained with $R = 10km$, $r = 500m$, $N = 80$.

7.3 Utility Measures

To assess the information loss incurred to achieve privacy and to reduce the amount of information to be transmitted, we study how much data utility is preserved after the transformations. Since the coordinator reconstructs the flows among the zones of the tessellation, we can represent such data as a directed graph, where the nodes represent the zones and an edge between two nodes represents the flows from one zone to the other. This graph-based model allows us to analytically evaluate the resulting aggregations by means of some network-based statistics, described below. The models can also be exploited for different application scenarios and for each of them we can

assess the quality of results after the transformations, since these mobility analyses can be performed on the transformed data too.

Network-based Measures. In order to assess the utility of the data collected by the coordinator we study how the distributions of general network-based measures are preserved. In particular we have considered the following measures:

Flow per Link: this measure evaluates the volume of flow in each move (edge), i.e., traffic between two adjacent zones.

Flow per Zone: for each zone we sum the flows of all the incoming and outgoing flows in a zone (node).

Node Degree: this measures is similar to the previous one, but there we consider the distinct number of origins and destinations for each zone, thus focusing on the topological properties of the resulting graph.

Clustering Coefficient [30]:given a node the clustering coefficient is defined as the probability that two randomly selected neighbors are connected to each other.

Node Betweeness: this function is a measure of a node's centrality in a network. It computes the number of shortest paths from all nodes to all others that pass through that node. This measure highlights the load placed on the given node in the network as well as the node's importance to the network than just connectivity.

Edge Betweeness: this function provides similar information to the previous one but considering the edge instead of the node. In other words, it measures the edge's centrality in a network.

Mobility Application Scenarios. The reconstructed GVF enables a traffic manager to evaluate the traffic condition by monitoring the status of the road network. We explored a visualization approach where the measures *Flow per link* and *Flow per zone* are rendered on a map. In particular, in Figure 1 the flows per link are presented as arrows whose thickness is proportional to the amount of traffic on that link. The flow per zone is rendered with a circle whose radius is proportional to the median value of all the zones and the color indicates if the flow is above (red) or below (cyan) the median. These two graphical representation allows to identify easily the portions of the road network with critical traffic conditions. Figure 1 shows the reconstructed map for different parameters for privacy preservation. This allows us to qualitatively choose a good trade-off between data privatization and data utility. For example, for low values of epsilon, e.g. $\epsilon = 0.2$, it is still possible to reason about traffic condition since the mayor flows for links are sufficiently preserved. The cumulative flows per zone are more robust to data transformation, since the randomization is performed on the edge level and, hence, in the same zone different perturbations on incident edges tend to compensate each others. From the figure we can notice the influence of the δ parameter on the transformed flows. In fact, fixed a value $\epsilon = 0.2$ the overall quality of the maps can be improved by increasing the second parameter for the $BalancedNoise$ transformation. In particular, it is evident how the resulting maps for $\delta = 0.1$ and $\delta = 0.2$ present a topology similar to the original flows. The strong relationship between ϵ and δ will be analytically discussed in Section 7.4.

The transformed data has also been used to study the aggregation of zone on the basis of their relative mobility, according to the approach presented in [32]. Starting from the graph-based model of flows, we apply a community discovery algorithm on the data to determine the groups of nodes strongly connected by high flows. We call such aggregation of zones as *Mobility Borders* to stress the definition of a boundary derived from mobility data. The result of *Mobility Borders* can be rendered visually by joining the geometries of the zones into a larger polygon according to the group they belong to. Figure 2 shows a visual comparison between the resulting aggregations for different combination of ϵ and δ and the aggregation resulting from the original data. The borders yielding from the original data are rendered as thicker lines to facilitate the comparison. The resulting borders for the transformed data are rendered by colors: zone in the same group are filled with the same color. The map shows the influence of the two parameters for the transformation, in particular we show the resulting maps for $\epsilon = 0.2$ and $\delta = 0.2$. We can notice how *Mobility Borders* results are very robust to data perturbation, since the majority of the zones are preserved even for low values of ϵ. However, it is possible to identify small variation on central zones of the map where we have an higher density of links and connections.

In general, the zones grouped for the original data tend to stay in the same group also for the transformed data. In some cases, it happens that an original group is split across two or three distinct new groups. To analytically evaluate such behavior, we consider two measures adapted from information retrieval research field: *precision* and *recall*. With precision we measure the ratio of zones in the same group in the original data that stay in the same group in the transformed one. The recall measures the contribution of several original groups to a group coming from transformed data. The resulting values for the two measures are showed in Figure 3. We can see that the precision (Figure 3(left)) remains very high for any value of ϵ for the motivations discussed above. Recall (Figure 3(right)), instead, tends to decrease for $\epsilon < 0.3$ and the overall result is increased by augmenting δ to 0.2 or 0.3.

We also studied the impact of the parameters of the privacy transformation in more detail by analyzing the spatial distributions of the errors, expressed as the logarithms of the ratios of the aggregated traffic values obtained from transformed data to those obtained from the original data. The use of the logarithms allowed us to reduce the impact of local outliers. The study was done using the results of 99 runs for all combinations of the values of ϵ from 0.1 to 0.9 with the step 0.1 and the values of delta $0.01, 0.02, 0.025, 0.03, 0.05$, and up to the value 0.2 with the step 0.025. The corresponding 99 spatial distributions of the errors were clustered by similarity using the k-means methods. We experimented with different k and found that, starting from $k = 9$, increasing the value of k just subdivides small clusters into yet smaller ones, mostly singletons. There is one large cluster (Cluster 7) consisting of 68 distributions that preserves when k increases. This cluster consists of the spatial distributions with the best (i.e., lowest) values of the errors. The area-wise median errors for this cluster are shown in the map in Figure 4 (left) by color-coding. Light yellow corresponds to values close to 0, shades of orange and red represent overestimates and shades of blue underestimates. The color legend is shown on the right of Figure 4. The prevalence of light yellow and light shades of orange means that the absolute values of the errors in cluster 7 are quite low. There are only a few high overestimates occurring in areas

with low traffic density. Cluster 7 includes all spatial distributions for values of $\epsilon = 0.4$ and higher and values of delta from 0.01 to 0.05 and almost all spatial distributions for epsilon 0.6 and higher irrespective of the value of δ. Hence, starting from $\epsilon = 0.6$, δ has no impact on the data quality. For comparison, the map on the right of Figure 4 represents the errors in another cluster, which includes the spatial situations for $\epsilon = 0.2$. Very high overestimates occur almost everywhere. For $\epsilon = 0.1$, the overestimates are even higher. This study clarifies what combinations of the parameter values should be used for obtaining good results in terms of utility of the transformed data.

7.4 Analytical evaluation

We now discuss the experiments conducted on the real-world data described above. To evaluate the data quality after the transformation we compare the transformed flows with the original one. According to the utility measures defined in Section 7.3, for each measure we compare the resulting statistics for each node and edge of the graph-model resulting from transformed data with the graph yielding from the original data. For example, in Figure 5 we show the comparison of the *Flow per link* measure for two different transformations, namely $\epsilon = 0.5$ (left) and $\epsilon = 0.2$ (right). The scatter plots highlight the differences between the two transformations, where the more protective transformation ($\epsilon = 0.2$) perturbs more the data, since the data points tend to go far from the fitting line.

To present the results for different comparisons of parameters and utility measures, we adopt the Pearson Correlation Coefficient (PCC) to represent analytically the amount of data perturbation introduced. The coefficient tends to 1 when the data points are close to the regression line, while it tends to zero when the data point are scattered away from the line. In the following we will consider only two methods: $UniversalNoise$ and $BalancedNoise$, since we found in our experiments that the $BoundedNoise$ may presents privacy loss. Indeed, we observed that usually in a time interval of one day each user has a high set of moves with low value, because typical users during the day go from an area to another only few times. This fact implies that the application of the $BoundedNoise$ method may lead to a too high privacy loss. In Figure 7 (top,right) we plot the percentage of cases where we have a resulting δ too high to be acceptable for privacy protection. In Figure 7 (bottom, right), we also noted that when we increase the time interval τ the privacy loss decreases and this supports our hypothesis that this naive approach can give good trade-off between privacy and data utility in scenarios where it is reasonable to have a wide time window, for example *one week*, and in contexts which are characterized by high frequencies of items. However, the utility provided by $BoundedNoise$ method is very good, as showed in Figure 7 (left), where it is reported the PCC for each network-based measure computed for different values of ϵ.

To assess the validity of the transformation approach, we compare the private data with the original data by varying the transformation parameters. The comparison is performed with two approaches by varying the values of ϵ and δ: we compare the resulting cumulative distribution of the utility measures and the linear correlations by means of the PCC. In Figure 6 we report, for each utility measure, the resulting distributions for $\epsilon = 0.1, 0.2, \ldots, 0.9$ and for the original data. From such plots it is possible

to estimate the best parameters that yield a good trade-off between data protection and data utility. For example, for the *Flow per link* measure (Figure 6(a)), we can notice a clear discontinuity for $\epsilon = 0.2$ and $\epsilon = 0.1$, suggesting that a good value for ϵ would be 0.3. However it is interesting to note how the δ parameter may contribute to increase data utility. In fact, considering a more protective value for ϵ, say $\epsilon = 0.2$, it is possible increasing δ to augment the resulting data utility. In Figure 6(b), we can see how the distributions tend to be similar to the curve for $\epsilon = 0.3$ when we increase δ. In particular, when $\delta = 0.2$ the curve is very similar to $\epsilon = 0.3$ even with a difference on the tails of the two curves. Similar results can be observed for *Flow per Zone* measure (Figure 6(c) and (d)), where the candidate value for ϵ is again 0.3. Also in this case, the δ parameter contributes to enhance the data protection by lowering the value for ϵ to 0.2 and increasing δ to 0.2. The *Clustering Coefficient* measure is very robust even for low values of ϵ (Figure 6(e) and (f)): we can appreciate a different distribution only when $\epsilon = 0.1$. This property confirms that the privacy transformation may perturb the local weights of edges but in general it preserves the topology of the graph. Another evidence of this phenomenon is given by the two measures of betweenness ((Figure 6(i), (j), (k) and (l))), where we can appreciate how the different parameters yield similar distribution. This means that, for example, the number of relevant edges within the graph is maintained across different transformations. This is evident also from the distribution of the *Node Degree* measure, where we can notice how the number of neighbors for each node tends to diminish when ϵ becomes smaller. We can relate this property to the pruning of some graph components that, however, are not relevant for the connectivity of the graph.

Besides the general distributions of the utility measures, we want also to assess how locally each component of the graph is transformed. Figure 8 shows, for each utility measure, the resulting PCC for different combination of δ and ϵ. Even at this level of detail, it is possible to identify the most promising ϵ values for the transformations. In particular, let us consider the *Flow per Link* correlation in Figure 8(a). As already observed for the cumulative distribution, the correlation index decreases considerably when ϵ is less than 0.3. Fixed a minimum PCC threshold, we can reasoning about the relation between ϵ and δ. Fixed a minimum value of 0.77 for PCC, we can reach a comparable quality result even if we decrease ϵ by increasing the value of δ. From the figure we can infer that the data utility provided by $\epsilon = 0.3$ is equivalent to the data utility for $\epsilon = 0.22$ and $\delta = 0.2$. Similarly, fixed a value for ϵ, say $\epsilon = 0.3$, by increasing δ it is possible to increase the data quality of the reconstructed flows. The relation between the two parameters enables the data owner to define the most suitable trade-off between data protection and data utility. The discussion for the choice of the correct ϵ parameter is even more crucial for the betweenness quality measures. Figures 8(i) and (k) show that the PCC drops when the threshold is below $\epsilon = 0.3$. However, when δ is increased the quality measure performance raises.

We have also empirically evaluated how the network-based measures are preserved after the sketch transformation. We have considered three different sketch transformations according to the parameter showed in Table 1. We found that in general the quality of network-based measures is reasonable though the approximation introduced by the Count sketches. We show the effect of the Count sketches for different size in Figure 8(b), (d), (f), (h), (j) and (l). We observe how the PCC is particularly well

	α	γ	Columns (w)	Rows (d)	$w \times d$
C_{3k}	0.03162	0.05	1,000	3	3,000
C_{5k}	0.03162	0.01	1,000	5	5,000
C_{10k}	0.03162	0.00005	1,000	10	10,000

Table 1: Count sketch size for different values of α and γ.

preserved for the two measures *Flow per Link* and *Flow per Zone*. The preservation of this measure is important because this enables analyses as those in Figure 1. Clearly, we observe for all measures an decreasing of the correlation when the rate compression increases.

8 Conclusion

In this paper, we have studied the problem of computing movement data aggregation based on trajectory generalization in a distributed system while preserving privacy. We have proposed three methods for protecting privacy based on the well-known notion of differential privacy that provides very nice data protection guarantees. Each solution is characterized by a different trade-off between privacy and data utility. S In particular, in our framework each vehicle, before sending the information about its movements within a time interval, applies to the data a transformation for achieving privacy and then, can creates a summarization of the private data (by using a sketching algorithm) for reducing the amount of information to be transmitted. The results obtained in our experiments show that the privacy transformations preserve some important properties of the original data allowing the analyst to use them for important mobility data analysis. We have validated the robustness and efficiency of our privacy-preserving data aggregation methods by extensive experiments on large, real GPS data.

Future investigations could be directed to explore other methods for achieving differential privacy; as an example, it would be interesting to understand the impact of the use of the geometric mechanism instead of the Laplace one for achieving differential privacy.

References

[1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *ICDE)*, pages 376–385, 2008.

[2] N. Andrienko and G. Andrienko. Spatial generalization and aggregation of massive movement data. *Trans. on Visualization and Computer Graphics*, 17:205-C219, 2011.

[3] M. Backes, S. Meiser. Differentially Private Smart Metering with Battery Recharging. *IACR Cryptology ePrint Archive* 2012: 183 (2012)

[4] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: a holsistic solution to contingency table release. In *PODS*, pages 273–282, 2007.

[5] A. Beimel, K. Nissim, and E. Omri. Distributed private data analysis: Simultaneously solving how and what. In CRYPTO, pages 451–468, 2008.

[6] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta. Discovering frequent patterns in sensitive data. In *KDD*, pages 503–512, 2010.

[7] R. Chen, B.C.M. Fung, B.C. Desai, and N.M. Sossou. Differentially private transit data publication: a case study on the montreal transportation system. In *KDD*, pp. 213–221, 2012.

[8] G.Cormode and S.Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1):58–75, 2005.

[9] G.Cormode and M. N. Garofalakis. Approximate continuous querying over distributed streams. *ACM Trans. Database Syst.*, 33(2), 2008.

[10] G. Cormode, M. N. Garofalakis, P. J. Haas, and C. Jermaine. Synopses for massive data: Samples, histograms, wavelets, sketches. *Foundations and Trends in Databases*, 4(1-3):1–294, 2012.

[11] G. Cormode, C. M. Procopiuc, D. Srivastava, E. Shen, and T. Yu. Differentially private spatial decompositions. In *ICDE*, pages 20–31, 2012.

[12] G. Cormode, C. M. Procopiuc, D. Srivastava, and T. T. L. Tran. Differentially private summaries for sparse data. In *ICDT*, pages 299–311, 2012.

[13] G. Cormode, M. Hadjieleftheriou: Methods for finding frequent items in data streams. *VLDB J.* 19(1): 3-20 (2010)

[14] B. Ding, M. Winslett, J. Han, and Z. Li. Differentially private data cubes: optimizing noise sources and consistency. In *SIGMOD*, pages 217–228, 2011.

[15] C. Dwork. Differential privacy: a survey of results. In *TAMC*, pages 1–19, 2008.

[16] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[17] T-H. H. Chan, E. Shi, and D. Song. Optimal lower bound for differentially private multi-party aggregation. In *ESA*, pages 277-288, 2012.

[18] D. Feldman, A. Fiat, H. Kaplan, and K. Nissim. Private coresets. In *STOC*, pages 361–370, 2009.

[19] A. Friedman and A. Schuster. Data mining with differential privacy. In *KDD*, pages 493–502, 2010.

[20] F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli, C. Renso, S. Rinzivillo, and R. Trasarti. Unveiling the complexity of human mobility by querying and mining massive trajectory data. In *The VLDBJ*, pages 695-719, 2011, Vol. 20, Issue 5.

[21] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. In *VLDB Endow.*, 3(1-2):1021–1032, 2010.

[22] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *SIGMOD*, pages 193–204, 2011.

[23] N. Li, W. H. Qardaji, D. Su, and J. Cao. Privbasis: Frequent itemset mining with differential privacy. *PVLDB*, 5(11):1340–1351, 2012.

[24] F. McSherry and R. Mahajan. Differentially-private network trace analysis. In *SIGCOMM 2010*, pages 123–134, 2010.

[25] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[26] N. Mohammed, R. Chen, B. C.M. Fung, and P. S. Yu. Differentially private data release for data mining. In *KDD*, 2011.

[27] A. Monreale, W. H. Wang, F. Pratesi, S. Rinzivillo, D. Pedreschi, G. Andrienko, and N. Andrienko. Privacy-preserving Distributed Movement Data Aggregation, In *AGILE*, 2013.

[28] A. Monreale, G. L. Andrienko, N. V. Andrienko, F. Giannotti, Dino Pedreschi, Salvatore Rinzivillo, and Stefan Wrobel. Movement data anonymity through generalization. *TDP*, 3(2):91–121, 2010.

[29] M. Nanni, R. Trasarti, G. Rossetti, and D. Pedreschi. Efficient distributed computation of human mobility aggregates through User Mobility Profiles. In *KDD Int. Workshop on Urban Computing*, pages 87 – 94, 2012.

[30] M. Newman. The Structure and Function of Complex Networks. *SIAM Review*, 45:2, pages 167–256, 2003.

[31] V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *SIGMOD*, pages 735–746, 2010.

[32] S. Rinzivillo, S. Mainardi, F. Pezzoni, M. Coscia, D. Pedreschi, F. Giannotti. Discovering the Geographical Borders of Human Mobility. *KI* 26(3): 253-260, 2012.

[33] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppresion. In *IEEE Symp. on Research in Security and Privacy*, pages 384–393, 1998.

[34] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information(abstract). In *PODS*, 1998.

[35] E. Shi, T.-H. H. Chan, E. G. Rieffel, R. Chow and D. Song, Privacy-Preserving Aggregation of Time-Series Data. In *NDSS*, 2011.

[36] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *MDM*, 2008.

[37] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. *TKDE*, 23(8):1200–1214, 2011.

[38] J. Xu, Z. Zhang, X. Xiao, Y. Yang, and G. Yu. Differentially private histogram publication. In *ICDE*, pages 32–43, 2012.

[39] R. Yarovoy, F. Bonchi, L.V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: how to hide a mob in a crowd? In *EDBT*, pages 72–83, 2009.

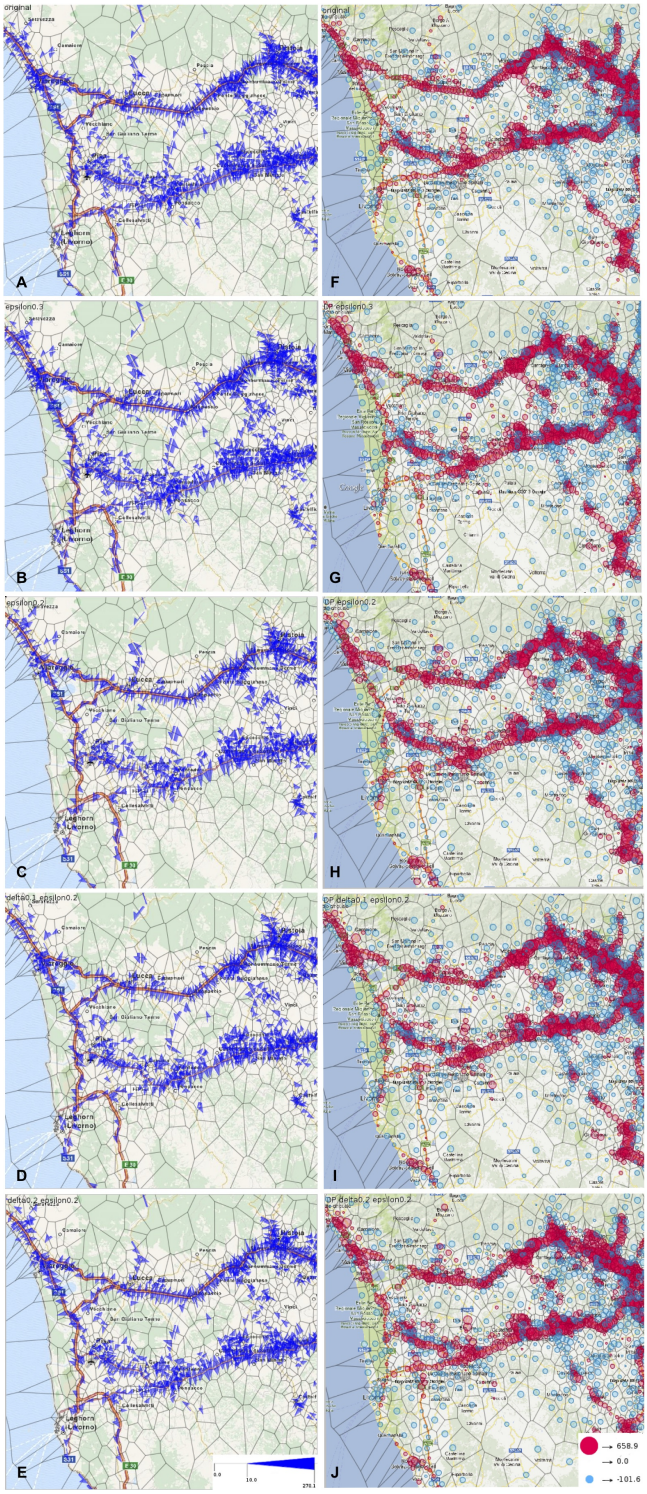


Figure 1: Traffic and density analysis: comparison between $UniversalNoise$ and $BalancedNoise$ approaches ($\epsilon = 0.2$, $\langle \epsilon = 0.2, \delta = 0.1 \rangle$ and $\langle \epsilon = 0.2, \delta = 0.2 \rangle$)

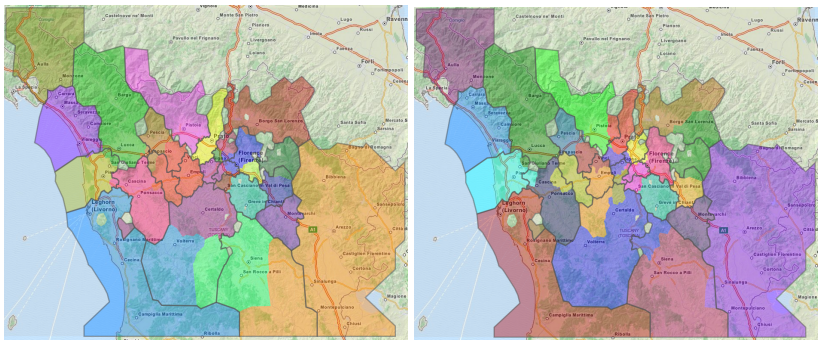


Figure 2: Mobility Borders results for $\epsilon = 0.2$ and $\langle \epsilon = 0.2, \delta = 0.2 \rangle$ compared with results from original data (thicker black lines)

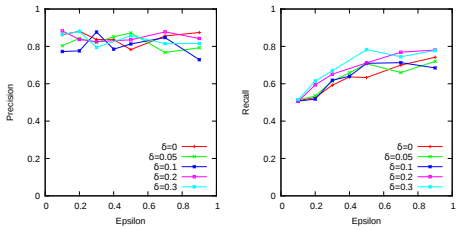


Figure 3: Quantitative measure for borders

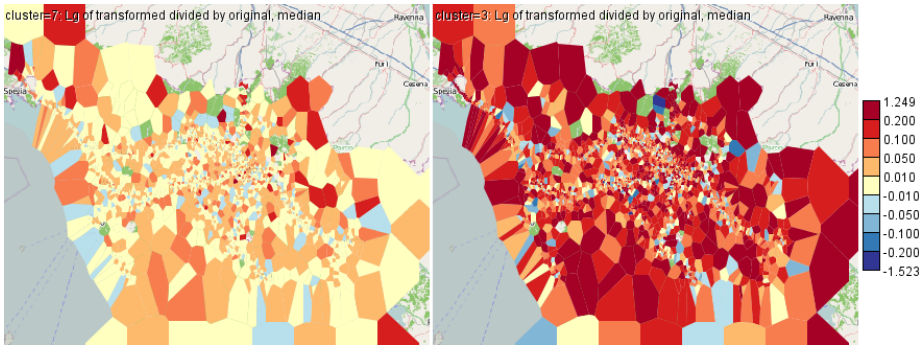


Figure 4: Comparison of the spatial distributions of the errors for different value combinations of ϵ and δ.

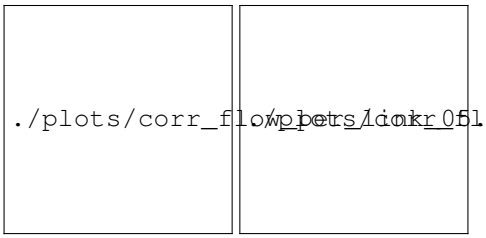


Figure 5: Correlations of flows after $UniversalNoise$

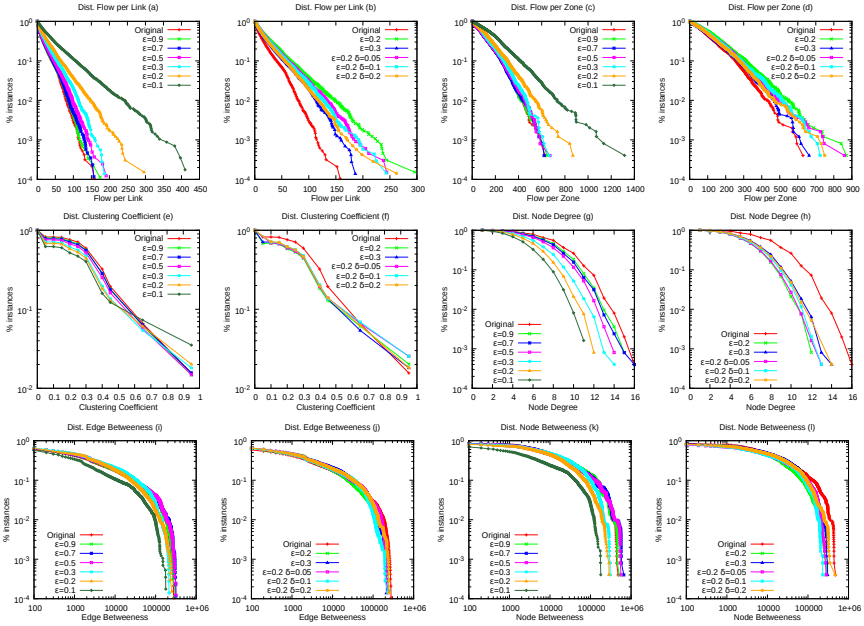


Figure 6: Cumulative Distributions of the network-based measures

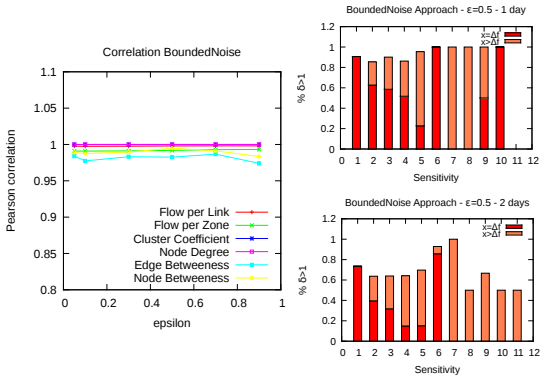


Figure 7: Study of the privacy transformation $BoundedNoise$

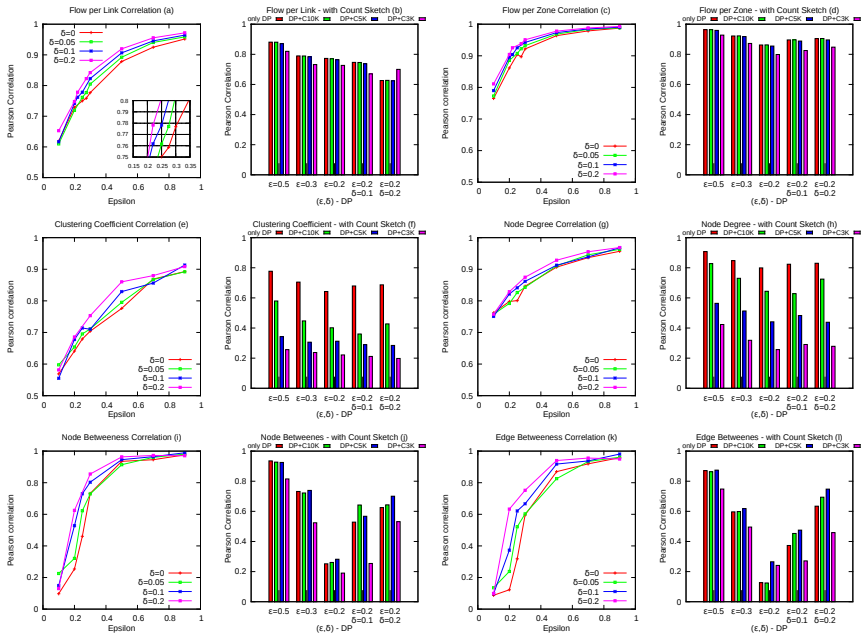


Figure 8: Distribution of the PCC of various network-based measures after the privacy transformation