PRIVACY-PRESERVING TRANSPORTATION TRAFFIC MEASUREMENT IN INTELLIGENT CYBER-PHYSICAL ROAD SYSTEMS

By

YIAN ZHOU

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2015

To my family, friends, teachers and colleagues, without whom I could never reach this far

ACKNOWLEDGMENTS

Pursuing a Ph.D. degree in computer engineering was a magnificent as well as challenging journey to me. Along this journey, I have received many inspiration, encouragement, and support from my family, friends, teachers, and colleagues. I would like to express my deepest appreciation to them. Without them, I could never reach this far.

My deepest gratitude goes first to Dr. Shigang Chen, my Ph.D. advisor and my mentor. He not only introduced me to the wonder world of academic research, but also expertly guided me through my graduate education. I am so lucky and grateful to join his research group. I have learned a lot from him over the past five years, not just how to conduct academic research, but also how to embrace the unexpected in life. His unwavering enthusiasm for research has always inspired me to uncover new field of study, and his personal generosity has helped make my life at US enjoyable. I am extremely thankful for his excellent guidance, caring, patience, and encouragement.

My appreciation also extends to my Ph.D. committee members. They are Dr. Sartaj Sahni, Dr. Yuguang Fang, Dr. Ye Xia, Dr. Jih-Kwon Peir, and Dr. Jose Fortes. I am really thankful for their advice and support during my Ph.D. study at University of Florida.

I also would like to thank all the fellow students and researchers in my research group. Their names are Ming Zhang, Tao Li, Yan Qiao, Wen Luo, Zhen Mo, Min Chen, You Zhou, Qingjun Xiao, Youlin Zhang, Jia Liu, Zhiping Cai, and Olufemi O Odegbile. Thanks for the fresh insights they gave me and all the fun they brought me.

Lastly, I am fully indebted to my parents, who always love me, believe me, and bless me with a life of joy and success.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

PRIVACY-PRESERVING TRANSPORTATION TRAFFIC MEASUREMENT IN INTELLIGENT CYBER-PHYSICAL ROAD SYSTEMS

By

Yian Zhou

December 2015

Chair: Shigang Chen
Major: Computer Engineering

Traffic volume measurement is critical in transportation engineering and vehicular networks. Existing research on traffic volume measurement mainly focuses on single-point traffic statistics. In this dissertation, we switch our view from *single-point* to *multi-point*, and study the important problem of privacy-preserving multi-point traffic volume measurement in intelligent cyber-physical road systems (CPRS), which complements the state of art. We take advantage of the capabilities provided by CPRS to exploit the potential for a fundamental shift in the way how traffic data in support of multi-point traffic volume measurement can be automatically collected. The objective is to allow transportation authorities to automatically collect and efficiently measure the aggregate multi-point traffic volume data from CPRS without learning information about individual vehicles.

In this dissertation, we start with the problem of privacy-preserving two-point traffic volume measurement in CPRS, and propose four novel measurement schemes to solve this problem, with varying degrees of efficiency, accuracy, and privacy. Our first two schemes protect vehicles' identities through keyed signatures based on a family of commutative one-way hash functions, and they can achieve exact measurement results. The third and fourth schemes achieve better privacy for vehicles through shared bit array masking, protecting vehicles' identities as well as their travelling trajectory. They are also much more efficient, and can gracefully control the tradeoff between vehicles' privacy and measurement accuracy. In particular, our third scheme utilizes fixed-length bit arrays, and it works great under the

assumption of similar traffic among different locations. Our fourth scheme removes this assumption of traffic similarity through variable-length bit arrays, and it can fit in more realistic situations where different locations observe different traffic volume.

After that, we extend our idea of variable-length bit array masking to address the problem of privacy-preserving three-point traffic measurement, and eventually present a framework to deal with the general problem of privacy-preserving multi-point traffic measurement. We demonstrate the feasibility, scalability, and superior performance of our solutions through mathematical proofs, numerical analysis, as well as extensive simulations. The research results in this dissertation can be applied to a broad spectrum of applications in vehicular networks and transportation engineering. Furthermore, they have potential applications beyond vehicular networks, such as privacy-preserving traffic estimation in a subway system with tagged toll cards. It is also possible for them to be used for estimating the movement patterns of mobile users in a corporate wireless network.

CHAPTER 1
INTRODUCTION

1.1 Research Background

Traffic volume measurement is critical in transportation engineering and vehicular networks. It provides essential inputs to the most basic functions of road planning and management [1]. Briefly speaking, traffic volume statistics can be summarized into two categories, *"single-point"* statistics and *"multi-point"* statistics. *Single-point* statistics state the number of vehicles traversing a specific point (geographical location) in the road system, while *multi-point* statistics describe the number of vehicles traveling through multiple points (geographical locations), during some measurement period. They are both very important to a variety of transportation studies. However, prior research on traffic volume measurement has mainly focused on *"single-point"* statistics [2–7], while the measurement of *"multi-point"* statistics remains an open research problem. The research scope of this dissertation is "multi-point" traffic measurement, which complements the state of art.

The blossom of intelligent Cyber-Physical Road Systems (CPRS) in recent years [8–19] exposes the potential for a fundamental shift in the way how traffic data can be collected. Enabled by latest technologies of wireless communications and on-board computer processing in CPRS, transportation systems can now automatically collect traffic data from vehicles on road, which can then be used in traffic volume measurement. However, challenges remain to be tackled before the beauty of CPRS can be fully appreciated by its large audience. As more and more people concern about their privacy, any traffic measurement scheme to be widely accepted and applied in CPRS should put travellers' privacy at its top priority. This motivates our work to investigate on privacy-preserving multi-point traffic volume measurement. The challenge of our work is to enable the automatic collection and efficient measurement of aggregate multi-point traffic data while preserving the privacy of individual vehicles (henceforth the privacy of travellers in the vehicles).

Traffic Volume Measurement: Research in transportation traffic measurement can be briefly summarized into two categories, measurement of *"single-point"* traffic statistics and measurement of *"multi-point"* traffic statistics. In the past, the research focus is on the estimation of *"single-point"* traffic statistics such as annual average daily traffic (AADT), which state the number of vehicles passing a specific *point* (geographical location) during some measurement period (e.g., a day for the case of AADT), and various predication models [2–7] have been proposed to measure them using data recorded by roadside units (RSU) in CPRS, such as automatic traffic recorders (ATR) installed at road sections. For example, Mohamad et al. develop a multiple linear regression model which incorporates demographic variables to measure AADT for country roads in [2], and Lam et al. adopt artificial neural networks to estimate AADT by using short period counts for urban areas in [3]. Other research efforts that belong to this category include the spatial statistical method proposed by Eom et al. in [4], the support vector regression model presented by Neto et al. in [5], the absolute deviation penalty procedure designed by Yang et al. in [6], and the regression and Bayesian based model derived by Tsapakis et al. in [7], etc.

"Multi-point" traffic statistics, on the other hand, describe the number of vehicles traveling through multiple points (geographical locations) during some measurement period. In particular, two-point (also commonly referred to as point-to-point) traffic volume measures how many vehicles have traversed two given locations, and three-point traffic volume measures how many vehicles have traversed three given locations, during a measurement period. Similar to single-point traffic statistics, multi-point traffic statistics provide essential input to a variety of studies, including estimation of traffic link flow distribution as part of investment plan, calculation of road exposure rates as part of safety analysis, and characterization of turning movements at intersections for signal timing determination, etc. [1] However, there are only a handful of efforts in literature that deal with the measurement of multi-point traffic statistics, let alone the more challenging problem where the privacy of vehicles (henceforth the privacy of travellers in the vehicles) is of concern. Furthermore, the existing solutions all have their

limitations, suffering from either high computation overhead or violation of vehicle's privacy. For example, Lou and Yin propose to infer two-point traffic statistics from single-point traffic data in the recent work of [20], but the practicability of their scheme is limited by its high computation overhead. Google announced to provide real-time traffic data service in Google maps [21], but their approach cannot assure vehicle's privacy since it uses GPS and Wi-Fi in phones to track locations [22]. Given the state of art, it is imperative to have an efficient scheme to measure multi-point traffic volume while preserving vehicle's privacy.

Intelligent Cyber-Physical Road Systems: CPRS has emerged as one of the most promising research areas in road networks. It integrates the latest technologies in wireless communications, on-board computer processing, sensors, GPS navigation, etc., into transportation systems to enhance its safety, efficiency, and resiliency, and improve the driving experience [8] [9]. In particular, IEEE has standardized Dedicated Short Range Communications (DSRC) under IEEE 802.11p [10], which supports transmitting/receiving messages between vehicles and RSUs. Also, the IntelliDrive [11] from USDOT [12] envisions a nationwide system where vehicles communicate with RSUs in real time via DSRC.

Greatly advanced by new technologies in vehicular communications and networking [13–19], CPRS provides the potential for a fundamental shift in how traffic data are collected: instead of the traditional methods of household interviews and road surveys, which are both time consuming and labor intensive, traffic data can now be automatically collected by RSUs while vehicles are on road. This advantage of CPRS also greatly facilitates traffic volume measurement. For example, when a vehicle passes by an RSU, it can report its unique ID, such as its Vehicle Identification Number (VIN), to the RSU. From the IDs collected by all RSUs, we can easily figure out the multi-point traffic data. However, this straightforward approach to measure multi-point traffic volume leads to serious privacy breaching as it also tracks the entire moving history of vehicles, which is clearly not acceptable to the travellers. In order for the beauty of CPRS to be fully embraced by its large audience, the privacy of individual vehicles must first be taken good care of.

Privacy Issues: As more and more people concern about their privacy, any traffic measurement scheme to be widely accepted and applied in CPRS should put travellers' privacy at its top priority. The transportation authorities from different countries have put forward a number of principles to protect travelers' privacy. For instance, the "anonymity by design" principle required by IntelliDrive [11] from USDOT [12] aims at privacy protection in the first place. Keeping the requirement of privacy preservation in mind, having the vehicles report their unique IDs such as their VINs is clearly not acceptable. Other permanently or temporarily fixed numbers also bare the potential of giving away the vehicles' moving trajectory, so having vehicles report them is not acceptable either. The challenge of addressing the privacy concerns of travellers while measuring multi-point traffic volume opens the door to an interesting research problem: How to design measurement schemes in which vehicles never transmit any unique ID or fixed number for privacy protection, yet the random and de-identified information that the vehicles report still supports the measurement of traffic among multiple different locations? This is where our work originates.

1.2 Dissertation Overview

In this dissertation, we focus on the important research problem of privacy-preserving multi-point traffic volume measurement in CPRS, which measures the number of vehicles traveling through multiple geographical locations during some measurement period. We take advantage of the capabilities provided by CPRS to exploit the potential for a fundamental shift in the way how traffic data in support of multi-point traffic volume measurement can be automatically collected. The objective is to allow transportation authorities to automatically collect and efficiently measure aggregate multi-point traffic data from CPRS without learning information about individual vehicles. During our course of research, we stress that transportation traffic volume measurement is a critical subject in CPRS. We also bare in mind that, in the broad context of vehicular and general computer networks, there are many other important topics such as wireless communication [23–27], network measurement [28–35], privacy and security [36–39], cloud computing [40–44], etc. Although we do not address those

topics in this dissertation, they may interact with traffic volume measurement in CPRS under certain scenarios where new research problems and applications may sprout.

In the remaining of this dissertation, we focus on the problem of privacy-preserving multi-point traffic volume measurement in the context of CPRS. We first formally define the research problem, and then design four novel schemes for privacy-preserving two-point (point-to-point) traffic measurement. We analyze their performance, and discuss their advantages and disadvantages. After that, we investigate the possibility to extend our design to address the more challenging problem of privacy-preserving three-point traffic measurement, and eventually present a general framework to measure traffic volume among three or more locations. Below is an overview of the dissertation.

In Chapter 2, we formally define the problem of privacy-preserving multi-point traffic volume measurement in the context of CPRS. We first introduce the system model, problem definition, and threat model, then present three important performance metrics to evaluate a traffic measurement scheme. After that, we discuss some straightforward solutions to privacy-preserving two-point traffic measurement as well as their limitations.

In Chapter 3, we propose two novel schemes for privacy-preserving two-point traffic measurement through keyed signatures [45]. The idea is that, since globally unique IDs like VINs and other permanently or temporarily fixed numbers that are transmitted repeatedly by a vehicle can be exploited for tracking purpose, IDs or other fixed numbers should be preprocessed and protected by keys before transmission to the RSUs. In other words, RSUs will only be able to collect keyed signatures of vehicles' IDs. To achieve the goals of both traffic measurement and privacy preservation, we utilize the nice properties of CPRS and also a family of commutative one-way hash functions (COHF) to come up with two novel measurement schemes through keyed signatures. In Chapter 3, we first introduce the family of COHFs, and discuss how to construct the COHFs. Then we describe our first two schemes through keyed signatures based on the COHFs. Both schemes contain three phases, initialization, online reporting, and offline measurement. The key process is: a common COHF is deployed to all

RSUs and vehicles, and vehicles apply the hash function to produce Keyed signatures of their IDs (referred to as KIDs) using the keys which are either obtained from RSUs that they pass by or randomly picked by the vehicles themselves. The KIDs, instead of real IDs, are reported to RSUs for two-point traffic volume measurement. We analyze the performance of both schemes, and summarize their advantages and disadvantages.

In Chapter 4, we present our third novel scheme for privacy-preserving two-point transportation traffic measurement [46] [47], which combines the beauty of both shared bit arrays and a statistical method, maximum likelihood estimate (MLE) [48]. We first discuss the motivation for us to change the perspective from using keyed signatures to utilizing shared bit arrays, then introduce our novel scheme based on shared bit array masking. We analyze its performance through mathematical proofs, numerical and simulation results, which demonstrate its applicability and scalability for large-scale road networks.

In Chapter 5, we propose our fourth novel design for privacy-preserving two-point traffic measurement [49], which is an extension of the previous scheme [47], to fit in a broader spectrum of real-life situations, where different RSUs may face different traffic volume. In contrast to the solution with fixed-length bit arrays in [47], our extension design utilizes variable-length bit arrays to encode traffic data reported by vehicles, where the length of the bit array in an RSU is determined by and reflect the single-point traffic volume at the corresponding location where the RSU is installed. In order to support traffic volume measurement based on those variable-length bit arrays, we also propose a novel "unfolding" technique. Through mathematical and numerical analysis as well as extensive simulations, we demonstrate that the extension scheme based on variable-length bit arrays has comparable efficiency with the previous scheme based on fixed-length bit arrays [47] and furthermore, it can easily fit in the more realistic transportation model and achieve far better privacy and accuracy than the previous scheme.

In Chapter 6 and 7, we further investigate the possibility to extend our existing designs of privacy-preserving two-point traffic measurement to solve the more challenging and general

problem of privacy-preserving multi-point traffic measurement [50], which measures the number of vehicles passing through an arbitrary set of three or more RSUs (locations) during any measurement period, while preserving the privacy of individual vehicles. The generalization process is very natural. In Chapter 5, we have shown that through allocating variable-length bit arrays to different RSUs based on their single-point traffic volume and having vehicles report random bits in the bit arrays as they pass by RSUs, we can well preserve the privacy of vehicles; and through the "unfolding" technique we can put together two variable-length bit arrays to measure the traffic volume between the two corresponding RSUs. If we can unfold two variable-length bit arrays to put together the corresponding two-point traffic volume, we may also be able to unfold three or more variable-length bit arrays to compute the corresponding multi-point traffic volume. Based on this idea, in Chapter 6, we design a novel scheme for privacy-preserving three-point traffic volume measurement in CPRS, and perform extensive simulations to demonstrate its applicability and scalability. In Chapter 7, we generalize our designs and eventually present a general framework for privacy-preserving multi-point traffic measurement, which can efficiently measure traffic volume among an arbitrary set of three or more points (locations) while preserving vehicles' privacy.

Finally, in Chapter 8, we conclude our work.

CHAPTER 2
PRELIMINARIES

In this chapter, we formally define the research problem of privacy-preserving multi-point transportation traffic volume measurement in the context of CPRS. The objective is to allow transportation authorities to automatically collect and efficiently measure aggregate multi-point traffic volume data from CPRS without learning information about individual vehicles. We first introduce the system model, the problem definition, and the threat model, then present three important performance metrics to evaluate a traffic measurement scheme. After that, we briefly discuss some straightforward solutions to privacy-preserving two-point traffic measurement as well as their limitations.

2.1 System Model

We consider an intelligent CPRS model as illustrated in Figure 2-1, which involves three groups of entities: vehicles, RSUs, and a central server, with the latter two forming the infrastructure. Each vehicle has a unique ID, such as its VIN or other number chosen permanently or temporarily. For example, each vehicle can randomly pick its ID (from a large space) at the beginning of a day. Each RSU also has its unique ID. Both vehicles and RSUs are equipped with computing and communication capabilities, such as on-board computer chips and communication modules. Vehicles can communicate with RSUs in real time via DSRC [10]. RSUs are connected to the central server through wired or wireless means. They collect information from vehicles and transfer it to the central server for further processing on a periodical basis, e.g., at the end of each measurement period (such as a day).

2.2 Problem Statement

Given any d locations where RSUs are installed, we define the set of vehicles that pass all the d locations during some measurement period T as a d-point traffic flow. The size of the d-point traffic flow is the number of vehicles in this set, called the d-point traffic volume. For example, the two-point traffic volume among a set of two RSUs $\{R_x, R_y\}$ measures the number of vehicles passing both R_x and R_y, while the three-point traffic volume among a set

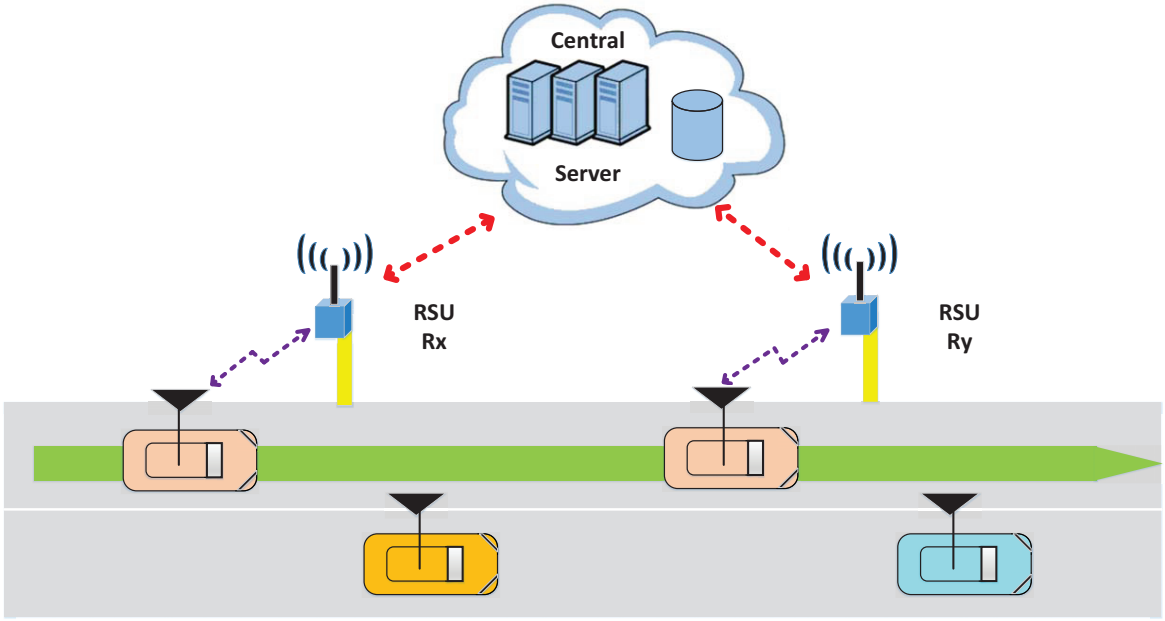


Figure 2-1. Intelligent cyber-physical road system model. It includes three groups of entities: vehicles, RSUs, and a central server. Each vehicle or RSU has a unique ID, and is equipped with computing and communication capabilities. Vehicles can communicate with RSUs in real time via DSRC. RSUs are connected to the central server through wired or wireless means.

of three RSUs $\{R_x, R_y, R_z\}$ describes the number of vehicles passing all three RSUs, R_x, R_y, and R_z. The problem is to measure the d-point traffic volume ($d > 1$) under the CPRS modeled above while protecting vehicles' privacy.

To achieve the privacy protection end, we need a solution in which a vehicle never transmits any unique identifier or any permanently or temporarily fixed data. Ideally, the information transmitted by the vehicles to the RSUs looks totally random, out of which neither the identity nor the trajectory of any vehicle can be pried with high probability. One typical application scenario is to measure multi-point traffic in a city with a typical measurement period of a day, where RSUs may be deployed at any interested locations in the city.

2.3 Threat Model

We assume RSUs are semi-trusted. On the one hand, all RSUs are from trustworthy authorities, which can be enforced by authentication based on the public key infrastructure (PKI), and RSUs will not be compromised. Each vehicle is pre-installed with the public keys

of the trusted third-parties. Each RSU must have a public-key certificate from them. It broadcasts the certificate in each query it sends out. When receiving a query from an RSU, the vehicle verifies the certificate, and then uses the RSU's public key to authenticate it. On the other hand, the authorities may exploit the information collected by RSUs to track individual vehicles when they need to do so. For instance, if a vehicle transmits any unique identifier upon each query, that identifier can be used for tracking purpose.

We consider a threat model with passive adversary. The adversary can be the semi-trusted RSUs, or an outsider of the system, which can eavesdrop on the communications between the vehicles and RSUs, and record and analyze all captured messages. But it will not perform any active attack to avoid being detected. Note that there are also other ways to track a vehicle, for example, tailgating the vehicle, or setting cameras near RSUs to take photos and using image processing to recognize it. These methods are beyond the scope of the work in this dissertation – we will focus on preventing automatic tracking caused by the traffic measurement scheme itself.

We also assume that a special MAC protocol such as SpoofMAC [51] is applied to support privacy preservation such that the MAC address of a vehicle is not fixed. For instance, when responding to an RSU, the vehicle may pick an MAC address randomly from a large space for one-time use. Since the number of vehicles in the vicinity of the RSU is limited, the probability for two vehicles to choose the same MAC address can be made negligibly small when the address space is sufficiently large. Through this, vehicles can report information to RSUs for traffic volume measurement without revealing their true identities.

2.4 Performance Metrics

In this dissertation, we consider three performance metrics to evaluate a traffic volume measurement scheme: measurement accuracy, computation overhead, and preserved privacy. They are defined in the following.

2.4.1 Measurement Accuracy

Let n_c be the true size of a d-point traffic flow and $\hat{n}_c$ be the corresponding measurement result. We can evaluate the measurement accuracy of a scheme in two different ways. First, for individual/single-time measurement result, we can observe the measurement accuracy through a parameter called error ratio, $r = \frac{|\hat{n}_c - n_c|}{n_c}$, which states the relative deviation of the measurement result from the real traffic flow size. Clearly, smaller r represents more accurate measurement result, and vice versa.

Second, we can statistically analyze the measurement accuracy of a traffic measurement scheme. For example, we can specify the measurement accuracy through a parameter β which satisfies the following requirement: the probability for n_c to fall into the interval $[\hat{n}_c \cdot (1-\beta), \hat{n}_c \cdot (1+\beta)]$ must be at least α, where α is a pre-determined parameter in the range of $[0, 1]$. For a given probability α, a smaller value of β means more accurate measurement results. For example, when $\alpha = 95\%$, a solution with $\beta = 0.05$ is more accurate than a solution with $\beta = 0.1$ because the former ensures the measured traffic flow size has a probability of 95% to be within $\pm 5\%$ deviation from the true value, while the latter only ensures the measured result to be within $\pm 10\%$ deviation from the true value under the same probability. An alternative way to measure the accuracy of a scheme is evaluating the bias and standard deviation of $\frac{\hat{n}_c}{n_c}$. Clearly, a good measurement scheme should have close-to-zero bias and relatively small standard deviation.

2.4.2 Computation Overhead

We consider the computation overhead for vehicles, RSUs, and the central server. For vehicles, we measure the computation overhead for each vehicle per RSU en route. For RSUs, we measure the computation overhead for each RSU per passing vehicle. For the central server, we measure the computation overhead for it to measure the d-point traffic volume among an arbitrary set of d RSUs. To scale to the large road systems as in nowadays, we require the computation overhead for each involving group of entities in the measurement scheme to be as small as possible.

2.4.3 Preserved Privacy

The essence of privacy preservation in multi-point transportation traffic measurement is allowing the adversary only a limited chance of identifying partially or fully any trajectory of any vehicle. Accordingly, we define the preserved privacy of a scheme from two levels: the first level is to hide the identity of each participating vehicle from unauthorized disclosure, and furthermore, the second level is to protect the "trace" of any vehicle from being identified, where a trace of a vehicle is a pair of RSUs it has passed by. The first level is straightforward, while the second level is more difficult to capture. In this dissertation, we quantify the second-level privacy of a measurement scheme through a parameter p which satisfies the following requirement: the probability for any "trace" of any vehicle to not be identified must be at least p. In other words, under the situations when a vehicle's identity is revealed at one location, the probability for the adversary to not be able to back trace any other location that this vehicle has traversed based on this one-time revealed identity must be at least p. For example, if a car keeps transmitting its fixed ID to the passing RSUs as it travels, p will be 0. One can see that a larger value of p means better privacy. Intuitively, a scheme with $p = 0.9$ is better than one with $p = 0.5$ in terms of privacy, because the latter gives the adversary a better chance to link traces of a vehicle to obtain its trajectory since it allows the traces to be identified with a higher probability, i.e., $1 - p$.

Note that our privacy definition agrees with the privacy requirements as proposed in [52] and [53]. The work [52] surveys different privacy metrics [53] [54] to characterize the vehicles' privacy level. In contrast to the anonymity set analytical models [53] which vary as the traffic conditions change, it is easier to judge the privacy level of a traffic measurement scheme through a single quantitative metric of probability which fits the global system and applies to various traffic conditions and scenarios. The work [54] considers the overall probability for an adversary to follow a vehicle from origin to destination (OD data) with an entropy perspective. However, we believe a stronger privacy, which considers the probability for the trajectory of a vehicle (as opposed to the narrower OD data) not to be identified by any adversary, is

desirable for CPRS. For example, the identity of a vehicle may be revealed at some location (not necessarily at the origin or destination of its trip), e.g., through a photograph triggered by the vehicle rushing a red light or by a police car stopping the vehicle. These circumstances are not covered by the privacy definition of [54], but are captured by ours.

2.5 Straightforward Solutions and Their Limitations

There are some straightforward solutions to measure the two-point traffic flow size between an arbitrary pair of RSUs in the road system. One approach is making vehicles report their IDs to all RSUs that they pass by. RSUs collect the IDs from the passing vehicles. At the end of each measurement period, all RSUs send their collected ID sets to the central server, which then measures the two-point traffic flow size between each pair of RSUs by simply comparing the two corresponding ID sets: if a vehicle ID appears in both ID sets, then the vehicle must have passed both RSUs. Thus, the number of IDs that appear in both ID sets equals the real two-point traffic flow size between the two corresponding RSUs. However, this simple approach leads to serious privacy breaching as it reveals vehicles' identities along the way.

A natural follow-up thinking is making vehicles report keyed signatures of their IDs (KIDs) instead of their real IDs to the RSUs en route [45]. The central server will compute the two-point traffic flow sizes based on the KID sets collected by RSUs. To prevent the adversary from using fixed KIDs to identify vehicles, each vehicle has many KIDs generated by different keys. However, the KIDs of a vehicle must satisfy the following property: they will produce the same result after a certain procedure of computations, allowing the central server to find out they represent the same vehicle. In this scheme, although vehicles' true identities are hidden (i.e., first-level privacy is preserved), traces of each vehicle are still revealed through time and can be linked to obtain the vehicle's full trajectory. We will discuss this scheme in more details in the next chapter.

An alternative approach is having the RSUs broadcast their IDs (RIDs). Each vehicle will record the RIDs of all RSUs it has passed by, and transmit them to every RSU that it passes

in the future. RSUs collect those RIDs from passing vehicles, and send them to the central server at the end of each measurement period. To compute the size of a two-point traffic flow between two RSUs, R_x and R_y, the central server simply goes through the RID set collected by R_y (R_x), and count the number of times that R_x (R_y) appears in this set. This is the directional two-point traffic flow size from R_x (R_y) to R_y (R_x). The undirectional two-point traffic flow size between R_x and R_y is the sum of both directional traffic flow sizes. Clearly, this approach also reveals a vehicle's trajectory in the form of a list of RIDs sent to each RSU that it passes. The identity of a vehicle may be revealed at some point by a photograph triggered by the vehicle rushing a red light or by a police car stopping the vehicle. When the identity is combined with the trajectory transmitted by the vehicle, the entire traveling path of the driver will be revealed, which is not acceptable either.

CHAPTER 3
PRIVACY-PRESERVING TWO-POINT TRAFFIC MEASUREMENT THROUGH KEYED SIGNATURES

In the following three chapters, we will present four novel solutions for privacy-preserving two-point traffic flow measurement. Recall that our goal is to allow transportation authorities to automatically collect and efficiently measure the aggregate two-point traffic flow data from CPRS without learning information about individual vehicles. To achieve this goal, we first propose two measurement schemes through keyed signatures [45] in this chapter. Our idea is that, since globally unique IDs like VINs and other permanently or temporarily fixed numbers that are transmitted repeatedly by a vehicle can be exploited for tracking purpose, IDs or other fixed numbers should be preprocessed and protected by keys before transmission. In other words, RSUs will only be able to collect keyed signatures of vehicles' IDs.

To achieve the goals of both traffic measurement and privacy preservation, we utilize the nice properties of CPRS and also a family of commutative one-way hash functions (COHF) to come up with two novel measurement schemes through keyed signatures. Here is an overview of both schemes: a common COHF is deployed to all RSUs and vehicles, and vehicles apply the hash function to produce Keyed signatures of their IDs (referred to as KIDs) using the keys which are either obtained from RSUs that they pass by or randomly picked by the vehicles themselves. The KIDs, instead of real IDs, are reported to RSUs for traffic flow measurement. At the end of each measurement period, RSUs will send their collected KID set to the central server, which will measure the traffic flow between two arbitrary RSUs based on the two corresponding KID sets.

In the remaining of this chapter, we first introduce the family of COHFs, and discuss how to construct the COHFs, then present our first two schemes that are based on the COHFs. A summary of the two measurement schemes through keyed signatures will be given at the end of this chapter, which further motivates our later idea of two-point traffic flow measurement based on shared bit array masking to be discussed in the next two chapters.

3.1 Commutative One-Way Hash Functions

In this section, we first introduce the family of COHFs, and then discuss how to construct the COHFs.

3.1.1 Definition of COHFs

Consider a hash function $h : A \times B \to C$, where the two arguments are a hash input and a hash key, respectively. A commutative one-way hash function, as its name suggests, satisfies both one-wayness and commutativity. The definitions of the properties below are collated from [55] and [56].

Definition 1. *A family of one-way hash functions (OHF) is a set of functions $h_n : V_n \times K_n \to Z_n$, which satisfy the following three properties:*

- *Ease of computation: there exists a polynomial P such that for each integer n, $h_n(v, k)$ is computable in time $P(n, |v|, |k|)$ for all $v \in V_n$ and all $k \in K_n$.*
- *Preimage resistance: there is no polynomial P such that, given n, $k \in K_n$, and $z \in Z_n$, there exists a probabilistic polynomial time algorithm which can find $v \in V_n$ satisfying $h_n(v, k) = z$ with probability greater than $1/P(n)$ for sufficiently large n, when k is chosen uniformly from K_n and z is chosen uniformly from Z_n.*
- *2nd-preimage resistance: there is no polynomial P such that, given n, $(v, k) \in V_n \times K_n$, and $k' \in K_n$, there exists a probabilistic polynomial time algorithm which can find $v' \in V_n$ satisfying $h_n(v, k) = h_n(v', k')$ with probability greater than $1/P(n)$ for sufficiently large n, when (v, k) is chosen uniformly among all elements of $V_n \times K_n$ and k' is chosen uniformly from K_n.*

In this case, h_n is said to have the one-wayness property.

In Definition 1, the first property requires that OHF is relatively easy to compute (in polynomial time). The second property requires that it is computationally infeasible to find an input which can be hashed to an arbitrarily pre-specified output. The third property requires that it is computationally infeasible to find a second input that can be hashed under an arbitrarily pre-specified key to the same output as an arbitrarily pre-specified input and key.

Definition 2. *A commutative hash function (CHF) is a hash function* $h_n : V_n \times K_n \rightarrow V_n$*, which satisfies the commutativity property: for all* $v \in V_n$ *and for all* $k, k' \in K_n$*,* $h_n(h_n(v,k),k') = h_n(h_n(v,k'),k)$.

From Definition 2, one can see that commutativity lies in the hash keys: given any input and two keys, commutativity states that changing the order in which the two keys are applied to the input won't change the hash result. Further observed, if the range of h_n equals the domain of its first argument, we can exploit a new family of commutative one-way hash functions which shall satisfy both one-wayness and commutativity.

Definition 3. *Commutative one-way hash functions (COHF) are a family of hash functions which have both one-wayness property and commutativity property.*

In the next two sections, we will see one crucial benefit of utilizing the family of COHFs: Vehicles can transmit their KIDs generated by hashing their IDs under totally different keys, and be sure that no one will be able to get their IDs, even knowing the keys used by the vehicles (one-wayness). Yet the KIDs still allow traffic flow measurement as demanded (through commutativity).

3.1.2 Construction of COHFs

Now we discuss how to construct COHFs. According to Definition 3, a COHF is a hash function that satisfies both one-wayness and commutativity. There can be different constructions of COHFs given different types of hash functions, and here we discuss one construction based on the exponentiation modulo n function, $h_n(v,k) = v^k \ mod\ n$. We claim that h_n is a COHF with some restrictions on n.

Definition 4. *A prime* p *is defined to be safe if* $p = 2p' + 1$ *where* p' *is an odd prime. A number* n *is defined to be a rigid integer if* $n = pq$ *where* p *and* q *are distinct large safe primes.*

Theorem 1. *The function* $h_n(v,k) = v^k \ mod\ n$ *is a commutative one-way hash function if* n *is a rigid integer.*

Proof: Clearly, h_n is commutative. As to the one-wayness, h_n satisfies the property of *ease-of-computation* since there are efficient methods to perform exponentiation of a base to an exponent in polynomial time (e.g., [57]). Note that the selection of n and h_n follows the RSA cryptosystem [58]. Therefore, the *preimage resistance* property of h_n also follows the cryptographic security of RSA [59]. The third property, *2nd-preimage resistance*, is rooted in the characteristics of rigid integers. It is demonstrated in [56] that if n is a rigid integer, finding collisions with specific constraints such as 2nd-preimage cannot be done in polynomial time. This completes the proof. □

Therefore, to construct a COHF based on the exponential function $h_n(v, k) = v^k \ mod \ n$, we only need to determine a large rigid integer n. There is a practical method to construct it, and the basic idea is that for $n = pq$ to be a rigid integer, each of p, q, $\frac{(p-1)}{2}$ and $\frac{(q-1)}{2}$ must be primes congruent to 5 modulo 6. Therefore, the process is to first select a "random" large integer p' that is congruent to 5 modulo 6 until one is found such that p' and $2p'+1$ are both prime, and then choose a suitable q' similarly. After that, n can be easily constructed by $n = pq = (2p'+1)(2q'+1)$.

3.2 First Scheme Based on COHFs

Taking advantage of the COHFs, we propose our first scheme for privacy-preserving two-point traffic flow measurement. In this scheme, each measurement period consists of three phases: initialization, online reporting, and offline measurement. First, during the initialization phase, vehicles and RSUs are pre-configured with a common COHF h_n, and each RSU also generates a unique key for itself. Then during the online reporting phase, vehicles will generate keyed-signatures (KIDs) using their IDs and the keys received from passing RSUs, and send their KIDs (instead of their real IDs) to the RSUs. Finally, the RSUs will send their keys and collected KID sets to the central server, who will measure the two-point traffic volume between two arbitrary RSUs based on their keys and collected KID sets. In this section, we first present the three measurement phases of our first scheme, then analyze its performance. We end this section with a brief discussion about a potential disadvantage of our first scheme, which

motivates our design of the second enhanced scheme based on COHFs to be presented in the next section.

3.2.1 Initialization

The first phase is initialization, when a common COHF h_n must be pre-distributed to all vehicles and RSUs. As we discussed earlier, the COHF h_n is determined by a large rigid integer n. Therefore, all RSUs and vehicles are pre-configured with a suitable value of n. Also, clocks of RSUs are loosely synchronized as they are all connected to the central server through wired or wireless means. Every RSU generates a random number as its hash key for the current measurement period. With the central server's assistance, all hash keys are unique: Let k_x be the hash key generated by RSU R_x. We require that, for any two RSUs R_x and R_y, their keys k_x and k_y be different. If the server finds two hash keys reported from RSUs are the same, it will inform one of them to regenerate a key. The key uniqueness requirement serves an important purpose of privacy perservation, which will be explained later.

3.2.2 Online Reporting

The online reporting phase securely collects information for traffic flow measurement. The RSUs broadcast queries in pre-set intervals (e.g., once a second), ensuring that each passing vehicle receives at least one query and meanwhile giving enough time for the vehicle to reply. Collisions can be resolved through well-established CSMA or TDMA protocols, which are not the focus of our design. Every query that an RSU sends out includes the RSU's ID, public-key certificate, as well as its current hash key. When a vehicle, whose ID is v, receives a query from an RSU R_x, it first verifies the certificate, and then uses the RSU's public key to authenticate the RSU. After verifying that R_x is from the trustworthy authority, the vehicle generates a KID based on its ID v and the RSU's key k_x by computing a hash $d = h_n(v, k_x) = v^{k_x} \ \textit{mod}\ n$. After that, it reports the KID d to the RSU, which then stores d in its local storage.

3.2.3 Offline Measurement

At the end of each measurement period, the traffic flow sizes between pairs of RSUs are computed based on the KIDs collected by RSUs during the online reporting phase. Specifically,

every RSU will send its key as well as the collected KID set to the central server, which will be in charge of the offline traffic flow size computation.

Thanks to the commutativity property of h_n, given two sets of KIDs, $H_x = \{h_n(\cdot, k_x)\}$ and $H_y = \{h_n(\cdot, k_y)\}$, collected by two RSUs R_x and R_y respectively, and the two corresponding keys, k_x and k_y, it is easy for the central server to determine the traffic flow size between R_x and R_y. In principle, changing the order in which two keys are applied to the same vehicle ID using COHFs won't change the final hash result. Therefore, the central server simply further hashes each RSU's KID set by the other RSU's key to obtain two double-hashed sets $H_{x,y} = \{h_n(h_n(\cdot, k_x), k_y)\}$ and $H_{y,x} = \{h_n(h_n(\cdot, k_y), k_x)\}$, and the traffic flow size between R_x and R_y simply equals the number of common elements in $H_{y,x}$ and $H_{x,y}$ according to Theorem 2 in the following. Note that if we take the timestamps of the KIDs into consideration, we can easily determine the size of a directional traffic flow for vehicles that appear at R_x first and then appear at R_y at a later time.

Theorem 2. *Given a commutative one-way hash function $h_n(v, k) = v^k \, mod \, n$, for arbitrary vehicle IDs v and v', and arbitrary keys k and k', $h_n(h_n(v, k), k') = h_n(h_n(v', k'), k)$ holds if and only if $v = v'$ holds.*

Proof: The sufficiency is clearly established given the commutativity of h_n. The necessity is granted through two facts. First, h_n is commutative. Second, since the number of vehicles in the vicinity of two RSUs is limited, and the hash space is sufficiently large, the probability for two distinct vehicle IDs to be hashed under the same key to the same value is negligibly small. This completes the proof. □

3.2.4 Scheme Analysis

The proposed scheme preserves vehicles' privacy. As vehicles only transmit their KIDs to RSUs, no one can obtain their real IDs thanks to the one-wayness of the COHF h_n. Vehicles are further protected from being tracked since no fixed information of them is transmitted because of the key uniqueness requirement.

The proposed scheme is also efficient. Each vehicle only needs to compute one hash for each passing RSU, so the computation overhead for each vehicle per RSU en route is $O(1)$. The RSU only need to store the KID that each passing vehicle reports, so the computation overhead for each RSU per passing vehicle is also $O(1)$. As for the central server, to compute a traffic flow size between two RSUs R_x and R_y, it needs to perform a hash for each KID value from the two KID sets, so the total number of hash operations is bounded by $O(n_x + n_y)$, where n_x and n_y are the total number of vehicles passing by R_x and R_y, respectively. Further, to find the common elements among the two double-hashed sets, it needs to sort the two double-hashed sets, which takes $O(n_x \log n_x + n_y \log n_y)$ comparison operations.

3.2.5 Identical-Key Attack

The above analysis assumes the transportation authority (who owns RSUs and the central server) is trustworthy. But this assumption also allows the transportation authority an easy way of tracking vehicles. It may simply set all or a portion of RSUs with the same key. When a vehicle passes these RSUs, its KID stays the same and therefore may be exploited for tracking purpose. To avoid transmitting the same number (KID), a vehicle may keep record of the RSU keys that it has seen before, and will not respond to an RSU if the key from that RSU is already in the vehicle's record.

This solution however causes an under-measurement problem. Suppose during a measurement period (e.g., a day), a vehicle passes by an RSU for two or more times. This is not uncommon in reality. For example, people driving to work are likely to follow the same route back home. While the vehicle contributes twice to traffic volume between home and workplace, it is counted only once (since the vehicle does not respond to the same key). To fully address this concern, we need to make a shift in who is responsible for key generation. We shall move that responsibility from RSUs to the vehicles in order to ensure that the key uniqueness requirement is met.

3.3 Enhanced Scheme Based on COHFs

Instead of using the keys generated by RSUs, we propose an enhanced scheme which lets vehicles choose their own keys to protect their IDs. Still, vehicles and RSUs are pre-configured with a common commutative one-way hash function h_n. RSUs will collect KIDs from vehicles, and a central server will compute traffic flow sizes based on the collected KID sets. The difference is that, RSUs will not just record the KIDs. Instead, it will store a set of $\langle$key, KID$\rangle$ pairs obtained from passing vehicles for measurement purpose. The enhanced scheme also has three phases: initialization, online reporting, and offline measurement.

3.3.1 Initialization

The initialization phase of our enhanced scheme is very simple and similar to the first scheme. First, a common COHF h_n is pre-distributed to all vehicles and RSUs through pre-determining a suitable value of n. Also, clocks of RSUs are loosely synchronized as they are all connected to the central server through wired or wireless means.

3.3.2 Online Reporting

During the online reporting phase, $\langle$key, KID$\rangle$ pairs are securely collected by RSUs from the passing vehicles. More specifically, when a vehicle v passes by an RSU R_x, the vehicle will first verify that the RSU comes from trusted authorities based on the public-key certificate received from the RSU's periodic broadcast. Then the vehicle will randomly choose a hash key k, and compute a hash $d = h_n(v, k) = v^k \ mod \ n$, which serves as a KID of v. After that, the vehicle reports the KID d and the key k to the RSU R_x, which stores this $\langle$key, KID$\rangle$ pair in its local storage.

3.3.3 Offline Measurement

At the end of each measurement period, all RSUs will send their collected data to the central server. Given two sets of $\langle$key, KID$\rangle$ pairs collected by two RSUs R_x and R_y, the central server can compute the size of the corresponding traffic flow based on the hash function h_n's commutativity. The process is to go through these two sets, and for each pair $\langle k_x, d_x\rangle$ collected by R_x, check if there is a pair $\langle k_y, d_y\rangle$ collected by R_y such that

$h_n(d_y, k_x) = h_n(d_x, k_y)$; we say the two pairs share a common double-hashed value in this case. If so, a vehicle is found to pass both RSUs. One can easily verify its correctness through Theorem 2.

3.3.4 Scheme Analysis

The enhanced scheme eliminates the under-measurement problem that is encountered by the previous scheme. Even if a vehicle may pass an RSU for several times, each time it uses a different key to produce a new KID, which will be recorded and counted towards the final measurement result. Therefore, the measured traffic flow sizes should always be equal to the real ones. Observe that the enhanced scheme improves the measurement accuracy at the cost of increased computation overhead for the central server. In order to compute the traffic flow size between two RSUs, R_x and R_y, the central server needs to perform a re-hash for each pair collected by R_x under every key from R_y, and do the same thing for R_y. Suppose the two RSUs have collected n_x and n_y pairs of $\langle$key, KID$\rangle$, respectively. The time complexity for the central server to compute the corresponding traffic flow size will be $O(n_x \cdot n_y)$.

3.3.5 Sampling

To address the efficiency problem, we propose to use sampling to estimate the traffic flow sizes. Given two sets of $\langle$key, KID$\rangle$ pairs collected by two RSUs R_x and R_y, $D_x = \{\langle k_{ix}, d_{ix}\rangle\}_{i=1}^{n_x}$, $D_y = \{\langle k_{iy}, d_{iy}\rangle\}_{i=1}^{n_y}$, it takes $O(n_x \cdot n_y)$ time to calculate the traffic flow size. To reduce computation overhead, we randomly select n'_x elements from D_x and n'_y elements from D_y, denoting them as D'_x and D'_y, respectively. It only takes $O(n'_x \cdot n'_y)$ time to compute the traffic flow size n'_{xy} from such a sample. Based on n'_{xy} and the sampling probabilities, we can construct the MLE estimator [48] of n_{xy} as

$$\hat{n}_{xy} = n'_{xy} \times \frac{n_x}{n'_x} \times \frac{n_y}{n'_y}, \tag{3–1}$$

which is derived as follows: The idea is that if two pairs from D_x and D_y share a common double-hashed value, we treat them as a common element in these two sets. So our problem is equivalent to the set-intersection estimation problem: Let X and Y be two sets with $|X| = a$,

$|Y| = b$, $|X \cap Y| = c$. We randomly choose two subsets of elements, X' and Y', with cardinalities a' and b', from X and Y. We find the number of common elements in X' and Y', denoted by c'. The problem is to construct the MLE of c based on c', a, b, a', and b'.

For a randomly selected $e \in X'$, the probability for $e \in X \cap Y$ is $\frac{c}{a}$. Under this condition $e \in X \cap Y$, the probability for $e \in Y'$ is $\frac{b'}{b}$. Combining them, we have $P(e \in Y' | e \in X') = \frac{cb'}{ab}$. There are a' elements in X', so the likelihood function for observing c' common elements in X' and Y' is

$$\mathcal{L} = \left(\frac{cb'}{ab}\right)^{c'} \left(1 - \frac{cb'}{ab}\right)^{a'-c'}. \tag{3–2}$$

We want to find the MLE of c, denoted as $\hat{c}$, which maximizes $\mathcal{L}$. To find $\hat{c}$, we take logarithm on both sides of (3–2):

$$\ln \mathcal{L} = c' \times \ln\left(\frac{cb'}{ab}\right) + (a' - c') \times \ln\left(1 - \frac{cb'}{ab}\right) \tag{3–3}$$

Take the first order derivative of (3–3) and let it be zero. We have $\hat{c} = c' \times \frac{a}{a'} \times \frac{b}{b'}$. By changing the notations to those for our problem, we have $\hat{n}_{xy} = n'_{xy} \times \frac{n_x}{n'_x} \times \frac{n_y}{n'_y}$, which is the MLE of n_{xy}. By adopting the sampling method, the computation overhead for the central server to measure the traffic flow size is reduced from $O(n_x \cdot n_y)$ to $O(n'_x \cdot n'_y)$.

3.4 Simulation

We evaluate the performance of our two schemes through simulations. The programs are written in Matlab, and the experimental platform is a PC featured with an Intel Core 2 E8400 CPU and 4GB RAM, running Windows XP. However, we expect the central server in practice to be much more powerful. The offline measurement may also be outsourced to cloud servers and benefit from parallel work. The datasets used in the simulations are generated such that each vehicle ID or key is a 32-bit number, and two RSUs, R_x and R_y, each store 3,000 vehicle records. There are 500 vehicles that pass both R_x and R_y, i.e., the actual two-point traffic flow size n_{xy} is 500.

Table 3-1. Average computation overhead for the two schemes based on keyed signatures. The unit for the time is thousand of seconds.

First	Second Scheme with Different Sampling Probabilities									
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.001	0.017	0.069	0.156	0.276	0.430	0.620	0.846	1.102	1.394	1.720

In the simulations, we consider two performance metrics. One is *measurement accuracy*, represented by error ratio r:

$$r = \frac{|\hat{n}_{xy} - n_{xy}|}{n_{xy}} \times 100\%, \tag{3–4}$$

where $\hat{n}_{xy}$ is the measured traffic flow size. Clearly, smaller r represents more accurate measurement result, and vice versa. The other is *computation overhead*, measured by time consumed for the central server to obtain $\hat{n}_{xy}$.

Our first scheme has an error ratio of 0% unless it does not respond to the keys that it has seen before (for privacy purpose as we have discussed in Section 3.2.5). Hence, we only measure its time cost. The enhanced scheme addresses the identical-key attack at the cost of higher computation overhead. It has an error ratio of 0% only when the sampling probability p is 1. In our simulations, we vary p from 0.1 to 1, with a step size of 0.1. For each sampling probability p, we randomly draw a fraction p of all records from R_x and do the same for R_y. The offline measurement is performed over the sampled subsets and the traffic flow size are estimated by (3–1). The time cost is measured and the error ratio is computed from (3–4). The process is repeated 10 times to show the statistic effect.

Table 3-1, Figure 3-1 and Figure 3-2 present our simulation results. Table 3-1 shows the computation overhead of the first scheme and the second scheme under varied sampling probabilities p. The two figures are drawn from the simulation results of the second enhanced scheme. Figure 3-1 shows the mean and standard deviation of the error ratio r under varied p. The length of each error bar is two times the standard deviation of r, whose mean is at the center of the bar. We see that both the mean and standard deviation of r decrease with the increment of p. Intuitively, when we increase the sample size, the measurement result is

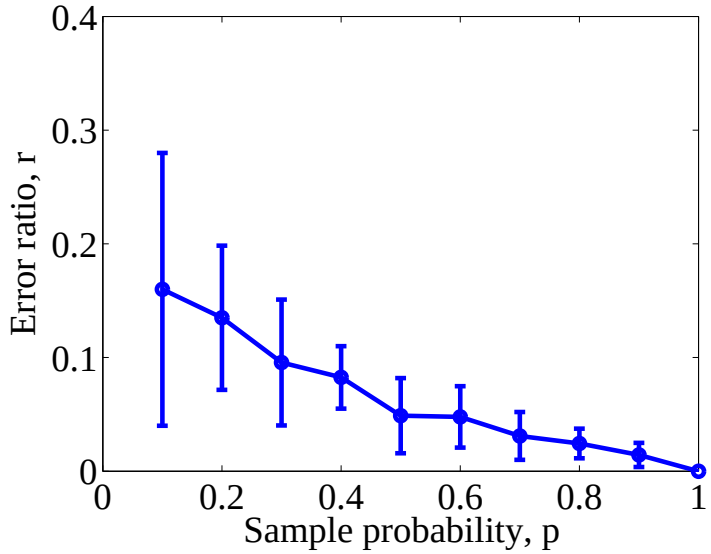


Figure 3-1. Mean and standard deviation of error ratios for the second two-point traffic flow measurement scheme.

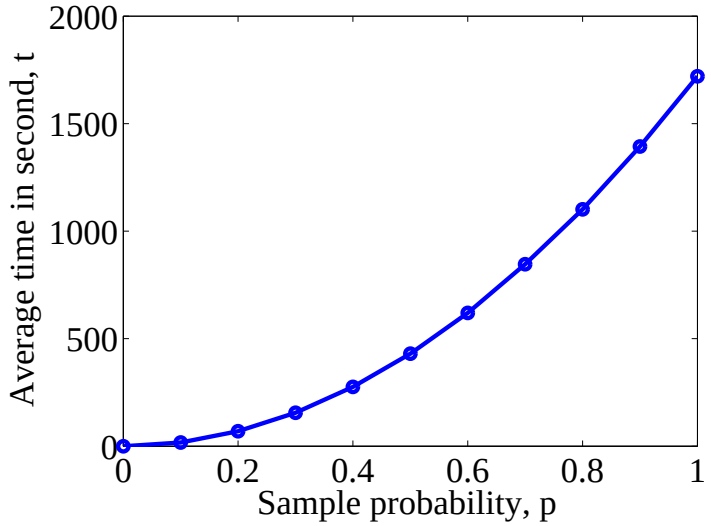


Figure 3-2. Average time overhead for the offline measurement phase of the second two-point traffic flow measurement scheme.

likely to be more accurate. When p equals 1, the error ratio is 0% (the rightmost point of the figure), which agrees with our theoretical prediction. Figure 3-2 shows the average time taken by the central server to measure the traffic flow size under each sampling probability. It is clear that the computation overhead increases quadratically with p, which is also consistent to our analysis in Section 3.3.5. We stress that this is offline computation.

3.5 Summary

In this chapter, we propose two novel two-point traffic measurement schemes through keyed signatures based on COHFs. The goal is to allow transportation authorities to automatically collect and efficiently measure the aggregate two-point traffic flow data from CPRS without learning information about individual vehicles. The idea is that, since globally unique IDs like VINs and other permanently or temporarily fixed numbers that are transmitted repeatedly by a vehicle can be exploited for tracking purpose, IDs or other fixed numbers should be preprocessed and protected by keys before transmission. In other words, RSUs will only be able to collect keyed signatures of vehicles' IDs (KIDs). To measure the two-point traffic flow sizes, we introduce a family of COHFs, and propose two novel traffic flow measurement schemes, which can protect the identities of vehicles. The first scheme is more efficient and can achieve exact measurement result, but it is vulnerable to an identical-key attack. The second scheme prevents this attack at the cost of increased computation overhead. To make it practical, we adopt statistical methods with sampling to construct an MLE estimator for the traffic flow size. The sampling can control the tradeoff between the computation efficiency and the measurement accuracy. We perform simulations, and the results demonstrate the feasibility of our schemes.

Now we should ask the question: can we do better? As we look at the performance metrics of a traffic flow measurement scheme, we think that we may improve the two existing schemes from two directions. First, can we further improve the computation overhead? The enhanced scheme (which is free from the identical-key attack) achieves a constant computation overhead for each vehicle per RSU en route as well as for each RSU per passing vehicle,

which is efficient enough. However, as for the central server, the computation overhead for it to compute the two-point traffic flow size between a pair of RSUs is in the quadratic form regarding to the number of vehicles passing by each RSU, i.e., $O(N^2)$ assuming each RSU has $O(N)$ cars passing by during this measurement period. Can we further improve the efficiency of the central server? This is one direction that we may consider.

Another improvement direction is, can we further preserve the vehicles' privacy? The current two schemes clearly preserve the first-level privacy of vehicles, but what about the second-level privacy? In these two solutions, since a pair of double-hashed values represents a same vehicle, which is the foundation for recognizing the common vehicles passing two arbitrarily specified RSUs and measuring the two-point traffic flow sizes, the "traces" of vehicles are still revealed to some degree. Although the owners of those "traces" are normally not trackable, it may happen that the identity of a vehicle is accidentally revealed at some point. For example, the identity of a vehicle may be revealed by a photograph triggered by the vehicle rushing a red light or by a police car stopping the vehicle. When the identity is combined with the KID information transmitted by the vehicle, the central server can also discover the full traveling path of this vehicle through checking every KID set of every RSU for common double-hashed values. Despite the fact that the thorough-checking operation to recover a vehicle's full path is quite expensive in terms of computation overhead, the linkable "traces" still leave room for us to consider in more depth.

CHAPTER 4
PRIVACY-PRESERVING TWO-POINT TRAFFIC MEASUREMENT THROUGH FIXED-LENGTH BIT ARRAY MASKING

In this chapter, we present our third novel scheme for privacy-preserving two-point traffic measurement in CPRS [46] [47], which combines the beauty of a compact data structure, shared bit arrays, and a statistical MLE method [48]. We first discuss the motivation for us to change the perspective from using keyed signatures to utilizing shared bit arrays, then introduce this novel scheme based on bit array masking, and then analyze its performance through mathematical proof, numerical analysis, as well as extensive simulations. Finally, we conclude this chapter with a summary of this measurement scheme.

4.1 From Keyed Signatures to Bit Array Masking

Recall that the previous two schemes for two-point traffic measurement suffer from two aspects: First, the computation overhead for the central server, which increases quadratically with the single-point traffic volume of the involving locations, is not efficient enough to suit for today's large-scale road systems; Second, although the first-level privacy of vehicles is well preserved, the second-level privacy is not. The privacy level of the previous two schemes is limited by their measurement foundation: the "traces" of a vehicle must present themselves to enable identification of common vehicles, and those traces are linkable. More specifically, if a vehicle's identity is revealed at some location, say by a photograph triggered by the vehicle rushing a red light or by a police car stopping the vehicle, the central server can check the KID set from another location and determine for sure whether or not the vehicle has been in that place. The common-vehicle checking process is deterministic, which originates from the fact that a common double-hashed value represents a same vehicle. In other words, if a common double-hashed value presents in both locations, then the vehicle must have been in both places. On the other hand, if there is no KID present at some location that shares a common double-hashed value with the one exposed, then the vehicle must have not been in that place. One can imagine that this single-time exposed information can actually link all traces of the

vehicle (despite the expensive computation overhead incurred with common-vehicle checking), leading to a potential threat to the vehicle's overall privacy.

To achieve a higher-level privacy, we need to think about how to avoid this kind of single point of failure, and reconsider how to better "cover" a vehicle's traces and break the "link" between the traces of each individual vehicle. Intuitively, to better cover their traces, vehicles should add more "noise" for protection when they report their information; to break the link among the traces of a vehicle, it should make its traces "undistinguishable" from other vehicles' traces. This calls for two requirements. First, each vehicle should deliver a different (presumably "random" from others' perspective) message at a different location. This "internal noise" makes each single trace of a vehicle hard to find. Second, even if the information transmitted by an individual vehicle at different locations may look the same, all other vehicles should have the same chance to report the exact same information. This "external noise" makes the traces of this individual vehicle no different from those of other vehicles. Therefore, even if the adversary manages to find some pieces of traces (through rare situations as our previous example), it still won't be able to link other traces that belong to the same vehicle.

A measurement scheme satisfying the above two requirements will introduce two levels of protection to the vehicles in terms of their privacy: First, given a same piece of information present at two different locations, the outside world won't be able to determine for sure whether it is the same vehicle that reports the same information OR it is the same information that is reported by different vehicles; Second, given that a piece of information reported by a vehicle at some location is not present at another location, the outside world won't be able to determine for sure that the vehicle has definitely not been in that location. In other words, such a measurement scheme should make the common-vehicle checking process non-deterministic.

Given above thoughts, we are motivated to deal with the challenges from another perspective: instead of the deterministic measurement using keyed signatures based on computation-intensive COHFs, we now resolve the problem through a non-deterministic

statistic method based on some common shared information "pools", called the shared bit arrays. This transition combines the beauty of both shared bit arrays and the statistic MLE method, and brings three-folded benefit: (1) the simplicity of bit array operations provides a fundamental improvement on the computation efficiency; (2) the shared bit array masking further preserves both the first-level and second-level privacy of individual vehicles; (3) the estimator derived from the rigorous MLE method gracefully controls the measurement accuracy of the aggregate two-point traffic flow data. In this chapter, we will start developing such a measurement scheme from the new perspective.

4.2 Measurement Scheme Based on Bit Array Masking

We start with an overview of the novel scheme: It utilizes shared bit arrays to encode "masked" data (random indices in the shared bit arrays) sent from vehicles to RSUs, and adopts the MLE method to obtain measurement results based on the shared bit arrays. There are two phases for each measurement period, online coding and offline decoding. Online coding is an interaction between vehicles and RSUs, where "masked" data for traffic flow measurement are transmitted by the vehicles and securely collected by the RSUs. Later in the offline decoding phase, the central server will use the information collected by RSUs to compute traffic flow sizes. In the following, we first describe the two measurement phases, and then evaluate this scheme with respect to the three performance metrics described in Section 2.4.

4.2.1 Online Coding Phase

In this scheme, each RSU R_x maintains a counter n_x, which keeps track of the total number of vehicles passing by during the current measurement period. R_x also maintains a bit array B_x with a fixed length m ($m > 1$) to mask vehicle identities. At the beginning of each measurement period, n_x and all the bits in B_x are set to zeros. In addition, each vehicle v has a logical bit array LB_v, which consists of s ($1 < s < m$) bits randomly selected from B_x. The indices of these bits in B_x are $H(v \oplus K_v \oplus X[0])$,..., $H(v \oplus K_v \oplus X[s-1])$, where $\oplus$ is the bitwise XOR, $H(...)$ is a hash function whose range is $[0, m)$, X is an integer array of randomly

chosen constants whose purpose is to arbitrarily alter the hash result, and K_v is the private key of v whose purpose is to protect the privacy of its logical bit array.

The online coding phase is quite simple. RSUs broadcast queries in pre-set intervals (e.g., once a second), ensuring that each passing vehicle receives at least one query and meanwhile giving enough time for the vehicle to reply. Collisions can be resolved through well-established CSMA or TDMA protocols, which are not the focus of our design. Every query that an RSU sends out includes the RSU's RID and its public-key certificate. Suppose a vehicle, whose ID is v, receives a query from an RSU, whose ID is R_x. The vehicle first verifies the certificate, and then uses the RSU's public key to authenticate the RSU. After verifying that R_x is from the trustworthy authority, the vehicle v will randomly select a bit from its logical bit array LB_v by computing an index $b = H(v \oplus K_v \oplus X[H(R_x \oplus t) \, mod \, s])$, where t is the current time stamp. The vehicle v then sends the resulting index b to the RSU R_x. Upon receiving the index b, R_x will first increase its counter n_x by 1, and then set the bth bit in B_x to 1:

$$B_x[H(v \oplus K_v \oplus X[H(R_x \oplus t) \, mod \, s])] = 1. \tag{4–1}$$

4.2.2 Offline Decoding Phase

At the end of each measurement period, all RSUs will send their counters and bit arrays to the central server, which then performs the offline measurement. We employs the MLE method [48] to measure the sizes of traffic flows based on the counters and bit arrays.

Suppose the set of vehicles that pass RSU R_x (R_y) is denoted as S_x (S_y) with cardinality $|S_x| = n_x$ ($|S_y| = n_y$). Clearly, the set of vehicles that pass both RSU R_x and R_y is $S_x \cap S_y$. Denote its cardinality as n_c, which is the value that we want to measure. Furthermore, denote by S the subset of vehicles in $S_x \cap S_y$ that happen to set the same bit in B_x and B_y, where B_x and B_y are the bit arrays at R_x and R_y, respectively. Let n_o be the cardinality of S, i.e., $n_o = |S|$. Clearly, $S \subseteq S_x \cap S_y$ and $0 \leq n_o \leq n_c$. For any vehicle, it has the same probability $\frac{1}{s}$ to set any bit in its s-bit logical bit array. As a result, the probability for an arbitrary vehicle v from $S_x \cap S_y$ to select the same bit in both B_x and B_y is $s \times \frac{1}{s} \times \frac{1}{s} = \frac{1}{s}$. Therefore, the

number of such vehicles, n_o, is binomially distributed according to $B(n_c, \frac{1}{s})$. Accordingly, the probability for $n_o = z (0 \leq z \leq n_c)$ is

$$P(n_o = z) = \binom{n_c}{z} \left(\frac{1}{s}\right)^z \left(1 - \frac{1}{s}\right)^{n_c - z}. \tag{4–2}$$

Given the counters n_x and n_y, and bit arrays B_x and B_y, we measure n_c as follows: First, take a bitwise AND of B_x and B_y, and denote the resulting bit array as B_c. Namely,

$$B_c[i] = B_x[i] \wedge B_y[i], \ \forall i \in [0, m-1]. \tag{4–3}$$

We can easily find out the number of 0's in B_c. Suppose it is denoted by U_c. In the following, we will analyze the probability for an arbitrary bit in B_c to remain '0' after the online coding phase, and use it to establish the likelihood function for us to observe U_c '0' bits in B_c. Maximizing that likelihood function with respect to n_c will give the MLE estimator of n_c.

Clearly, the event for an arbitrary bit b in B_c to remain '0' after online coding is equivalent to the combination of the following two events: (1) *Event 1: None of the vehicles in S has chosen b at R_x and R_y.* If a vehicle $v \in S$ chooses b, then bit b in B_x and B_y are both set to '1' by v (hence bit b in B_c is also '1'). Since each vehicle has probability $\frac{1}{m}$ to set bit b to '1', the probability for the vehicle not to choose bit b is $1 - \frac{1}{m}$. There are n_o vehicles in S. Therefore, the probability for the first event to happen is

$$q_1 = \left(1 - \frac{1}{m}\right)^{n_o}. \tag{4–4}$$

(2) *Event 2: Either none of the vehicles in $S_x - S$ has chosen b at R_x or none of the vehicles in $S_y - S$ has chosen b at R_y.* Otherwise, bit b in both B_x and B_y will be '1' (hence bit b in B_c is '1'). The probability for bit b not chosen by any vehicle in $S_x - S$ is $(1 - \frac{1}{m})^{n_x - n_o}$, and the probability for bit b not chosen by any vehicle in $S_y - S$ is $(1 - \frac{1}{m})^{n_y - n_o}$. Therefore, the probability for the second event to happen is

$$
\begin{aligned}
q_2 &= 1-\left(1-\left(1-\frac{1}{m}\right)^{n_x-n_o}\right)\times\left(1-\left(1-\frac{1}{m}\right)^{n_y-n_o}\right) \\
&= \left(1-\frac{1}{m}\right)^{n_x-n_o}+\left(1-\frac{1}{m}\right)^{n_y-n_o}-\left(1-\frac{1}{m}\right)^{n_x+n_y-2\times n_o}. \qquad (4\text{–}5)
\end{aligned}
$$

Combining above analysis, the conditional probability for bit b in B_c to remain '0' given $n_o = z$ is $q_1 \times q_2$, namely,

$$
\begin{aligned}
q(n_c|n_o=z) &= q_1\times q_2 \\
&= \left(1-\frac{1}{m}\right)^{n_x}+\left(1-\frac{1}{m}\right)^{n_y}-\left(1-\frac{1}{m}\right)^{n_x+n_y-z}. \qquad (4\text{–}6)
\end{aligned}
$$

Given $q(n_c|n_o = z)$ and the distribution of n_o, the overall probability $q(n_c)$ for an arbitrary bit b in bit array B_c to remain '0' is

$$
\begin{aligned}
q(n_c) &= \sum_{z=0}^{n_c} q(n_c|n_o=z)\times P(n_o=z) \\
&= \sum_{z=0}^{n_c} q(n_c|n_o=z)\times\binom{n_c}{z}\left(\frac{1}{s}\right)^z\left(1-\frac{1}{s}\right)^{n_c-z} \\
&= \left(1-\frac{1}{m}\right)^{n_x}+\left(1-\frac{1}{m}\right)^{n_y}-\left(1-\frac{1}{m}\right)^{n_x+n_y}\left(\frac{\frac{1}{s}+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m}\right)}{1-\frac{1}{m}}\right)^{n_c}. \qquad (4\text{–}7)
\end{aligned}
$$

Knowing that each bit in B_c has a probability $q(n_c)$ to remain '0', we can establish the likelihood function for us to observe U_c '0' bits in B_c (hence $m - U_c$ '1' bits in B_c):

$$
\mathcal{L} = (q(n_c))^{U_c}\times(1-q(n_c))^{m-U_c}. \qquad (4\text{–}8)
$$

The MLE estimator of n_c is the optimal value of n_c that maximizes the likelihood function in (4–8), namely,

$$
\hat{n_c} = \arg\max_{n_c}\{\mathcal{L}\}. \qquad (4\text{–}9)
$$

To find $\hat{n_c}$, we take logarithm on both sides of (4–8):

$$\ln \mathcal{L} = U_c \times \ln q(n_c) + \big(m - U_c\big) \times \ln \big(1 - q(n_c)\big). \tag{4–10}$$

Take the first order derivative of (4–10), we have

$$\frac{d \ln \mathcal{L}}{dn_c} = \left(\frac{U_c}{q(n_c)} - \frac{m - U_c}{1 - q(n_c)} \right) \times q'(n_c), \tag{4–11}$$

where $q'(n_c)$ can be computed from (4–7) as follows,

$$\begin{aligned} q'(n_c) &= \frac{dq(n_c)}{dn_c} \\ &= -\left(1 - \frac{1}{m}\right)^{n_x + n_y} \left(\frac{\frac{1}{s} + \big(1 - \frac{1}{s}\big)\big(1 - \frac{1}{m}\big)}{1 - \frac{1}{m}} \right)^{n_c} \ln \left(\frac{\frac{1}{s} + \big(1 - \frac{1}{s}\big)\big(1 - \frac{1}{m}\big)}{1 - \frac{1}{m}} \right) \end{aligned} \tag{4–12}$$

To compute $\hat{n_c}$, we set the right side of (4–11) to 0:

$$\left(\frac{U_c}{q(n_c)} - \frac{m - U_c}{1 - q(n_c)} \right) \times q'(n_c) = 0. \tag{4–13}$$

Observe from (4–12) that $q'(n_c)$ cannot be 0 when $m > 1$ and $s > 1$. Therefore, we have

$$\frac{U_c}{q(n_c)} - \frac{m - U_c}{1 - q(n_c)} = 0. \tag{4–14}$$

Substituting (4–7) to (4–14), we obtain the MLE estimator $\hat{n_c}$ of the desired traffic flow size n_c as follows:

$$\hat{n_c} = \frac{\ln \left(\big(1 - \frac{1}{m}\big)^{n_x} + \big(1 - \frac{1}{m}\big)^{n_y} - \frac{U_c}{m} \right) - \big(n_x + n_y\big) \ln \big(1 - \frac{1}{m}\big)}{\ln \left(\frac{\frac{1}{s} + \big(1 - \frac{1}{s}\big)\big(1 - \frac{1}{m}\big)}{1 - \frac{1}{m}} \right)}. \tag{4–15}$$

4.2.3 Measurement Accuracy

In the following subsections, we evaluate the performance of this measurement scheme with respect to the three performance metrics described in Section 2.4. We start with analyzing the measurement accuracy of the MLE estimator $\hat{n_c}$. The standard theory of

MLE [60] states when m, n_x, and n_y are large enough, the MLE estimator $\hat{n}_c$ approximately follows the normal distribution:

$$\hat{n}_c \sim Norm\left(n_c, \frac{1}{\mathcal{I}(\hat{n}_c)}\right), \tag{4–16}$$

where $\mathcal{I}(\hat{n}_c)$ is the fisher information of $\mathcal{L}$, defined as:

$$\mathcal{I}(\hat{n}_c) = -E\left[\frac{d^2 \ln \mathcal{L}}{dn_c^2}\right]. \tag{4–17}$$

We compute the second-order derivative of $\ln \mathcal{L}$ from (4–11):

$$\begin{aligned} \frac{d^2 \ln \mathcal{L}}{dn_c^2} &= \left(-\frac{U_c \cdot q'(n_c)}{q^2(n_c)} - \frac{(m-U_c)\cdot q'(n_c)}{(1-q(n_c))^2}\right) \cdot q'(n_c) \\ &+\left(\frac{U_c}{q(n_c)} - \frac{m-U_c}{1-q(n_c)}\right) \cdot q'(n_c) \cdot \ln C, \end{aligned} \tag{4–18}$$

where $C = \frac{\frac{1}{s}+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m}\right)}{1-\frac{1}{m}}$ and $q'(n_c)$ is given in (4–12).

For an arbitrary bit b in B_c, it has probability $q(n_c)$ to remain '0'. U_c is the number of '0's in B_c. Therefore, U_c follows a binomial distribution $B(m, q(n_c))$. Accordingly,

$$E(U_c) = m \cdot q(n_c). \tag{4–19}$$

Substituting (4–18) and (4–19) to compute (4–17), we have

$$\begin{aligned} \mathcal{I}(\hat{n}_c) &= \left(\frac{m \cdot q'(n_c)}{q(n_c)} + \frac{m \cdot q'(n_c)}{1-q(n_c)}\right) \times q'(n_c) \\ &= \frac{m(q'(n_c))^2}{q(n_c)(1-q(n_c))}. \end{aligned} \tag{4–20}$$

According to (4–16), the variance of $\hat{n}_c$ is

$$Var(\hat{n}_c) = \frac{1}{\mathcal{I}(\hat{n}_c)} = \frac{q(n_c)(1-q(n_c))}{m(q'(n_c))^2}. \tag{4–21}$$

Therefore, the confidence interval of our measurement is

$$\hat{n}_c \pm Z_\alpha \times \sqrt{\frac{q(n_c)(1-q(n_c))}{m(q'(n_c))^2}}, \tag{4–22}$$

where α is the confidence level and Z_α is the α percentile for the standard Gaussian distribution [61]. For example, when $\alpha = 95\%$, $Z_\alpha = 1.6$.

4.2.4 Preserved Privacy

Next, we evaluate the preserved privacy of this measurement scheme. Note that in this scheme, the only information that a vehicle v ever transmits to an RSU en route is an index of a bit b randomly selected from its s-bit logical bit array, LB_v. Since the s bits in each vehicle's logical bit array are chosen randomly from the RSUs' physical bit arrays, from the adversary's point of view, every vehicle has the same probability to set any arbitrary bit of an RSU's bit array. In other words, the adversary cannot get the identity of a vehicle simply given its reported index. Therefore, the first-level privacy of each individual vehicle is clearly preserved.

We now focus on the second-level privacy that our scheme preserves. Again, since each vehicle just transmits a random bit index to each passing RSU, from the adversary's point of view, it can only identify the trace of a vehicle passing by two RSUs R_x and R_y through the observation of the bits that are set to '1' in both B_x and B_y; these bits will be '1' in B_c. Therefore, the second-level privacy of our scheme is actually a conditional probability which states to what degree an observed '1' in B_c does not represent a common vehicle passing by both R_x and R_y. We derive this conditional probability in the following.

Firstly, consider the probability for the adversary to observe an arbitrary bit, b, to be set to '1' in both B_x and B_y (event A), $P(A)$. Obviously, the probability $P(A)$ equals 1 minus $q(n_c)$ given our analysis in Section 4.2.2:

$$P(A) = 1 - \left(1-\frac{1}{m}\right)^{n_x} - \left(1-\frac{1}{m}\right)^{n_y} + \left(1-\frac{1}{m}\right)^{n_x+n_y}\left(\frac{\frac{1}{s}+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m}\right)}{1-\frac{1}{m}}\right)^{n_c}. \tag{4–23}$$

Secondly, consider the conditional probability for such a bit, b, to not represent a common vehicle passing both R_x and R_y (event E), $P(E|A)$. This is the second-level privacy p that we want to derive. Note that event E happens if and only if bit b in B_x is set only by vehicles passing only RSU R_x (i.e., in set $S_x - S_y$), and bit b in B_y is set only by vehicles passing only RSU R_y (i.e., in set $S_y - S_x$). Denote these two events as E_x and E_y, respectively. There are n_x (n_y) vehicles passing R_x (R_y), and n_c vehicles among them pass both R_x and R_y. Since each vehicle has a probability $\frac{1}{m}$ to set bit b to '1', the probability for E_x (E_y) to happen is:

$$P(E_x) = \left(1 - \left(1 - \frac{1}{m}\right)^{n_x - n_c}\right) \times \left(1 - \frac{1}{m}\right)^{n_c}, \tag{4–24}$$

$$P(E_y) = \left(1 - \left(1 - \frac{1}{m}\right)^{n_y - n_c}\right) \times \left(1 - \frac{1}{m}\right)^{n_c}. \tag{4–25}$$

Combining the above analysis, we have the formula for the preserved privacy of this scheme as follows:

$$\begin{aligned} p &= P(E|A) = \frac{P(E_x) \times P(E_y)}{P(A)} \\ &= \frac{\left(\left(1 - \frac{1}{m}\right)^{n_c} - \left(1 - \frac{1}{m}\right)^{n_x}\right) \times \left(\left(1 - \frac{1}{m}\right)^{n_c} - \left(1 - \frac{1}{m}\right)^{n_y}\right)}{P(A)}, \end{aligned} \tag{4–26}$$

where $P(A)$ is given in (4–23).

Observe that there are 2 parameters, s and m, that determine the value of $P(E|A)$. Among them, s only appears in the denominator $P(A)$, and it influences $P(E|A)$ through varying the value of $P(A)$. m influences both the denominator and the numerator. In the following, we first examine the influence of s on $P(A)$ (hence on $P(E|A)$), and then analyze how m affects the value of $P(E|A)$.

4.2.4.1 Influence of s on $P(A)$

To examine how s affects $P(A)$, we take partial derivative of (4–23) with respect to s:

$$\frac{\partial P(A)}{\partial s} = -\left(1 - \frac{1}{m}\right)^{n_x+n_y} \times \frac{n_c}{(m-1)s^2} C^{n_c-1}. \tag{4–27}$$

Recall that $C = \frac{\frac{1}{s}+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m}\right)}{1-\frac{1}{m}}$. Clearly, $\frac{\partial P(A)}{\partial s} < 0$. Therefore, with the increment of s, the value of $P(A)$ decreases, and in turn, the value of $P(E|A)$ increases. In other words, the preserved privacy will be better with a larger value of s. The numerical results are shown in Figure 4-1 where $n_x = n_y = n = 50,000$, $n_c = 5,000$, and $s = 2, 5, 10$, corresponding to three curves in each plot. Clearly, as s increases, the probability $P(A)$ decreases.

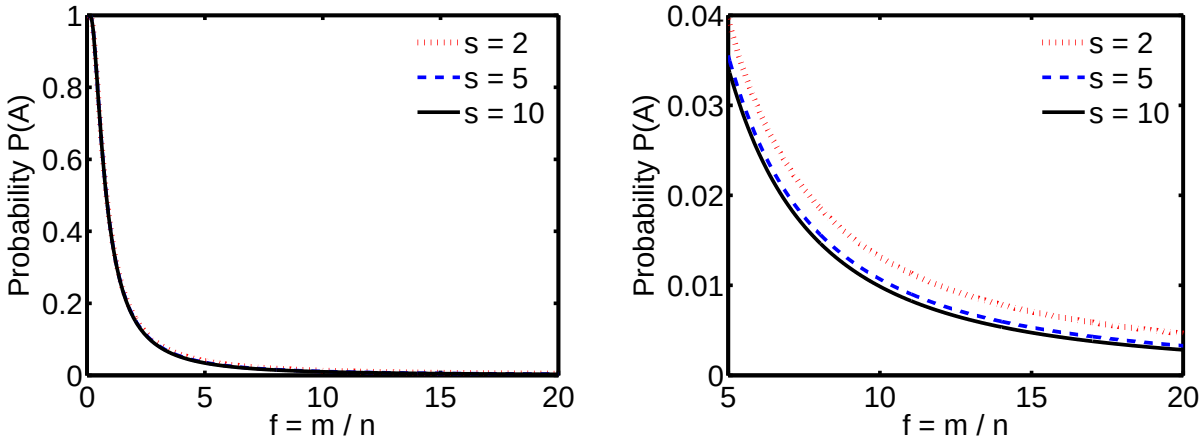


Figure 4-1. $n_x = n_y = n = 50,000$, $n_c = 5,000$; *Left Plot*: probability $P(A)$ when m varies from $0.1n$ to $20n$, controlled by different $s = 2, 5, 10$; *Right Plot*: a zoom-in of the left plot when m varies from $5n$ to $20n$.

Another observation from the numerical results is that when $s > 5$, the difference in probability $P(A)$ under different s becomes quite small. For instance, with $m \in [5n, 20n]$, the difference in $P(A)$ when $s = 5$ and $s = 10$ is smaller than 0.0005 (see the two lower curves in the right plot of Figure 4-1). When $n > 10$, that difference becomes negligible. Therefore, when we analyze the effect of m on $P(E|A)$ in the following subsection, and later when we set up the parameters for our simulations, we will only consider the cases of $s = 2, 5, 10$, with an established understanding that larger values of s will only make negligible difference.

4.2.4.2 Influence of m on $P(E|A)$

To examine the effect of m on $P(E|A)$, we take the partial derivative of (4–26) with respect to m and obtain the following:

$$\frac{\partial P(E|A)}{\partial m} = \frac{\frac{\partial P(E)}{\partial m} \times P(A) - \frac{\partial P(A)}{\partial m} \times P(E)}{P(A)^2}, \tag{4–28}$$

where $P(E) = P(E_x) \times P(E_y)$. $P(E_x)$ and $P(E_y)$ are given in (4–24) and (4–25), respectively. Therefore, the partial derivative of $P(E)$ with respect to m is:

$$\begin{aligned}\frac{\partial P(E)}{\partial m} &= \frac{1}{m(m-1)}\Bigg((n_x+n_y)\left(1-\frac{1}{m}\right)^{n_x+n_y} + 2n_c\left(1-\frac{1}{m}\right)^{2n_c} \\ &\quad - (n_c+n_x)\left(1-\frac{1}{m}\right)^{n_c+n_x} - (n_c+n_y)\left(1-\frac{1}{m}\right)^{n_c+n_y}\Bigg).\end{aligned} \tag{4–29}$$

In addition, from (4–23), we can compute the derivative of $P(A)$ with respect to m:

$$\begin{aligned}\frac{\partial P(A)}{\partial m} &= \frac{1}{m^2}\Bigg(-n_x\left(1-\frac{1}{m}\right)^{n_x-1} - n_y\left(1-\frac{1}{m}\right)^{n_y-1} \\ &\quad + \left(1-\frac{1}{m}\right)^{n_x+n_y-2} \times C^{n_c} \times \left((n_x+n_y)\left(1-\frac{1}{m}\right) - \frac{n_c}{s \times C}\right)\Bigg).\end{aligned} \tag{4–30}$$

We have proved that $\frac{\partial P(A)}{\partial m} < 0$, which means $P(A)$ will decrease with the increment of m. In addition, $\frac{\partial P(E)}{\partial m}$ will also be negative when m exceeds a certain value, which means $P(E)$ will also decrease with the increment of m afterwards. Intuitively, increasing m gives each vehicle a smaller chance $\frac{1}{m}$ to set an arbitrary bit, b. Hence, $P(E)$ and $P(A)$ also drop. The effect that m has on $P(E|A)$ is twofold: on the one hand, the increment of m decreases the denominator $P(A)$, which improves the privacy; on the other hand, the increment of m decreases the numerator $P(E)$, which reduces the privacy. With the combination of these two effects, the partial derivative of $P(E|A)$ with respect to m can be positive, negative, or 0,

according to (4–28). Therefore, given a value of s, we can choose an optimal m to achieve the best degree of privacy. The optimal m is obtained by setting the right side of (4–28) to 0.

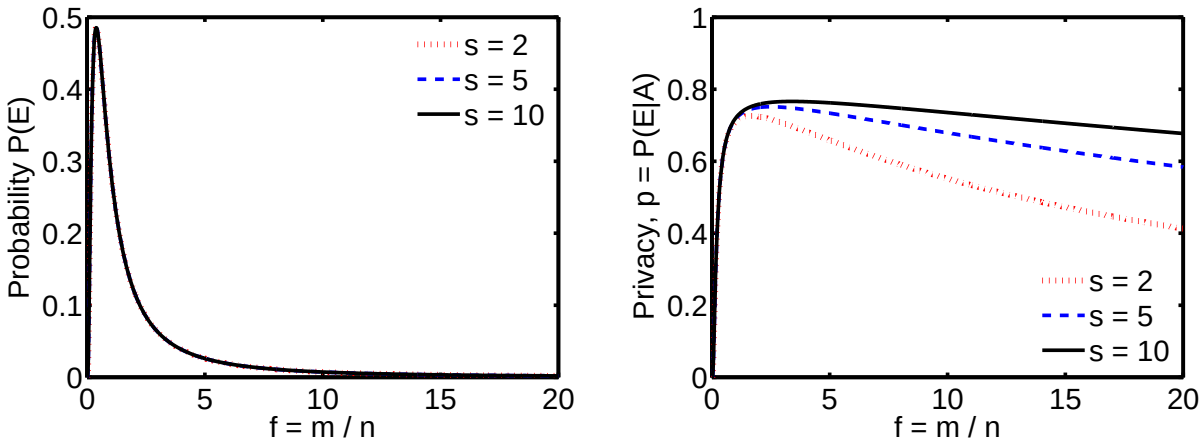


Figure 4-2. $n_x = n_y = n = 50,000$, $n_c = 5,000$. *Left Plot*: probability $P(E)$ when m varies from $0.1n$ to $20n$, under different $s = 2$, 5 or 10; *Right Plot*: probability $P(E|A)$ when m varies from $0.1n$ to $20n$, under $s =$2, 5 or 10.

Figure 4-2 shows the numerical results for the probability $P(E)$ and the second-level privacy $p = P(E|A)$ under different m when $n_x = n_y = n = 50,000$, $n_c = 5,000$, and $s = 2, 5, 10$. From the left plot, one can see that the three different values of s yield the same curve of $P(E)$ (or the three curves of $P(E)$ corresponding to $s = 2, 5, 10$ overlap completely). In other words, the value of s is irrelevant to the probability $P(E)$, which is consistent with our previous analysis. The value of m, on the other hand, has a clear impact on the value of $P(E)$. Specifically, there exists an optimal point where m^* (i.e., $f^* \times n$) produces a maximum value of $P(E)$. When $m < m^*$, the value of $P(E)$ increases dramatically with the increment of m. When $m > m^*$, the value of $P(E)$ decreases with a slower and slower pace. In the figure, $m^* = 0.39n$ results in an optimal value of $P(E) = 0.4856$. Recall from Figure 4-1 that the value of $P(A)$ always decreases with the increment of m. Combining these results, we learn that as m exceeds a certain value m^*, the probability $P(E)$ and $P(A)$ will both drop if we further increases m, which is also consistent to our theoretic analysis.

Finally, the right plot of Figure 4-2 gives the combined effect of s and m on $P(E|A)$, the second-level privacy of this scheme. The smallest value of $s = 2$ yields the bottom curve

that represents the least privacy, while the largest value of $s = 10$ yields the top curve that represents the best privacy, which agrees with our previous analysis that a larger value of s brings better privacy. Clearly, in each curve, $P(E|A)$ first increases quickly and then decreases slowly with respect to m. There is an optimal value of m that gives the optimal privacy. For instance, $m = 3.6n$ gives the optimal privacy 0.7661 when $s = 10$. Another observation is, when s is large (5 or 10), there always exists a smooth interval of m near its optimal point that can achieve near-optimal privacy. For example, when $s = 10$, the values of m in the interval $[3.6n, 11.2n]$ achieve privacy that is within 5% deviation from the optimal privacy 0.7661. In practice, this smooth interval allows us to adjust the value of m to achieve better measurement results while preserving near-optimal privacy.

4.2.5 Computation Overhead

We conclude the discussion about the performance of this measurement scheme by a quick remark on the computation overhead incurred to each group of entities involved in the system. In this scheme, when a vehicle v passes an RSU R_x, the vehicle v only needs to compute two hashes to obtain an index of a random bit in its logical bit array LB_v, and the RSU R_x only needs to set one bit in its bit array B_x, as described in Section 4.2.1. Therefore, the computation overhead for each vehicle per RSU as well as that for each RSU per vehicle are both $O(1)$. As for the central server, in order to compute the two-point traffic flow size between a pair of locations, it only needs to perform a bitwise AND operation over two m-bit arrays, count the number of '0's in the resulting bit array, and use formula (4–15) to compute the MLE estimator. Therefore, the computation overhead for the central server is $O(m)$. Recall that m is the size of each RSU's bit array.

4.3 Simulation

In this section, we evaluate the performance of this measurement scheme based on shared bit arrays through simulations. The simulation platform is a PC featured with an Intel Core i7-3770 CPU and 8GB RAM, running Windows 8 Pro, and the programs are written in C++. The simulations are performed under five system parameters, n_x, n_y, n_c, s, and m. For a

Table 4-1. Values for m to achieve optimal p under different s.

s	2	5	10
optimal m	$1.7n$	$2.6n$	$3.6n$
optimal p	0.7258	0.7513	0.7661

pair of RSUs, R_x and R_y, n_x (n_y) is the number of vehicles passing by R_x (R_y). There are n_c vehicles passing both R_x and R_y, which means the real two-point traffic flow size is n_c. s is the number of bits that each vehicle chooses in its logical bit array, and m is the number of bits in the RSUs' bit array. Our simulations consist of two parts. For each part, we first describe the settings of the system parameters, then report the simulation results, and finally discuss how the simulation results are related to our previous theoretic analysis.

4.3.1 Measurement of Traffic Flow Sizes

We first measure the two-point traffic flow sizes with respect to different settings of system parameters, and observe how different values of s influence the gap between the measured traffic flow sizes and the real traffic flow sizes when the optimal second-level privacy is preserved. We choose the five parameters as follows: $n_x = n_y = n = 50,000$, $100,000$, or $500,000$, and n_c varies from $1\%n$ to $50\%n$, with a step size of $0.1\%n$; $s = 2, 5, 10$, and m is chosen to achieve the optimal privacy p, as determined in Section 4.2.4. Table 4-1 lists the values for the bit array size m to achieve the optimal second-level privacy p under different values of s.

Figure 4-3, 4-4, and 4-5 show our simulation results when $n = 50,000$, $100,000$, and $500,000$, respectively. For each figure, there are three plots, corresponding to the results of three sets of simulations controlled by parameter s, where $s = 2$, 5, and 10. Each plot shows the measured two-point traffic flow sizes $\hat{n_c}$ (y-axis) with respect to different real two-point traffic flow sizes n_c (x-axis) under a given setting of n, s, and m, where m is chosen as described in Table 4-1 so that the optimal privacy is achieved. We also draw the equality line $y = x$ in each plot for reference. Clearly, the closer a point is to the equality line, the smaller the difference between the measured traffic flow sizes and the real traffic flow sizes, and in turn, the more accurate the measurement result.

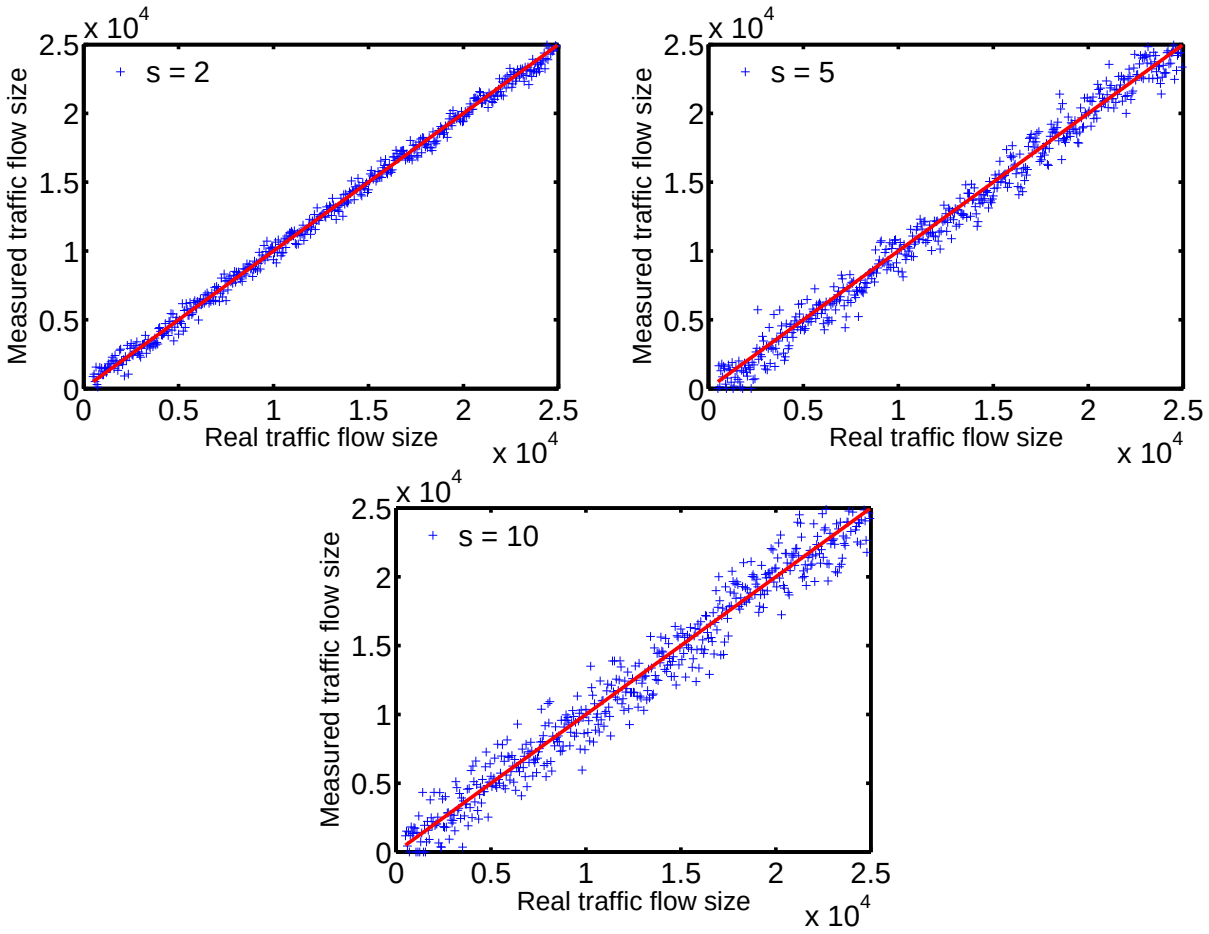


Figure 4-3. Measurement accuracy with the optimal privacy, $n_x = n_y = n = 50,000$, $n_c = [0.01n, 0.5n]$. The x-axis shows real two-point traffic flow sizes, and the y-axis shows the corresponding measured two-point traffic flow sizes. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

From the three figures, one can see that our measurement scheme is quite accurate because most of the points in all plots of the three figures lie closely to the equality line. In particular, given other parameters, our MLE estimator produces almost perfect results when $s = 2$ (the first plot in Figure 4-3, 4-4, and 4-5). When s becomes larger, there are slightly more points deviating from the equality line (the third plot in Figure 4-3, 4-4, and 4-5), which indicates larger values of s yield less accurate measurement results.

Recall that a larger value of s brings better privacy (Table 4-1). For example, the optimal privacy is 0.7661 when $s = 10$, better than the optimal privacy of 0.7258 when $s = 2$. This implies a tradeoff between the preserved privacy and the measurement accuracy. From

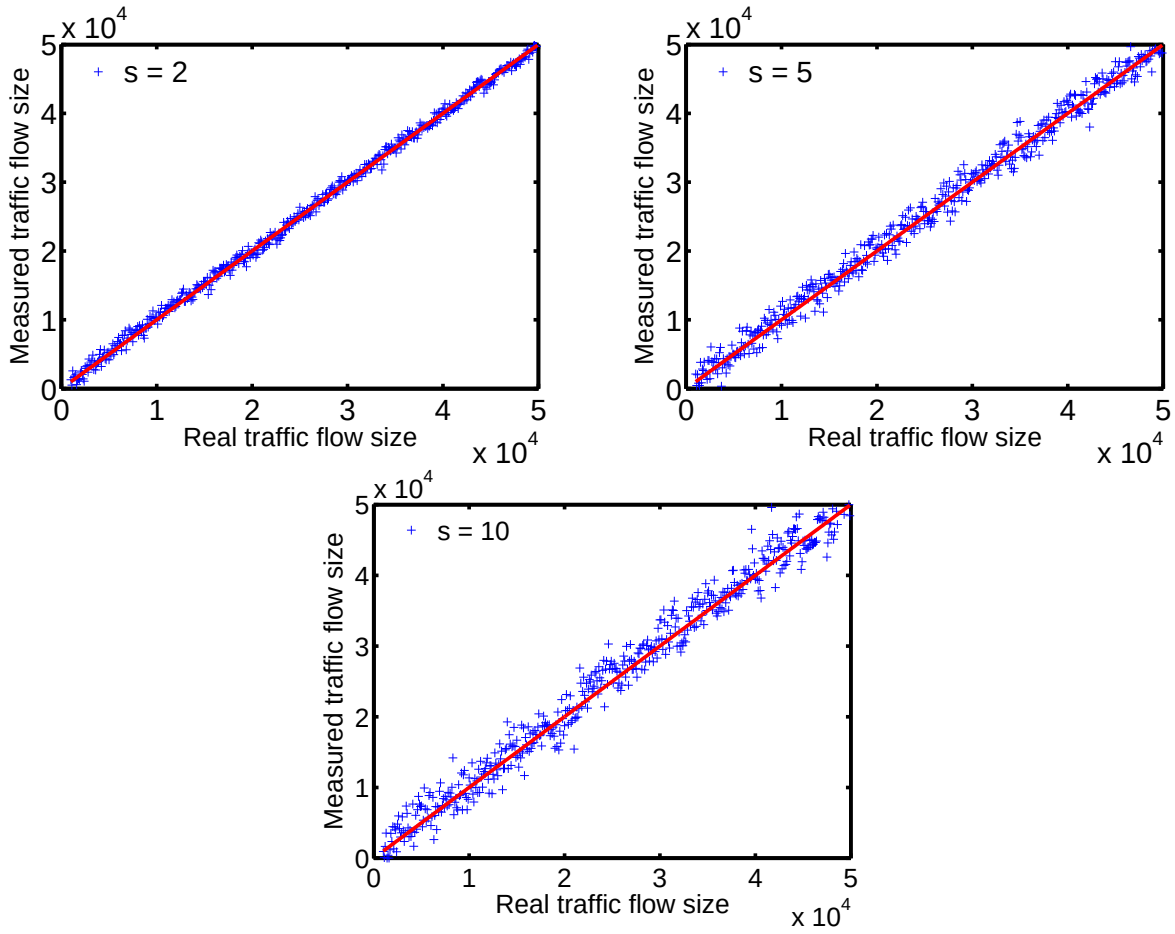


Figure 4-4. Measurement accuracy with the optimal privacy, $n_x = n_y = n = 100,000$, $n_c = [0.01n, 0.5n]$. The x-axis shows real two-point traffic flow sizes, and the y-axis shows the corresponding measured two-point traffic flow sizes. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

Section 4.2.4, we know when s is large, there always exists a smooth interval of m near its extreme point that can achieve comparable privacy as the optimal. For example, when $n_x = n_y = n = 50,000$, $n_c = 5,000$, and $s = 10$, the values of m within the interval $[3.6n, 11.2n]$ achieve privacy that is within just 5% drop of the optimal privacy 0.7661. In reality, one can choose a relatively large value for s (e.g., 5 or 10), and adjust the value of m to achieve better measurement results while still preserving comparable privacy as the optimal.

Finally, the measurement results are more accurate with larger values of n. There are fewer points deviating from the equality line $\hat{n_c} = n_c$ in the three plots of Figure 4-5 than those

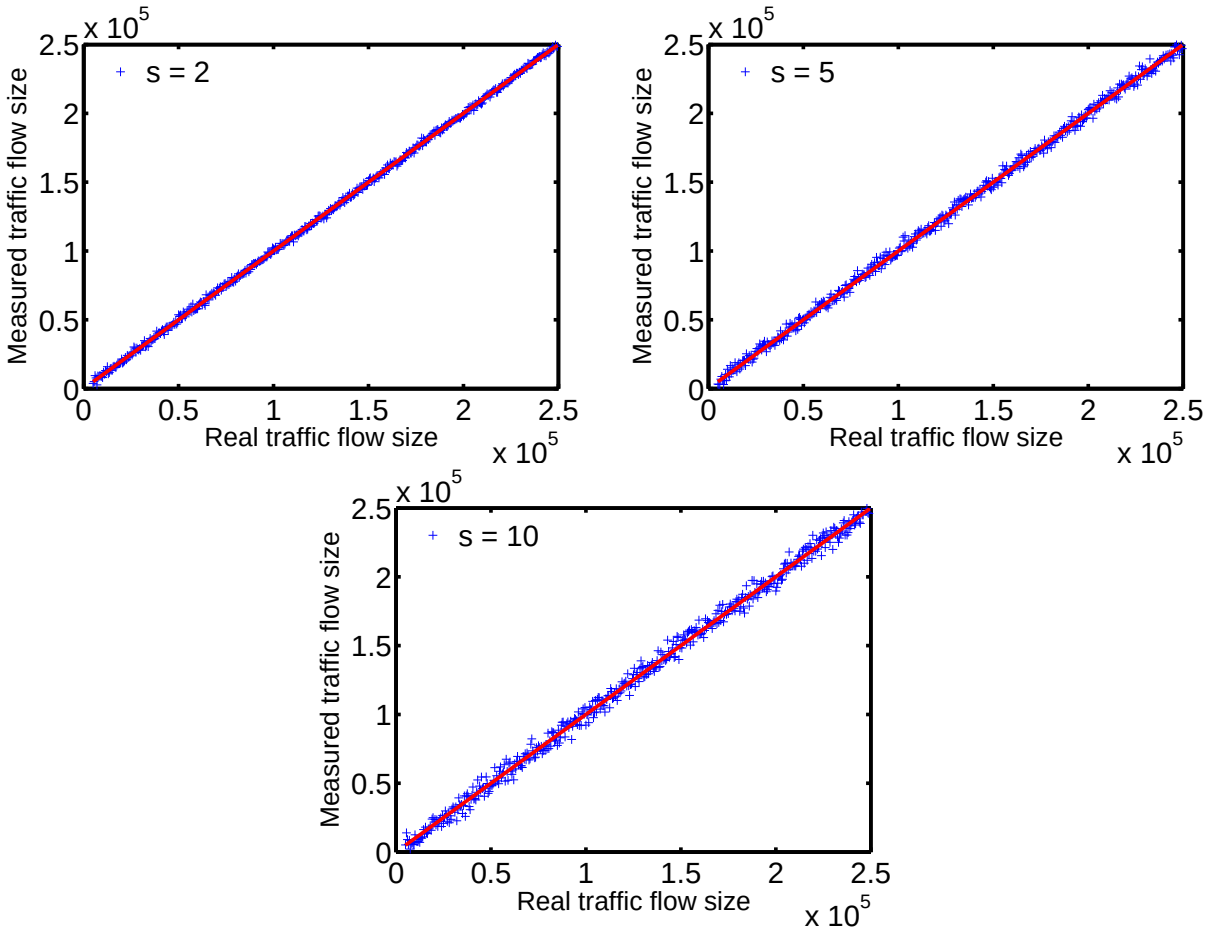


Figure 4-5. Measurement accuracy with the optimal privacy, $n_x = n_y = n = 500,000$, $n_c = [0.01n, 0.5n]$. The x-axis shows real two-point traffic flow sizes, and the y-axis shows the corresponding measured two-point traffic flow sizes. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

in the corresponding plots of Figure 4-3. This is also a natural phenomenon given that the measurement result is obtained through a statistical MLE estimator.

4.3.2 Measurement Bias and Relative Standard Error

Next, we study the measurement accuracy of the MLE estimator $\hat{n_c}$ in terms of bias and relative standard error. Similar to the previous part, there are three sets of simulations, each corresponding to $n_x = n_y = n = 50,000$, $100,000$, and $500,000$. For each set, there are three simulations controlled by different values of s, where $s = 2, 5, 10$. m is still chosen to achieve the optimal privacy p under each fixed s, as listed in Table 4-1. We conduct $5,000$ independent runs for each simulation to observe statistical effects. For each run, we randomly choose a

value for n_c from the range of $[0, 0.5n]$, and apply our measurement method to obtain the corresponding value for $\hat{n_c}$. Now, we try to figure out the measurement bias $E(\hat{n_c} - n_c)$ and relative standard error $\frac{\sqrt{Var(\hat{n_c})}}{n_c}$ of our MLE estimator from the result of the $5,000$ independent runs of each simulation.

To better illustrate the simulation results, we divide the range of n_c, $[0, 0.5n]$, into 50 measurement scales, each of width $1\%n$, and group the values of n_c and corresponding $\hat{n_c}$ from different runs into these 50 scales, and then numerically evaluate the measurement bias and relative standard error of the MLE estimator $\hat{n_c}$ with respect to each scale of n_c. The simulation results are presented in Figure 4-6 - Figure 4-11, where the first three figures (Figure 4-6, 4-7, 4-8) show the measurement bias and the remaining three figures (Figure 4-9, 4-10, 4-11) show the relative standard error.

Figure 4-6, 4-7, and 4-8 show the measurement bias of $\hat{n_c}$ with respect to each scale of n_c under different values of n, where $n = 50,000$, $100,000$, and $500,000$. Each figure consists of three plots, each corresponding to a fixed value of s, where $s = 2, 5, 10$. For each plot, the y-axis represents the measurement bias $E(\hat{n_c} - n_c)$, and the x-axis represents the mean value of n_c in each scale. The y-coordinate is within 2.5% of n, i.e., ranging from $-2.5\%n$ to $2.5\%n$. Note that the optimal privacy is always guaranteed for all simulations by setting m in accordance with s. From the figures, one can see that the measurement bias fluctuate around the zero-bias line for different scales of n_c. In addition, observed from the three plots of each figure, under a fixed n, the measurement bias tend to fluctuate more often with higher amplitudes for larger values of s (in particular, compare the first plot of Figure 4-6, 4-7, and 4-8 with the third plot of the same figures), which implies larger values of s will result in more $\hat{n_c}$ deviating from n_c, and in turn, yield less accurate measurement results. This observation agrees with our simulation results from the previous part. Furthermore, if we compare the plots from different figures (for instance, first plot of each figure), it is clear that under the same value of s, increasing the value of n will reduce the fluctuation amplitudes of $\hat{n_c}$, which

means our scheme will produce more stable and accurate measurement results for systems with relatively larger scales.

Figure 4-9, 4-10, and 4-11 show the relative standard error of $\hat{n_c}$ with respect to each scale of n_c under different values of n, where $n = 50,000$, $100,000$, and $500,000$. There are also three plots in each figure, each corresponding to a group of simulation results controlled by $s = 2$, $s = 5$, and $s = 10$, respectively. For each plot, the y-axis represents the relative standard error of $\hat{n_c}$, $\frac{\sqrt{Var(\hat{n_c})}}{n_c}$, and the x-axis represents the mean value of n_c in each scale. Still, optimal privacy is guaranteed through setting appropriate m. The major observation is that, given n, when s becomes larger, the relative standard error of $\hat{n_c}$ with respect to each scale of n_c also becomes larger. For instance, when $n = 50,000$, the relative standard error of $\hat{n_c}$ is about 0.017 for the scale of n_c ranging from $[8500, 9000]$ when $s = 2$, while its value reaches to about 0.13 when $s = 10$, almost 8 times higher than the former value. Since the relative standard error for each scale of n_c becomes larger, the variance for the MLE estimator also becomes larger, which means the measured traffic flow sizes will be more spread out from the real flow sizes. This observation also agrees with our previous simulation results, where there are relatively more points not close to the equality line for larger values of s under fixed n. Similarly, the variance becomes smaller when we increase the number n of vehicles in the system. One can observe that the relative standard error values are closer to 0 in Figure 4-11 than those in Figure 4-9, assuming the same value of s is applied.

4.4 Summary

In this chapter, we propose a third novel scheme for privacy-preserving two-point traffic flow measurement in CPRS. The proposed scheme utilizes a compact data structure, shared bit arrays, to automatically collect "masked" data from vehicles on road, and adopts the statistical MLE method to obtain the measurement result based on the shared bit arrays. The novel scheme achieves the following advantages:

First, in this novel scheme, the computation overhead for the central server to compute the two-point traffic flow size between a pair of RSUs is $O(m)$, where m is the size of each

RSU's bit array. In practice, the value of m is often related to and determined by the value of n, where n is the number of vehicles passing by each RSU. Usually, m is several times of n (recall Table 4-1). In other words, the computation overhead for the central server is actually $O(n)$. This is a huge improvement compared with the previous two schemes, which incurs $O(n^2)$ computation overhead for the central server to compute the size of the same two-point traffic flow. At the same time, the computation overhead for both vehicles and RSUs remains the same. The computation efficiency of our novel scheme is further improved given the simplicity of the bit array operations, comparing with the computation-intensive COHFs as deployed in the previous schemes. We have demonstrated through simulations that our novel scheme is so efficient that it can easily scale to large road systems.

Second, with our careful design of the shared bit array masking, vehicles' privacy is better preserved. In the previous two schemes based on keyed signatures, only the first-level privacy of vehicles is preserved; while our novel scheme based on shared bit arrays not only guarantees the first-level privacy, but also maintains a good second-level privacy. We have mathematically derived the formula for the second-level privacy of our novel scheme as a probability p, such that the probability for any "trace" of any vehicle to not be identified must be at least p. Through both mathematical and numerical analysis, we show the novel scheme can indeed maintain a good second-level privacy for vehicles.

Third, the measurement accuracy of this novel scheme can be gracefully controlled through the rigorous derivation of the MLE estimator. We have mathematically derived the formula for the measurement accuracy, and extensive simulations have been conducted to show that this novel scheme can indeed achieve sound measurement results and scale to large road systems.

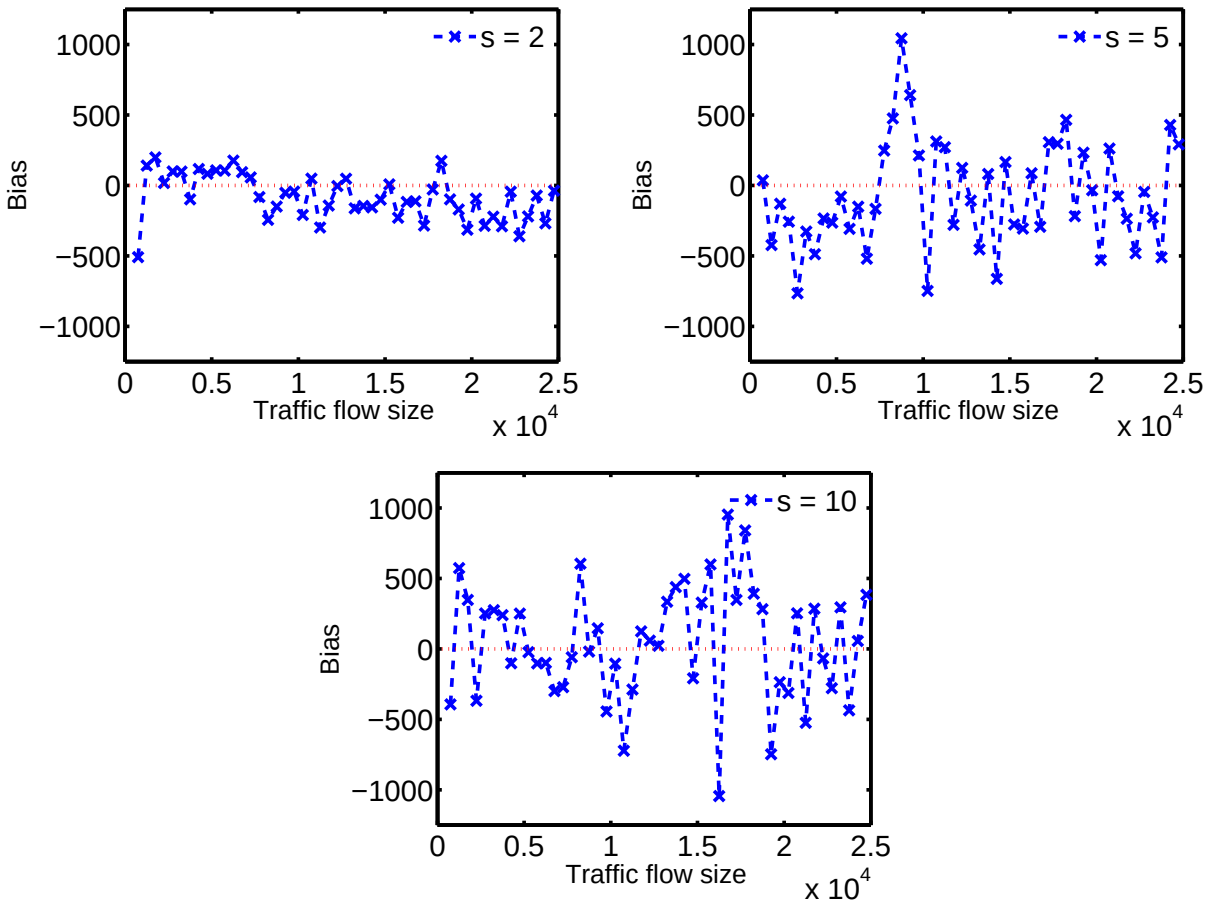


Figure 4-6. Measurement bias with the optimal privacy, $n_x = n_y = n = 50,000$, $n_c = [0, 0.5n]$. The x-axis shows scales of two-point traffic flow sizes n_c, and the y-axis shows the corresponding measurement bias $E(\hat{n}_c - n_c)$. The y-coordinate is within 2.5% of n, i.e.,$[-2.5\%n, 2.5\%n]$. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

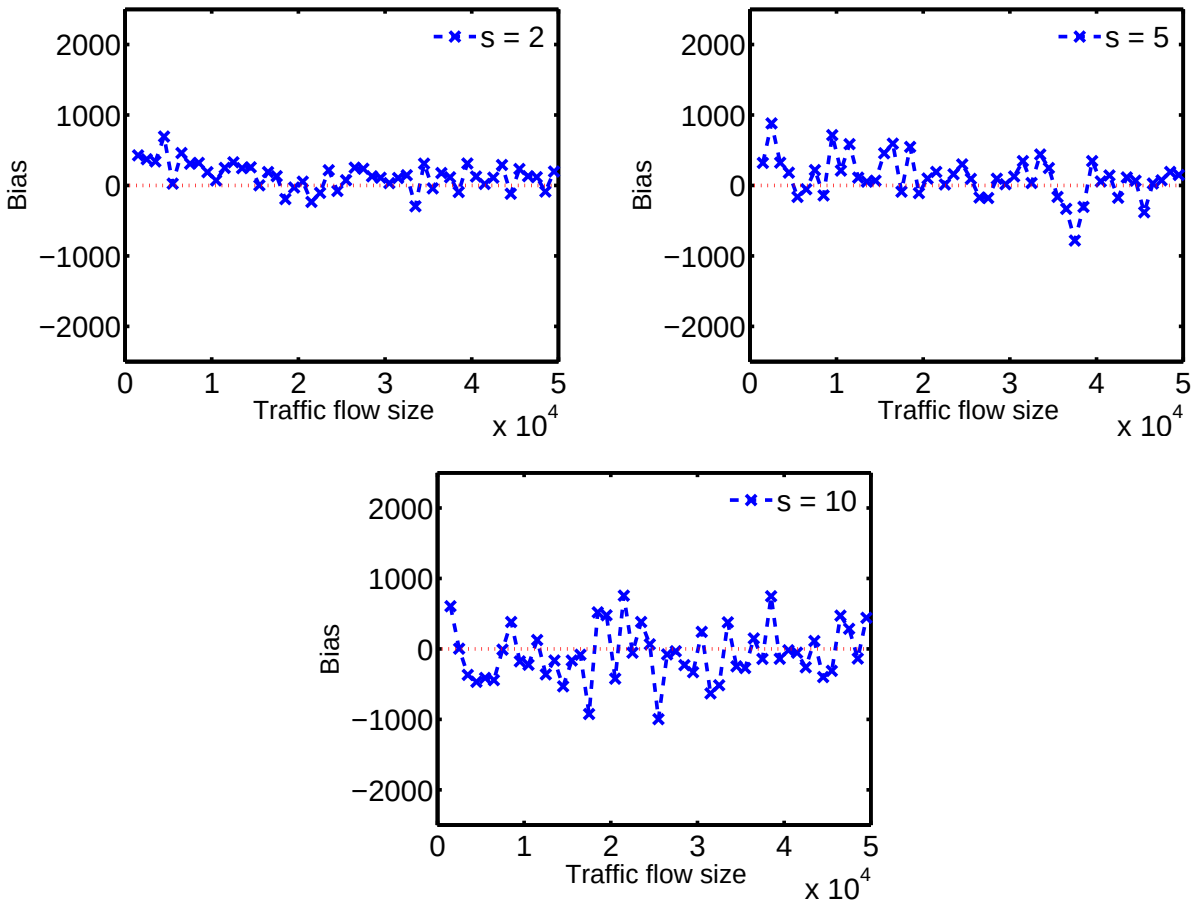


Figure 4-7. Measurement bias with the optimal privacy, $n_x = n_y = n = 100,000$, $n_c = [0, 0.5n]$. The x-axis shows scales of two-point traffic flow sizes n_c, and the y-axis shows the corresponding measurement bias $E(\hat{n_c} - n_c)$. The y-coordinate is within 2.5% of n, i.e.,$[-2.5\%n, 2.5\%n]$. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

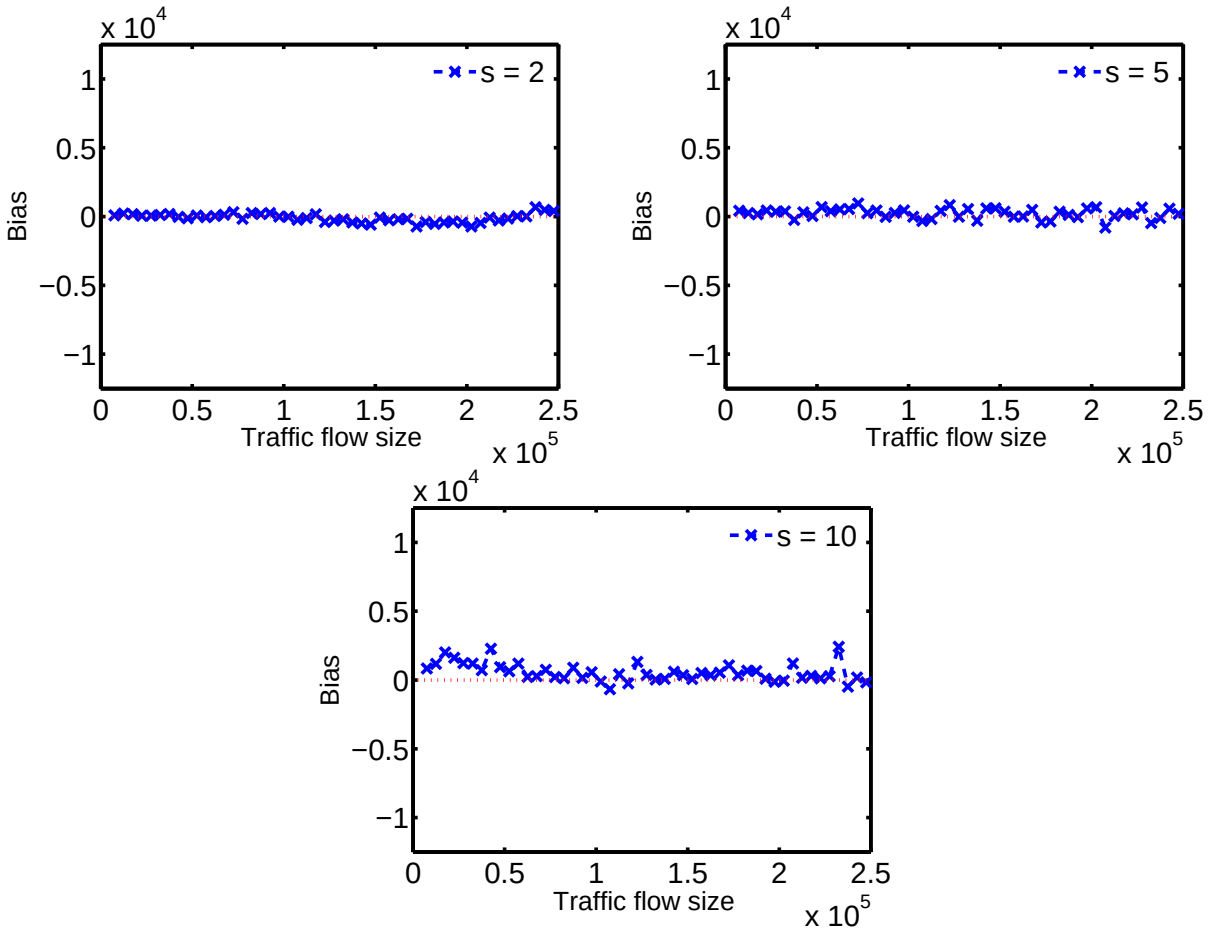


Figure 4-8. Measurement bias with the optimal privacy, $n_x = n_y = n = 500,000$, $n_c = [0, 0.5n]$. The x-axis shows scales of two-point traffic flow sizes n_c, and the y-axis shows the corresponding measurement bias $E(\hat{n_c} - n_c)$. The y-coordinate is within 2.5% of n, i.e.,$[-2.5\%n, 2.5\%n]$. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

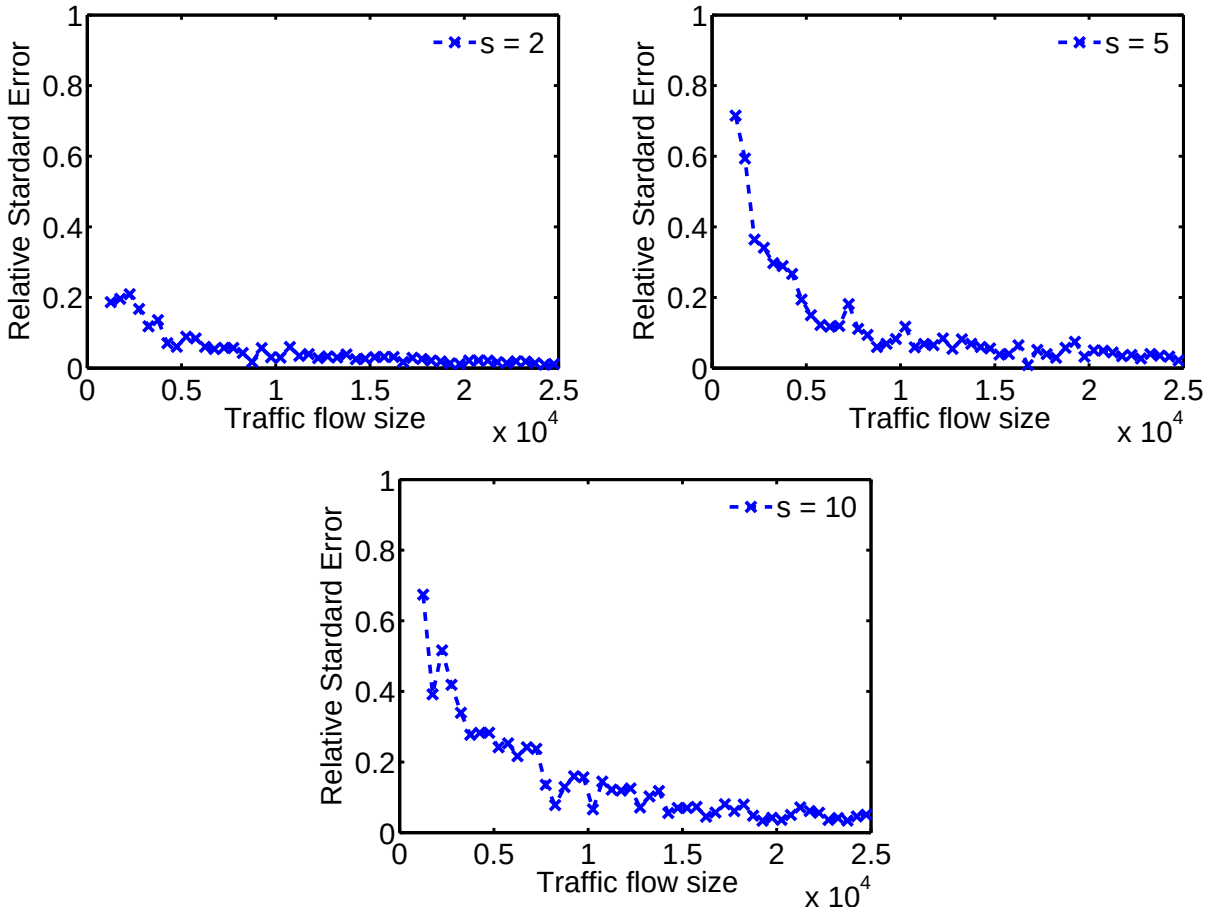


Figure 4-9. Measured relative standard error with the optimal privacy, $n_x = n_y = n = 50,000$, $n_c = [0, 0.5n]$. The x-axis shows scales of two-point traffic flow sizes n_c, and the y-axis shows the corresponding relative standard error $\frac{\sqrt{Var(\hat{n_c})}}{n_c}$. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

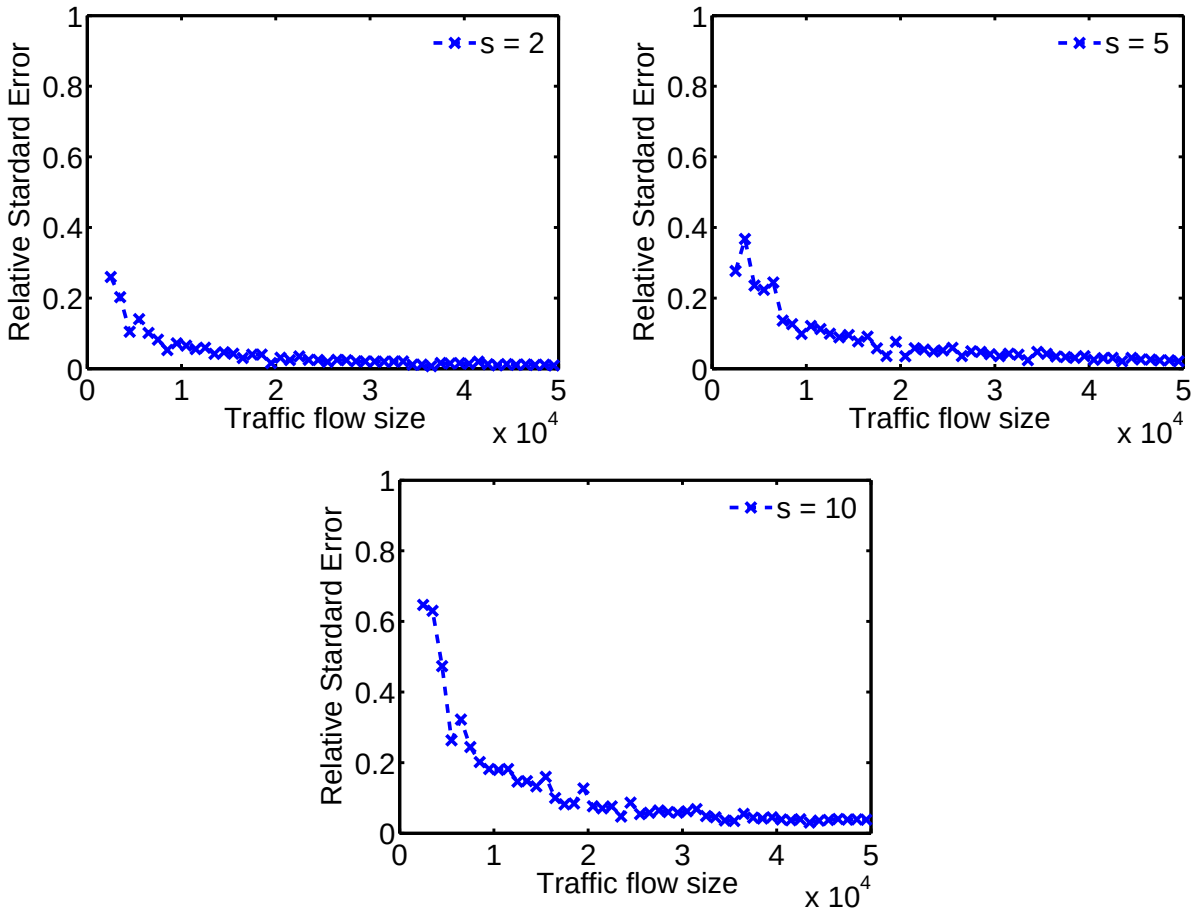


Figure 4-10. Measured relative standard error with the optimal privacy, $n_x = n_y = n = 100,000$, $n_c = [0, 0.5n]$. The x-axis shows scales of two-point traffic flow sizes n_c, and the y-axis shows the corresponding relative standard error $\frac{\sqrt{Var(\hat{n_c})}}{n_c}$. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

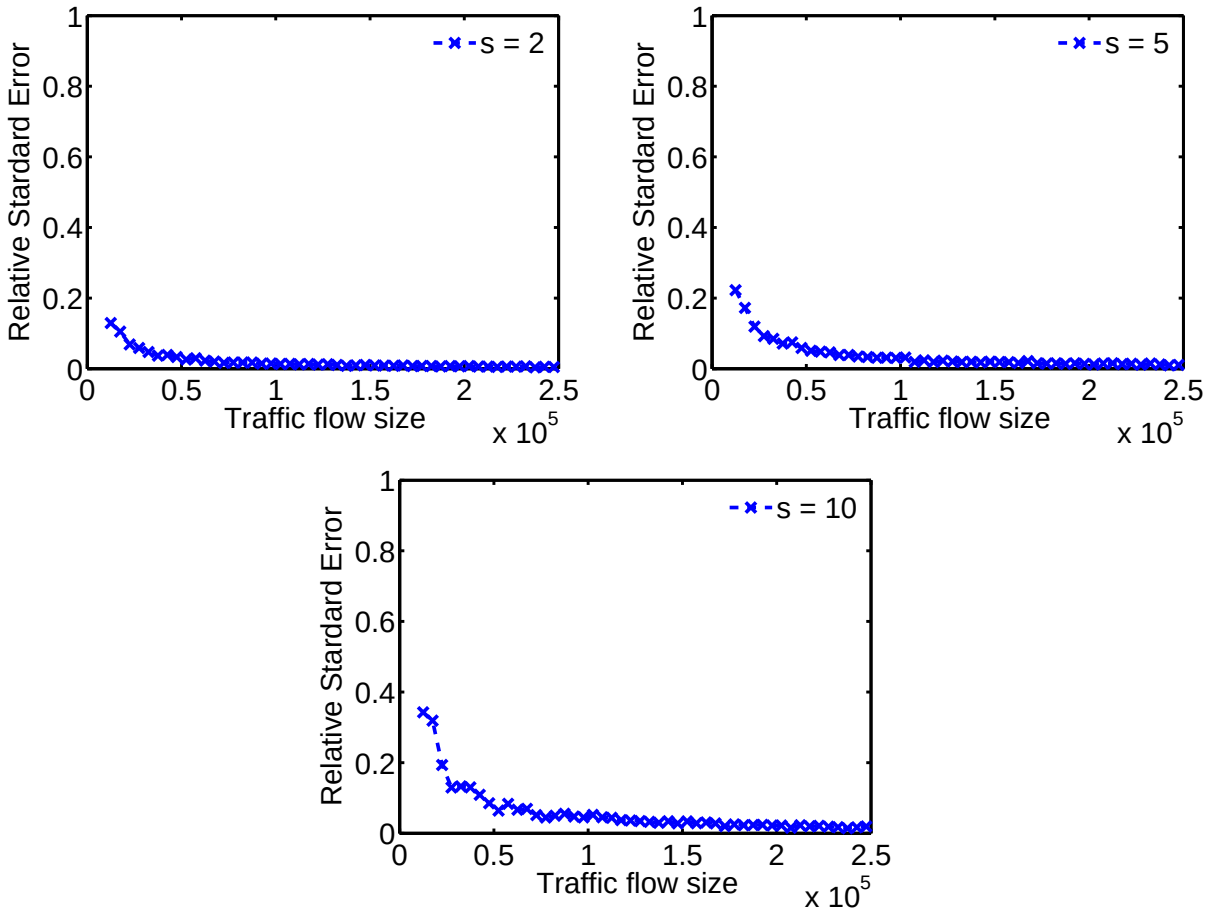


Figure 4-11. Measured relative standard error with the optimal privacy, $n_x = n_y = n = 500,000$, $n_c = [0, 0.5n]$. The x-axis shows scales of two-point traffic flow sizes n_c, and the y-axis shows the corresponding relative standard error $\frac{\sqrt{Var(\hat{n_c})}}{n_c}$. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

CHAPTER 5
PRIVACY-PRESERVING TWO-POINT TRAFFIC MEASUREMENT THROUGH VARIABLE-LENGTH BIT ARRAY MASKING

So far, the problem of privacy-preserving two-point traffic flow measurement in CPRS has been partially solved by [20] and our previous schemes [45] [46] [47]. [20] tries to infer two-point statistics from point data, but its practicability is limited by high computation overhead. Our first two schemes [45] use keyed signatures based on COHFs to protect vehicles' identities. The computation efficiency is improved to $O(n_x n_y)$ for each pair of RSUs, where n_x and n_y denote the number of vehicles passing them, respectively. This is better than [20], but the overhead is still too high for today's large-scale road networks. Moreover, it cannot protect vehicles' traces (second-level privacy). Motivated by [28] and [62], we propose a third scheme [46] [47] which further improves the computation efficiency to $O(n_x + n_y)$ and protects vehicles' identities as well as their traces, through the design of shared bit arrays. But it makes an unrealistic assumption about traffic similarity, and uses bit arrays of equal length at different RSUs to encode the passing vehicles, such that the bit arrays from two RSUs can be bitwise compared to extract a statistical result for two-point traffic volume. The scheme works great when all RSUs observe comparable numbers of vehicles. However, in reality, the traffic volume at different RSUs varies a lot. For example, according to the 2012 yearly traffic volume report from the New York State Department of Transportation [63], major intersections in New York have hundreds of thousands of cars passing by every day, while light-traffic intersections only have a few hundreds of cars passing by during the same period. Considering this more realistic situation where different RSUs observe varied amount of traffic, the performance of [47] decreases dramatically in terms of both vehicle privacy and measurement accuracy, which therefore limits its practicability.

As a continuous effort in improving the efficiency, privacy and accuracy, in this chapter, we propose our fourth novel scheme [49] for privacy-preserving two-point traffic flow measurement, which is an extension of our previous scheme [47] to remove the similar traffic assumption. The extension design utilizes variable-length bit arrays to encode traffic data reported by

vehicles, and a novel "unfolding" technique to support traffic flow measurement based on those variable-length bit arrays. Through mathematical and numerical analysis as well as extensive simulations, we demonstrate that the extension scheme based on variable-length bit arrays [49] has comparable efficiency with the previous scheme based on fixed-length bit arrays [47] and furthermore, it can easily fit in the more realistic transportation model where different RSUs observe varied amount of traffic, and achieve far better privacy and accuracy than the previous scheme. In the following, we first introduce the extension design, then analyze its performance, and compare it with the previous best scheme [47]. Finally, a summary of the extension design is given to conclude this chapter.

5.1 From Fixed-Length Bit Arrays to Variable-Length Bit Arrays

The previous scheme [47] uses fixed-length bit arrays to encode traffic data, and it works great when the single-point traffic volume of all RSUs are comparable. When it comes to the more realistic situation where the number of cars passing by different RSUs actually varies a lot, its performance decreases dramatically. The problem lies in the great difficulty of determining an appropriate fixed bit array size, m. If a large m is chosen to accommodate the large traffic volume in major intersections, it will greatly hurt the privacy for the cars passing by light-traffic RSUs (will explain more later in Section 5.4). If a small m is chosen to provide relatively good privacy for all cars, the accuracy for measuring the two-point traffic flow sizes between heavy-traffic RSUs will be dramatically decreased (will explain more later in Section 5.5).

Can one achieve the goal of both obtaining sound measurement results and maintaining good privacy for all cars? The solution with fixed-length bit arrays is not applicable, so how about variable-length bit arrays? Intuitively, to maintain good privacy, light-traffic RSUs should have smaller bit arrays, and to achieve sound measurement results, heavy-traffic RSUs should have larger bit arrays. In other words, the sizes of bit arrays should be related to the single-point traffic volume of the corresponding RSUs. This motivates our extension design based on variable-length bit arrays.

5.2 Measurement Scheme Based on Variable-Length Bit Array Masking

In this section, we introduce our extension design of privacy-preserving two-point traffic flow measurement scheme based on variable-length bit array masking. The key difference is that, in the extension design, different RSUs will use different-length bit arrays to collect vehicle's information. In principle, the size of an RSU's bit array is related to its single-point traffic volume, such that heavy-traffic RSUs will have larger bit arrays than light-traffic RSUs. To enable comparison of variable-length bit arrays, we propose an "unfolding" technique, and require the size of all bit arrays to be power of 2, i.e., in the form of 2^k.

The extension scheme also consists of two phases: online coding phase for storing de-identified vehicle information in variable-length bit arrays of RSUs, and offline decoding phase for measuring the two-point traffic flow sizes between pairs of RSUs using the MLE method based on the reported bit arrays. We will first describe the two phases in the proposed scheme, and then mathematically derive the MLE estimator used to measure two-point traffic flow sizes. The computation overhead is analyzed at the end of this section, and the other two performance metrics, measurement accuracy and preserved privacy, will be discussed in the next two sections.

5.2.1 Online Coding Phase

In the extension scheme, each RSU R_x maintains a counter n_x, which keeps track of the total number of passing vehicles during the current measurement period. R_x also maintains a bit array B_x with length m_x to mask vehicle identities. We require the lengths of all bit arrays to be power of 2, i.e., m_x must be in the form of 2^k, to facilitate the comparisons of variable-length bit arrays (more explanation later). We set the value of m_x to be $m_x = 2^{\lceil \log_2(\bar{n}_x \times \bar{f}) \rceil}$, where $\bar{n}_x$ is the expected traffic at R_x during the measurement period based on historical average traffic at the same location and the same time, and $\bar{f}$ is a system wide parameter whose value affects the tradeoff between measurement accuracy and level of privacy. Clearly, m_x is the smallest integer that is power of 2 and no less than $\bar{n}_x \times \bar{f}$. At the beginning of each measurement period, n_x and all bits in B_x are set to zeros.

Each vehicle v has a logical bit array LB_v, which consists of s bits randomly selected from an imaginary array B_* whose size m_* equals that of the largest bit array among all RSUs, where $s \ll m_*$. The indices of these bits in B_* are $H(v \oplus K_v \oplus X[0])$, ..., $H(v \oplus K_v \oplus X[s-1])$, where $\oplus$ is the bitwise XOR, $H(...)$ is a hash function whose range is $[0, m_*)$, X is an integer array of randomly chosen constants to arbitrarily alter the hash result, and K_v is the private key of v to protect its privacy.

Given above notations and data structures, the online coding phase works as follows. RSUs broadcast queries in pre-set intervals (e.g., once a second), ensuring that each passing vehicle receives at least one query and meanwhile giving enough time for the vehicle to reply. Collisions can be resolved through well-established CSMA or TDMA protocols, which are not the focus of this work. Every query that an RSU sends out includes the RSU's RID, its public-key certificate, and the size of its bit array. Suppose a vehicle, whose ID is v, receives a query from an RSU, whose ID is R_x and bit array size is m_x. It first verifies the certificate to authenticate the RSU. After verifying that R_x is from the trustworthy authority, v will randomly select a bit from its logical bit array LB_v by computing an index $b = H(v \oplus K_v \oplus X[H(R_x \oplus t) \, mod \, s])$, where t is the current time stamp. Then v generates an index b_x in the range of $[0, m_x)$ corresponding to b by a modulus operation, where $b_x = b \, mod \, m_x$, and sends b_x to R_x. Upon receiving the index b_x, R_x will first increase its counter n_x by 1, and then set the b_xth bit in B_x to 1. Therefore, the overall effect that v produces on R_x is:

$$n_x = n_x + 1, \tag{5–1}$$

$$B_x[H(v \oplus K_v \oplus X[H(R_x \oplus t) \, mod \, s]) \, mod \, m_x] = 1. \tag{5–2}$$

Note that the same vehicle may transmit different bit indices at two RSUs. The probability for this to happen is $1 - \frac{1}{s}$, which is larger when the size of LB_v is larger. Different vehicles may send the same index because their logical bit arrays share bits from B_x. As any vehicle does not have to transmit any fixed number in support of traffic measurement, we

improve privacy protection. This is true even when there is a single vehicle passing through two RSUs.

5.2.2 Offline Decoding Phase

At the end of each measurement period, all RSUs will send their counters and bit arrays to the central server, which first updates the history average single-point traffic volume for the RSUs to take into account the traffic data in the current measurement period, and then measures the two-point traffic volume between two arbitrary RSUs based on the reported counters and bit arrays.

Suppose the set of vehicles that pass RSU R_x (R_y) is denoted as S_x (S_y) with cardinality $|S_x| = n_x$ ($|S_y| = n_y$). Clearly, the set of vehicles that pass both RSU R_x and R_y is $S_x \cap S_y$. Denote its cardinality as n_{xy}, i.e., $|S_x \cap S_y| = n_{xy}$, which is the value that we want to measure. Denote the size of the bit array B_x (B_y) stored in RSU R_x (R_y) as m_x (m_y). Without loss of generality, we assume that $m_x \leq m_y$ (otherwise, change the role of R_x and R_y). Given the counters n_x and n_y, and bit arrays B_x and B_y, the server measures n_{xy} as follows:

First, our previous work [47] shows that when two bit arrays have the same length, we are able to combine them through bitwise operation and produce a good estimate for two-point traffic. Now we have to deal with two bit arrays of different lengths. In order to combine the information of the two arrays through bitwise operation, the central server expands the smaller bit array B_x to the same size of B_y through a process called **"unfolding"**, which is simply duplicating B_x multiple times until it reaches the size of B_y. Because m_x and m_y are both powers of 2 and $m_x \leq m_y$, it will always be true that m_y is divisible by m_x, which means that we can unfold B_x to the size of B_y by duplicating B_x for $\frac{m_y}{m_x}$ times. (When we derive the new formula for estimating the two-point traffic volume, we will mathematically account for the impact of duplication.) The "unfolded" bit array of B_x is denoted as B_x^u. Specifically,

$$B_x^u[i] = B_x[i \bmod m_x],\ \forall i \in [0, m_y). \tag{5–3}$$

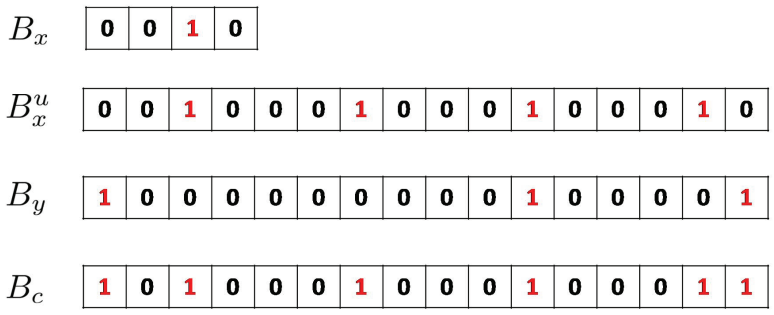


Figure 5-1. An example of unfolding and bitwise-OR operation.

Second, the server takes a bitwise OR operation on B_x^u and B_y to obtain a new bit array B_{xy}:

$$B_{xy}[i] = B_x^u[i] \vee B_y[i], \ \forall i \in [0, m_y). \tag{5–4}$$

The bitwise OR operation is granted since the two bit arrays, B_x^u and B_y, are of the same size. Clearly, through requiring the size of all bit arrays to be power of 2, we facilitate the comparison of variable-length bit arrays: the overall computation overhead to compare B_x and B_y is just $O(m_y)$, as contrast to $O(m_x \times m_y)$ without the "power of 2" requirement. Fig. 5-1 shows an example of unfolding and bitwise OR operation. In this example, B_x is unfolded to B_x^u, and a bitwise-OR is performed on B_x^u and B_y to produce B_{xy}.

Finally, **given B_{xy}, B_x (B_x^u), and B_y, the central server uses the following formula to estimate the two-point traffic flow size between R_x and R_y:**

$$\hat{n}_{xy} = \frac{\ln(V_{xy}) - \ln(V_x) - \ln(V_y)}{\ln(1 - \frac{s-1}{s} \times \frac{1}{m_y}) - \ln(1 - \frac{1}{m_y})} \tag{5–5}$$

where V_{xy}, V_x, and V_y are random variables (R.V.) which represent the fraction of bits whose values are zeros in B_{xy}, B_x, and B_y, correspondingly. Their values can be easily found by counting the number of zeros in B_{xy}, B_x, and B_y, denoted by U_{xy}, U_x, and U_y respectively (note U_{xy}, U_x, and U_y are also R.V.s), and dividing them by the corresponding bit array size m_y, m_x, and m_y. That is, $V_{xy} = \frac{U_{xy}}{m_y}$, $V_x = \frac{U_x}{m_x}$, and $V_y = \frac{U_y}{m_y}$. Note that the fraction of zero bits in B_x^u is the same as B_x.

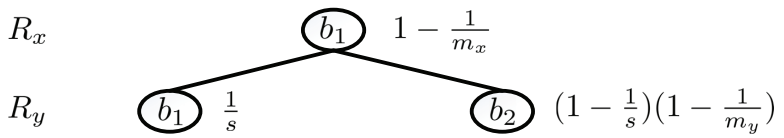


Figure 5-2. Decision tree for an arbitrary bit b in B_{xy} to remain '0' after a car v passing by both RSUs R_x and R_y (i.e., $v \in S_x \cap S_y$) sets bits in the two bit arrays (B_x and B_y). The number inside each node represents the index that v chooses for the corresponding RSU, and the math formula next to the node represents the probability for v to choose that index, given the condition that all ancestor nodes have been chosen.

5.2.3 Derivation of the MLE Estimator $\hat{n}_{xy}$

Now we follow the standard MLE method [48] to derive $\hat{n}_{xy}$ given by (5–5). Its accuracy will be analyzed in Section 5.3. We first derive the probability $q(n_{xy})$ for an arbitrary bit in B_{xy} to be '0', and use $q(n_{xy})$ to establish the likelihood function $\mathcal{L}$ to observe U_{xy} '0' bits in B_{xy}. Finally, maximizing $\mathcal{L}$ with respect to n_{xy} will lead to the MLE estimator, $\hat{n}_{xy}$.

Consider an arbitrary bit b in B_{xy}. Let A_b be the event that the bth bit in B_{xy} remains '0', then $q(n_{xy})$ is the probability for A_b to occur. Since the set of all vehicles passing R_x and/or R_y (i.e., $S_x \cup S_y$) can be partitioned into three sets, $S_x \cap S_y$, $S_x - S_y$, and $S_y - S_x$, it is clear that event A_b is equivalent to the combination of the following three events:

(I) *Event E_1: For vehicles passing both R_x and R_y (i.e., in the set $S_x \cap S_y$), none of them have chosen bit $(b \, mod \, m_x)$ in B_x or bit b in B_y.* Otherwise, according to (5–3) and (5–4), bit b in B_{xy} will be '1'. Fig. 5-2 shows the decision tree for a vehicle to not set the bits. For any vehicle, it has the same probability $\frac{1}{s}$ to select any bit in its s-bit logical bit array. So the probability for an arbitrary vehicle v from $S_x \cap S_y$ to select the same bit from its logical bit array in both R_x and R_y is $s \times \frac{1}{s} \times \frac{1}{s} = \frac{1}{s}$. In other words, if v chooses $(b' \, mod \, m_x) \neq (b \, mod \, m_x)$ in R_x, it has a probability of $\frac{1}{s}$ to choose the same bit b' in R_y (hence will not set bit b in B_y), and probability of $1 - \frac{1}{s}$ to choose a separate bit b'' randomly from B_y. The probability for $b'' \neq b$ is $1 - \frac{1}{m_y}$, and the probability for v to choose $(b' \, mod \, m_x)$

in B_x is $1-\frac{1}{m_x}$. There are n_{xy} cars in set $S_x \cap S_y$, so the probability of E_1 is

$$\begin{aligned} P_1 &= \left\{\left(1-\frac{1}{m_x}\right)\left[\frac{1}{s}+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m_y}\right)\right]\right\}^{n_{xy}} \\ &= \left(1-\frac{1}{m_x}\right)^{n_{xy}}\left(1-\frac{s-1}{s}\times\frac{1}{m_y}\right)^{n_{xy}} \end{aligned} \tag{5–6}$$

where the first term denotes the probability of v not choosing bit $(b \, mod \, m_x)$ in B_x, and the second term captures the probability of v not choosing bit b in B_y.

(II) *Event E_2: For vehicles passing only R_x (i.e., in the set $S_x - S_y$), none of them have chosen bit $(b \, mod \, m_x)$ in B_x.* Otherwise, from (5–3), bit b in B_x^u is '1' (so bit b in B_{xy} is '1'). Since each vehicle in $S_x - S_y$ has probability $\frac{1}{m_x}$ to set bit $(b \, mod \, m_x)$, and there are $n_x - n_{xy}$ vehicles in $S_x - S_y$, the probability of E_2 is

$$P_2 = \left(1-\frac{1}{m_x}\right)^{n_x-n_{xy}}. \tag{5–7}$$

(III) *Event E_3: For vehicles passing only R_y (i.e., in the set $S_y - S_x$), none of them have chosen bit b in B_y.* Otherwise, bit b in B_y will be '1' (hence bit b in B_{xy} is also '1'). Similarly, we can derive its probability as

$$P_3 = \left(1-\frac{1}{m_y}\right)^{n_y-n_{xy}}. \tag{5–8}$$

Combining above analysis, we can obtain the overall probability $q(n_{xy})$ for bit b in B_{xy} to remain '0' as follows:

$$\begin{aligned} q(n_{xy}) &= P_1 \times P_2 \times P_3 \\ &= \left(1-\frac{1}{m_x}\right)^{n_x}\left(1-\frac{1}{m_y}\right)^{n_y}\left(\frac{1-\frac{s-1}{sm_y}}{1-\frac{1}{m_y}}\right)^{n_{xy}} \end{aligned} \tag{5–9}$$

Since the bits in any logical bit array are selected from the largest physical bit array uniformly at random, the vehicles in set S_x (S_y) have the same probability of $\frac{1}{m_x}$ ($\frac{1}{m_y}$) to choose any bit in B_x (B_y). For any bit in B_x (B_y), the probability for it to be '0' after n_x (n_y)

vehicles each choosing a random bit from B_x (B_y) is

$$q(n_x) = \left(1 - \frac{1}{m_x}\right)^{n_x}, \tag{5–10}$$

$$q(n_y) = \left(1 - \frac{1}{m_y}\right)^{n_y}. \tag{5–11}$$

Therefore, the number of zero bits in B_x follows a binomial distribution $U_x \sim B(m_x, q(n_x)) = B(m_x, (1 - \frac{1}{m_x})^{n_x})$, while the number of zero bits in B_y follows another binomial distribution $U_y \sim B(m_y, q(n_y)) = B(m_y, (1 - \frac{1}{m_y})^{n_y})$. From the property of binomial distribution [48], and $V_x = \frac{U_x}{m_x}$ and $V_y = \frac{U_y}{m_y}$, the expected values for V_x and V_y are

$$E(V_x) = E\left(\frac{U_x}{m_x}\right) = \frac{m_x(1 - \frac{1}{m_x})^{n_x}}{m_x} = q(n_x), \tag{5–12}$$

$$E(V_y) = E\left(\frac{U_y}{m_y}\right) = \frac{m_y(1 - \frac{1}{m_y})^{n_y}}{m_y} = q(n_y). \tag{5–13}$$

Substituting (5–10), (5–11), (5–12) and (5–13) to (5–9), and replacing $E(V_x)$ and $E(V_y)$ by their instance values, V_x and V_y, we have the following instance value for $q(n_{xy})$:

$$q(n_{xy}) = V_x \times V_y \times \left(\frac{1 - \frac{s-1}{s} \times \frac{1}{m_y}}{1 - \frac{1}{m_y}}\right)^{n_{xy}}. \tag{5–14}$$

Given the probability for each bit in B_{xy} to be '0' as $q(n_{xy})$, we can establish the likelihood function $\mathcal{L}$ for us to observe U_{xy} '0' bits in B_{xy} (so $m_y - U_{xy}$ '1' bits in B_{xy}):

$$\mathcal{L} = (q(n_{xy}))^{U_{xy}} \times (1 - q(n_{xy}))^{m_y - U_{xy}}. \tag{5–15}$$

The MLE estimator of n_{xy} is the optimal value of n_{xy} that maximizes the likelihood function in (5–15). To find $\hat{n}_{xy}$, we take logarithm on both sides of (5–15), and then take the first order derivative to obtain:

$$\frac{d \ln \mathcal{L}}{dn_{xy}} = \left(\frac{U_{xy}}{q(n_{xy})} - \frac{m_y - U_{xy}}{1 - q(n_{xy})}\right) \times q'(n_{xy}), \tag{5–16}$$

where $q'(n_{xy})$ can be computed from (5–9) as follows:

$$q'(n_{xy}) = q(n_{xy}) \times \ln\left(\frac{1 - \frac{s-1}{s} \times \frac{1}{m_y}}{1 - \frac{1}{m_y}}\right). \tag{5–17}$$

Since $q'(n_{xy})$ cannot be 0 when $m_x > 1$, $m_y > 1$, and $s < m_y$, setting the right side of (5–16) to 0 gives

$$q(n_{xy}) = \frac{U_{xy}}{m_y} = V_{xy}. \tag{5–18}$$

Substituting (5–18) to (5–14) and reordering the items, we get

$$\left(\frac{1 - \frac{s-1}{s} \times \frac{1}{m_y}}{1 - \frac{1}{m_y}}\right)^{n_{xy}} = \frac{V_{xy}}{V_x \times V_y}. \tag{5–19}$$

Solving (5–19) gives the MLE estimator $\hat{n}_{xy}$ as described in (5–5).

5.2.4 Computation Overhead

We conclude this section by a discussion about the computation overhead of our extension scheme. We compare it with the previous best state of art [47]. The other two performance metrics will be analyzed in the following two sections.

Clearly, the computation overhead for the vehicles and RSUs of our extension scheme are comparable to [47]. In our extension scheme, when a vehicle v passes an RSU R_x, the vehicle v only needs to compute two hashes to obtain an index of a random bit, and the RSU R_x only needs to set 1 bit in its bit array B_x, as described in Section 5.2.1. So the computation overhead for each vehicle per RSU as well as for each RSU per vehicle are both $O(1)$.

As for the central server, the task it performs is a little bit more complicated than [47], but the computation overhead is comparable. First, the server expands the smaller bit array B_x to B_x^u, which has the same size as B_y, by duplicating its content. This operation costs $O(m_y)$ time. Second, it performs a bitwise OR over two m_y-bit bit arrays, B_x^u and B_y, to create a new bit array B_{xy} of size m_y, which also costs $O(m_y)$ time. Last, the server counts the number of zeros in B_x, B_y, and B_{xy}, which takes $O(m_y)$ time as well. Therefore, the overall computation overhead for the server to measure the traffic volume between a pair of

RSUs, R_x and R_y, is $O(m_y)$, where m_y is the size of the larger bit array of the two RSUs. Since our previous scheme with fixed-length bit arrays [47] assumes that $m_x = m_y = m$ and its computation overhead for the server is $O(m)$, one can see that our extension scheme indeed achieves comparable computation overhead as [47].

5.3 Analysis on Measurement Accuracy

In this section, we analyze the measurement accuracy of the MLE estimator $\hat{n}_{xy}$ mathematically. We measure the accuracy by evaluating the bias and standard deviation of $\frac{\hat{n}_{xy}}{n_{xy}}$. Clearly, a good measurement scheme should have close-to-zero bias and relatively small standard deviation.

According to (5–5), $\hat{n}_{xy}$ involves three random variables V_{xy}, V_x, and V_y. Therefore, we first study the mean and variance of V_{xy}, V_x, and V_y, based on which we derive the formula for the bias and standard deviation of $\frac{\hat{n}_{xy}}{n_{xy}}$.

5.3.1 Mean and Variance of V_{xy}, V_x, and V_y

The mean values of V_x and V_y are given in (5–12) and (5–13). Their variance can be computed from the variance of U_x and U_y. Since U_x and U_y each follows a binomial distribution as mentioned in Section 4.2.2, we have

$$Var(V_x) = \frac{Var(U_x)}{{m_x}^2} = \frac{q(n_x) \times (1 - q(n_x))}{m_x}, \tag{5–20}$$

$$Var(V_y) = \frac{Var(U_y)}{{m_y}^2} = \frac{q(n_y) \times (1 - q(n_y))}{m_y}, \tag{5–21}$$

where $q(n_x)$ and $q(n_y)$ are given by (5–10) and (5–11).

Since the probability for any bit in B_{xy} to be '0' is $q(n_{xy})$, the number of zeros in B_{xy} also follows a binomial distribution $U_{xy} \sim B(m_y, q(n_{xy}))$. Therefore, the mean of U_{xy} is $m_y \times q(n_{xy})$ and the variance of U_{xy} is $m_y \times q(n_{xy}) \times (1 - q(n_{xy}))$. Since $V_{xy} = \frac{U_{xy}}{m_y}$, the mean and variance of V_{xy} can be derived accordingly:

$$E(V_{xy}) = E\left(\frac{U_{xy}}{m_y}\right) = \frac{m_y \times q(n_{xy})}{m_y} = q(n_{xy}), \tag{5–22}$$

$$Var(V_{xy}) = \frac{Var(U_{xy})}{{m_y}^2} = \frac{q(n_{xy}) \times (1 - q(n_{xy}))}{m_y}, \tag{5–23}$$

where $q(n_{xy})$ is given by (5–9).

5.3.2 Mean and Variance of $\ln(V_{xy})$, $\ln(V_x)$, and $\ln(V_y)$

Now we derive the mean and variance of $\ln(V_{xy})$, $\ln(V_x)$, and $\ln(V_y)$. First, we define a function $f(V) = \ln V$, and expand the function by its Taylor series about the mean value of V, denoted as $w = E(V)$, to obtain

$$\begin{aligned} f(V) &= f(w) + (V - w)f'(w) + \frac{1}{2}(V - w)^2 f''(w)... \\ &= \ln(w) + \frac{V - w}{w} - \frac{(V - w)^2}{2w^2}... \end{aligned} \tag{5–24}$$

To get the expected value of $\ln(V)$, we truncate (5–24) after the third term since expected value of the second term is 0, and the third term is the first nonzero bias:

$$E(\ln(V)) = \ln(w) - \frac{E((V - w)^2)}{2w^2} = \ln(w) - \frac{Var(V)}{2w^2}. \tag{5–25}$$

According to (5–25), we can compute the mean of $\ln(V_{xy})$, $\ln(V_x)$, and $\ln(V_y)$ based on the mean and variance values of V_{xy}, V_x, and V_y. Below are the results:

$$E(\ln(V_x)) = \ln(q(n_x)) - \frac{1}{2m_x} \times \frac{1 - q(n_x)}{q(n_x)}, \tag{5–26}$$

$$E(\ln(V_y)) = \ln(q(n_y)) - \frac{1}{2m_y} \times \frac{1 - q(n_y)}{q(n_y)}, \tag{5–27}$$

$$E(\ln(V_{xy})) = \ln(q(n_{xy})) - \frac{1}{2m_y} \times \frac{1 - q(n_{xy})}{q(n_{xy})}. \tag{5–28}$$

To get the variance, we truncate (5–24) after two terms:

$$Var(\ln(V)) = Var\left(\ln(w) + \frac{V - w}{w}\right) = \frac{Var(V)}{w^2}. \tag{5–29}$$

Again, according to (5–29), we can compute the variance of $\ln(V_{xy})$, $\ln(V_x)$, and $\ln(V_y)$ based on the mean and variance values of V_{xy}, V_x, and V_y. Below are the results:

$$Var(\ln(V_x)) = \frac{Var(V_x)}{(E(V_x))^2} = \frac{1}{m_x} \times \frac{1 - q(n_x)}{q(n_x)}, \tag{5–30}$$

$$Var(\ln(V_y)) = \frac{Var(V_y)}{(E(V_y))^2} = \frac{1}{m_y} \times \frac{1 - q(n_y)}{q(n_y)}, \tag{5–31}$$

$$Var(\ln(V_{xy})) = \frac{Var(V_{xy})}{(E(V_{xy}))^2} = \frac{1}{m_y} \times \frac{1 - q(n_{xy})}{q(n_{xy})}. \tag{5–32}$$

5.3.3 Mean and Variance of $\hat{n}_{xy}$

Based on the mean of $\ln(V_{xy})$, $\ln(V_x)$, and $\ln(V_y)$ derived previously, we obtain the mean of $\hat{n}_{xy}$:

$$E(\hat{n}_{xy}) = \frac{E(\ln(V_{xy})) - E(\ln(V_x)) - E(\ln(V_y))}{\ln(1 - \frac{s-1}{s} \times \frac{1}{m_y}) - \ln(1 - \frac{1}{m_y})}, \tag{5–33}$$

where $E(\ln(V_{xy}))$, $E(\ln(V_x))$, and $E(\ln(V_y))$ are given in (5–26), (5–27), and (5–28). The estimation bias is

$$Bias\left(\frac{\hat{n}_{xy}}{n_{xy}}\right) = E\left(\frac{\hat{n}_{xy}}{n_{xy}}\right) - 1 = \frac{E(\hat{n}_{xy})}{n_{xy}} - 1. \tag{5–34}$$

We can also derive the variance of $\hat{n}_{xy}$ as

$$Var(\hat{n}_{xy}) = \frac{F + G}{\left[\ln\left(1 - \frac{s-1}{s} \times \frac{1}{m_y}\right) - \ln\left(1 - \frac{1}{m_y}\right)\right]^2} \tag{5–35}$$

where $F = Var(\ln(V_{xy})) + Var(\ln(V_x)) + Var(\ln(V_y))$, and $G = -G_1 - G_2 + G_3$ with $G_1 = Cov(\ln(V_{xy}), \ln(V_x))$, $G_2 = Cov(\ln(V_{xy}), \ln(V_y))$, and $G_3 = Cov(\ln(V_x), \ln(V_y))$. The first part F is easy to compute since we already derived the variances $Var(\ln(V_{xy}))$, $Var(\ln(V_x))$, and $Var(\ln(V_y))$ in (5–32), (5–30), and (5–31). The three covariance terms can be derived by expanding the Taylor series of $\ln(V_{xy})$, $\ln(V_x)$, and $\ln(V_y)$ about the mean values

of V_{xy}, V_x, and V_y. For example, G_1 is derived as

$$\begin{aligned} G_1 &= -E(\ln(V_{xy}))E(\ln(V_x)) + E(\ln(V_{xy})\ln(V_x)) \\ &= -E(\ln(V_{xy}))E(\ln(V_x)) - \ln(E(V_{xy}))\ln(E(V_x)) \\ &+ \ln(E(V_{xy}))E(\ln(V_x)) + \ln(E(V_x))E(\ln(V_{xy})) \end{aligned} \quad (5\text{–}36)$$

Substituting the formula of $E(V_{xy})$, $E(V_x)$, $E(\ln(V_{xy}))$, and $E(\ln(V_x))$, which we have already computed, we can obtain $Cov(\ln(V_{xy}), \ln(V_x))$. $Cov(\ln(V_{xy}), \ln(V_y))$ and $Cov(\ln(V_x), \ln(V_y))$ can be derived similarly. After obtaining the covariances, we can compute the variance of $\hat{n}_{xy}$ based on (5–35). Finally, given $Var(\hat{n}_{xy})$, the standard deviation of $\frac{\hat{n}_{xy}}{n_{xy}}$ is computed as follows:

$$StdDev\left(\frac{\hat{n}_{xy}}{n_{xy}}\right) = \frac{\sqrt{Var(\hat{n}_{xy})}}{n_{xy}}. \quad (5\text{–}37)$$

5.4 Analysis on Preserved Privacy

Now we analyze the preserved privacy of our extension scheme. Recall from Section 5.2, similar to the previous scheme based on fixed-length bit arrays, the only information that a vehicle v ever transmits to an RSU en route is an index of a bit b randomly selected from its s-bit logical bit array, LB_v. Since the s bits in each vehicle's logical bit array are chosen randomly from the RSUs' physical bit arrays, from the adversary's point of view, every vehicle has the same probability to set any arbitrary bit of an RSU's bit array. In other words, the adversary cannot determine the identity of a vehicle simply given its reported index. Therefore, the first-level privacy of each individual vehicle is clearly preserved.

In the following, we focus on the second-level privacy. Again, since each vehicle just transmits a random bit index to each passing RSU, from the adversary's point of view, it can only attempt to identify the trace of a vehicle passing by two RSUs R_x and R_y through the observation of the bits that are set to '1' in both B_x and B_y. Therefore, the second-level privacy of our extension scheme is also the conditional probability which states to what degree

observing a same bit to be set in both bit arrays of two RSUs does not represent a common vehicle passing by both RSUs. Below, we mathematically derive this conditional probability.

5.4.1 Derivation of the Preserved Privacy

First, consider the probability for an arbitrary bit, b, to be '1' in both B_x^u and B_y (event A), $P(A)$. Denote its complementary event as $\bar{A}$. Clearly, $P(A) = 1 - P(\bar{A})$. Denote by S the subset of vehicles in $S_x \cap S_y$ that happen to choose the same bit in its logical bit array at both R_x and R_y. Let n_s be the cardinality of S, i.e., $n_s = |S|$. Clearly, $S \subseteq S_x \cap S_y$ and $0 \leq n_s \leq n_{xy}$. As we mentioned earlier, the probability for $v \in S_x \cap S_y$ to select the same bit at both R_x and B_y is $\frac{1}{s}$. Therefore, the number of such vehicles, n_s, is binomially distributed according to $B(n_{xy}, \frac{1}{s})$. The probability for $n_s = z\,(0 \leq z \leq n_{xy})$ is

$$P(n_s = z) = \binom{n_{xy}}{z} \left(\frac{1}{s}\right)^z \left(1 - \frac{1}{s}\right)^{n_{xy}-z}. \tag{5–38}$$

Clearly, event $\bar{A}$ is equivalent to the combination of the following two events: (1) *Event E_4: None of the vehicles in S has chosen b at R_x and R_y.* Otherwise, bit $(b \bmod m_x)$ in B_x (hence bit b in B_x^u) and bit b in B_y are both set to '1'. Clearly, the probability of E_4 is

$$q_4 = \left(1 - \frac{1}{m_y}\right)^{n_s}. \tag{5–39}$$

(2) *Event E_5: Either none of the vehicles in $S_x - S$ has chosen $(b \bmod m_x)$ at R_x or none of the vehicles in $S_y - S$ has chosen b at R_y.* Otherwise, the two corresponding bits are both set to '1'. Clearly, the probability of E_5 is

$$q_5 = 1 - \left[1 - \left(1 - \frac{1}{m_x}\right)^{n_x - n_s}\right]\left[1 - \left(1 - \frac{1}{m_y}\right)^{n_y - n_s}\right] \tag{5–40}$$

Combining above analysis, the probability of event $\bar{A}$ is

$$\begin{aligned} P(\bar{A}) &= \sum_{z=0}^{n_{xy}} q_4(n_s|n_s=z)q_5(n_s|n_s=z)P(n_s=z) \\ &= \left(1-\frac{1}{m_x}\right)^{n_x} \times C_1^{n_{xy}} + \left(1-\frac{1}{m_y}\right)^{n_y} \\ &\quad -\left(1-\frac{1}{m_x}\right)^{n_x}\left(1-\frac{1}{m_y}\right)^{n_y} \times C_2^{n_{xy}}, \end{aligned} \tag{5–41}$$

where C_1 and C_2 are both constants with values

$$C_1 = \frac{1}{s} \times \frac{1-\frac{1}{m_y}}{1-\frac{1}{m_x}} + \left(1-\frac{1}{s}\right), \tag{5–42}$$

$$C_2 = \frac{1}{s} \times \frac{1}{1-\frac{1}{m_x}} + \left(1-\frac{1}{s}\right). \tag{5–43}$$

Secondly, consider the conditional probability for such a bit, b, to not represent a common vehicle passing both R_x and R_y (event E), $P(E|A)$. Note that event E happens if and only if bit $(b \, mod \, m_x)$ in B_x (hence bit b in B_x^u) is set only by vehicles passing only RSU R_x (i.e., in $S_x - S_y$), and bit b in B_y is set only by vehicles passing only RSU R_y (i.e., in $S_y - S_x$). Denote these two events as E_x and E_y, respectively. We can easily derive their probability as:

$$P(E_x) = \left(1-\left(1-\frac{1}{m_x}\right)^{n_x-n_{xy}}\right) \times \left(1-\frac{1}{m_x}\right)^{n_{xy}}, \tag{5–44}$$

$$P(E_y) = \left(1-\left(1-\frac{1}{m_y}\right)^{n_y-n_{xy}}\right) \times \left(1-\frac{1}{m_y}\right)^{n_{xy}}. \tag{5–45}$$

Therefore, the preserved privacy of our novel scheme is:

$$\begin{aligned} p &= P(E|A) = \frac{P(E_x) \times P(E_y)}{P(A)} \\ &= \frac{1}{1-P(\bar{A})} \times \left[\left(1-\frac{1}{m_x}\right)^{n_{xy}} - \left(1-\frac{1}{m_x}\right)^{n_x}\right] \\ &\quad \times \left[\left(1-\frac{1}{m_y}\right)^{n_{xy}} - \left(1-\frac{1}{m_y}\right)^{n_y}\right], \end{aligned} \tag{5–46}$$

where $P(\bar{A})$ is given in (5–41). Note that if we set $m_x = m_y = m$ in (5–46), we get the same formula as [47], which means that [47] is just a special case of our extension scheme.

5.4.2 Privacy Comparison with the Best State of Art

Note that the previous scheme [47] based on fixed-length bit arrays works great only if all RSUs face comparable traffic. We have mentioned that, for that scheme, if a large m is chosen to accommodate heavy-traffic RSUs, the privacy of cars passing light-traffic RSUs will be greatly hurt. Here we give more explanations through numerical analysis.

The first plot of Figure 5-3 shows the second-level privacy p of [47] when m varies from $0.1n$ to $50n$, controlled by $s = 2, 5, 10$, where $n_x = n_y = n$. From the plot, one can see that the privacy of [47] is actually determined by the ratio f (called load factor) of m over n, and the optimal privacy is achieved at the optimal load factor f^* (approximately from 2 to 4). An important observation is that, when m is fixed, the privacy will vary a lot given different n (hence different f). If we choose a large m to accommodate RSUs with large n, say $n = n' = 500,000$, and $m = f_1 n' = 2n'$, then the privacy of cars passing RSUs with smaller n, say $n = n'' = \frac{n'}{25} = 20,000$, will be greatly hurt since the load factor for those RSUs will be $f_2 = 25 f_1 = 50$ (see the rightmost point of the three curves). Specifically, the privacy suffers most for small values of s. For example, when $s = 2$, the privacy is only about 0.2. One can expect more drop in privacy for cars passing RSUs with less traffic. To guarantee a minimum privacy of cars regardless of RSUs, the value of m should be determined by the least traffic volume among all RSUs, n_{min}. For example, m should be no larger than $15n_{min}$ to guarantee a minimum privacy of 0.5 when $s = 2$. However, this brings another problem: the measurement accuracy for heavy-traffic RSUs will dramatically decrease (more on Section 5.5).

The problem of plummeted privacy in [47] originates from the fact that different RSUs have different traffic volume, and using same-length bit arrays will cause "unbalanced load factors". Below, we show that by using variable-length bit arrays so that their load factors are comparable, our novel scheme not only solves the plummeted privacy problem in [47], but also improves the optimal privacy when the traffic volume differs.

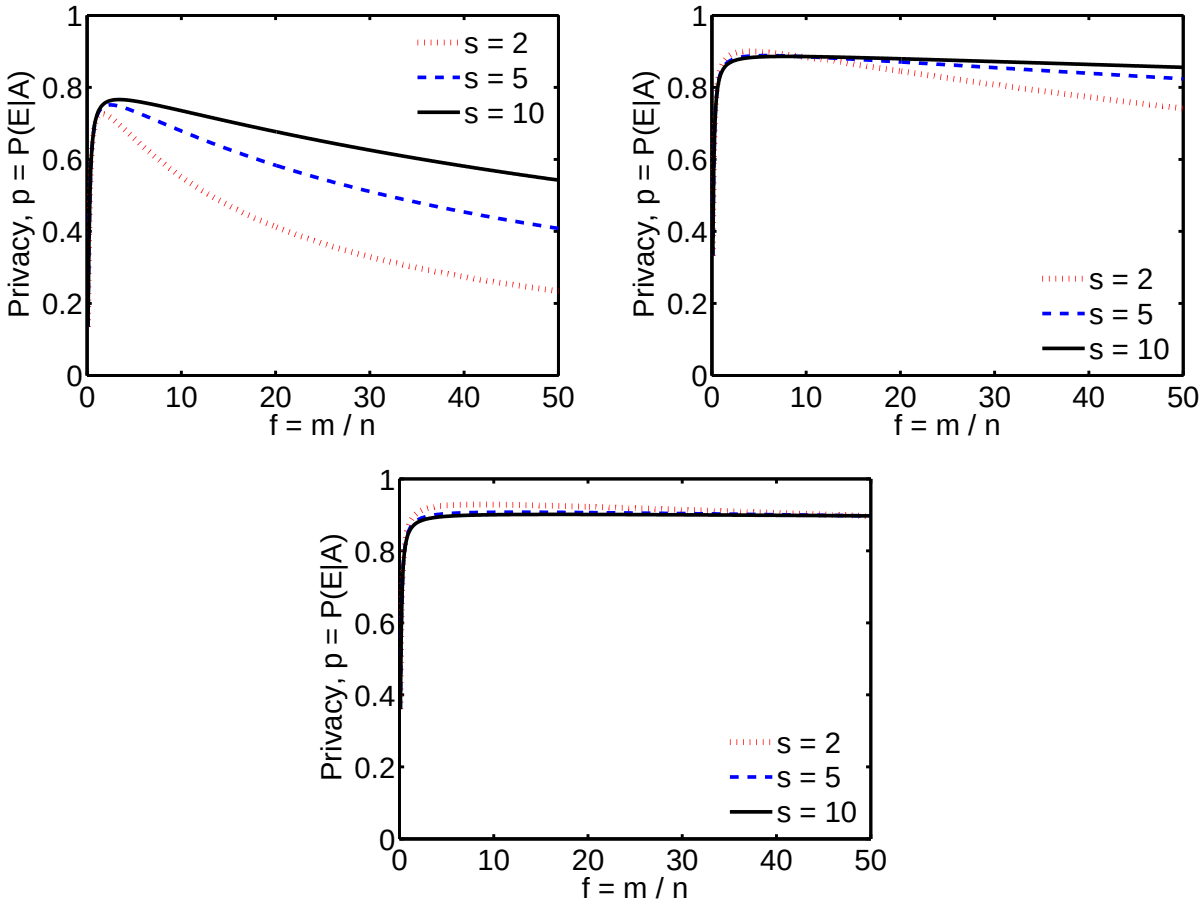


Figure 5-3. Preserved privacy of the two-point traffic flow measurement schemes with bit array masking. *First Plot*: the privacy of both schemes with equal m; *Second Plot*: the privacy of our extension scheme when $n_y = 10n_x$; *Third Plot*: the privacy of our extension scheme with $n_y = 50n_x$.

Figure 5-3 shows the second-level privacy p of our novel extension scheme when the load factor varies from 0.1 to 50. Note that we use the same load factor $\bar{f}$ for all RSUs (so the lengths of bit arrays will vary given different traffic volume n at different RSUs). When the single-point traffic volume of R_x and R_y are comparable, their bit arrays will have the same length, i.e., $m_x = m_y = m$, so the privacy formula for both schemes will be the same, resulting in the same graph as shown in the first plot of Figure 5-3. We stress that for our extension scheme, since all RSUs use the same load factor $\bar{f}$, the privacy of all cars, regardless of the traffic volume of RSUs that they pass, will always be comparable as the optimal privacy if $\bar{f} = f^*$. For example, when $s = 5$, the privacy p of the cars passing comparable-traffic RSUs

will be more than 0.75. For RSUs with different traffic volume, the extension scheme has another advantage, which is improving the privacy of the cars passing those RSUs. The second and the third plot of Figure 5-3 show the privacy that our extension scheme preserves for cars passing RSUs with different traffic volume where $n_y = 10n_x$ and $n_y = 50n_x$, respectively. One can see that given $\bar{f} = f^*$, both plots show better optimal privacy than the case with comparable-traffic RSUs. For instance, given $\bar{f} = 3$ when $s = 5$, the optimal privacy is 0.89 for $n_y = 10n_x$, and 0.91 for $n_y = 50n_x$, both greater than the optimal privacy of 0.75 for $n_x = n_y$. The improved privacy originates from the variable-length bit arrays. During the "unfolding" process, the content of B_x is duplicated to generate B_x^u. This effectively creates more common '1' bits in B_x^u and B_y that are not caused by common cars, thus adding one more level of "mask" for the traces of common vehicles.

5.5 Simulation

In this section, we compare the performance of our extension scheme with the previous best state of art [47] through simulations. There are two sets of simulations: the first set of simulations considers a real Sioux Falls road network with known vehicle trip tables, while the second set considers a larger network with randomly generated traffic.

5.5.1 Simulation Results of the Sioux Falls Network

We first consider a real road network of Sioux Falls with known vehicle trip tables. First published by Lebranc etc. in [64], the Sioux Falls network has made its appearance in thousands of conference papers, journals and books (e.g., [65], [66], [67]). As illustrated by Figure 5-4, the Sioux Falls network contains 24 nodes (RSUs) with 76 arcs (road segments). In our simulations, we generate traffic according to the known vehicle trip table in [64] under the Sioux Falls network, and compute the daily two-point traffic volume between each pair of nodes using both the scheme in [47] and our extension scheme. The parameters for the two schemes are determined as follows. For both schemes, the number of bits in the logical bit array of each vehicle, s, is set to $2, 5, 10$ as [47]. $\bar{f}$ and m are chosen to guarantee a minimum

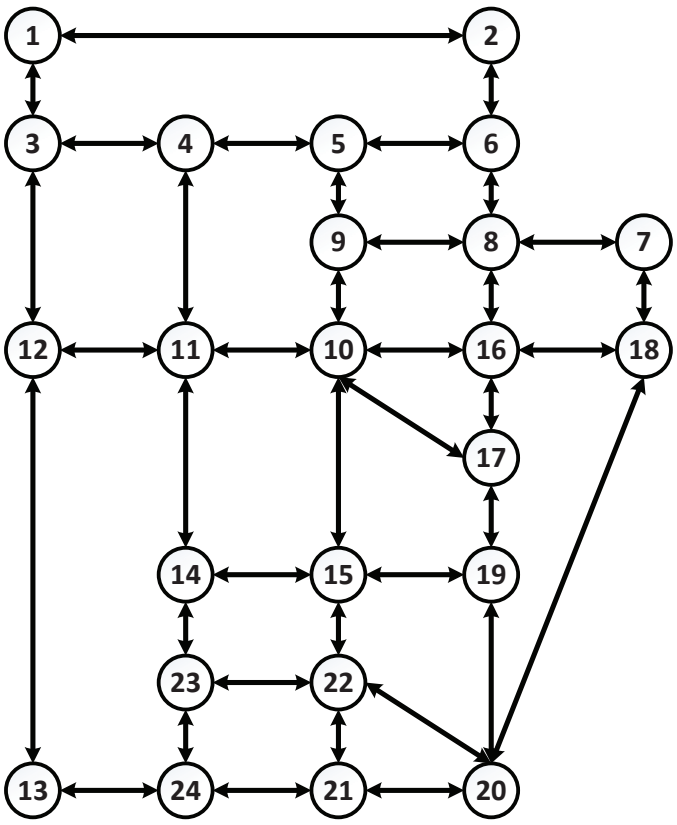


Figure 5-4. Sioux Falls Road Network.

privacy of at least 0.5. Recall that $\bar{f}$ is the fixed load factor for all bit arrays in our extension scheme, and m is the fixed bit array length in [47].

Table 5-1 shows the simulation results of eight typical node pairs in the Sioux Falls network of our extension scheme and the previous scheme in [47] under $s = 2$. Note that the unit for the traffic volume is thousands of vehicles/day. Also, since node 10 has the largest traffic volume among all 24 nodes, it is chosen to be RSU R_y with $n_y = 451$. The other RSU R_x in each pair is randomly selected from the remaining nodes, and they are sorted according to their traffic difference ratio against R_y (i.e., $d = \frac{n_y}{n_x}$). The two-point traffic volume between each pair of R_x and R_y is measured by both our extension scheme and the scheme in [47], and the error ratio against the real traffic volume n_{xy}, i.e., $r = \frac{|\hat{n}_{xy} - n_{xy}|}{n_{xy}} \times 100\%$, is also calculated to better show the results. Clearly, the smaller the error ratio, the better the measurement result.

Table 5-1. Simulation results for the Sioux Falls network of our two schemes based on bit array masking. The unit for the traffic volume is thousands of vehicles/day. $R_y = 10$, $n_y = 451$. RSUs are sorted according to the traffic difference ratio against R_y, i.e., $d = \frac{n_y}{n_x}$. The error ratio for both scheme is defined as $r = \frac{|\hat{n}_{xy} - n_{xy}|}{n_{xy}} \times 100\%$.

R_x	15	12	7	24	6	18	2	3
n_x	213	140	121	78	76	47	40	28
d	2.117	3.221	3.727	5.782	5.934	9.596	11.275	16.107
n_{xy}	40	20	19	8	8	7	6	3
$\hat{n}_{xy}$ ([47])	40.048	19.881	19.195	7.215	7.517	6.106	6.637	2.638
$\hat{n}_{xy}$	39.950	19.972	18.982	7.976	7.988	6.979	5.999	3.005
r ([47])	0.12%	0.60%	1.03%	9.81%	6.04%	12.77%	10.62%	12.07%
r	0.13%	0.14%	0.09%	0.30%	0.15%	0.30%	0.02%	0.17%

From Table 5-1, one can see that when the traffic difference ratio d is small (i.e., the traffic volume of R_x and R_y are comparable), e.g., $n_y \approx 2n_x$ in the second column of Table 5-1, both measurement schemes can achieve very accurate results (both around 0.1%). However, when the gap of traffic volume between two RSUs enlarges, the scheme in [47] starts to produce less and less accurate results. One can see that the error ratio r of [47] increases by an order of magnitude when the traffic difference ratio $d \approx 4$ (the fourth column of Table 5-1), and over 2 orders of magnitude when $d \approx 16$ (the last column of Table 5-1). On the other hand, our extension scheme remains accurate for all RSU pairs, with error ratio r constantly below 0.3%, which reflects its superior performance over [47].

5.5.2 Simulation Results of Randomly Generated Traffic

Next, we consider a larger network where the traffic is randomly generated. The simulations are controlled by six parameters, n_x, n_y, n_{xy}, s, $\bar{f}$, and m. Their values are chosen as follows: $n_x = 10,000$, $n_y = n_x\,(10,000)$, $10n_x\,(100,000)$, or $50n_x\,(500,000)$, n_{xy} varies from $0.01n_x$ to $0.5n_x$, with step size of $0.001n_x$. s is set to $2, 5, 10$, and $\bar{f}$ and m are chosen to guarantee a minimum privacy of at least 0.5.

Figure 5-5 shows the simulation results for [47], and Figure 5-6 shows the results for our extension scheme, both under $s = 2$. Since the simulations for $s = 5$ and $s = 10$ show similar results, here we omit them to save space. For each figure, there are three plots, corresponding to the results of three groups of simulations controlled by n_y and n_x, where

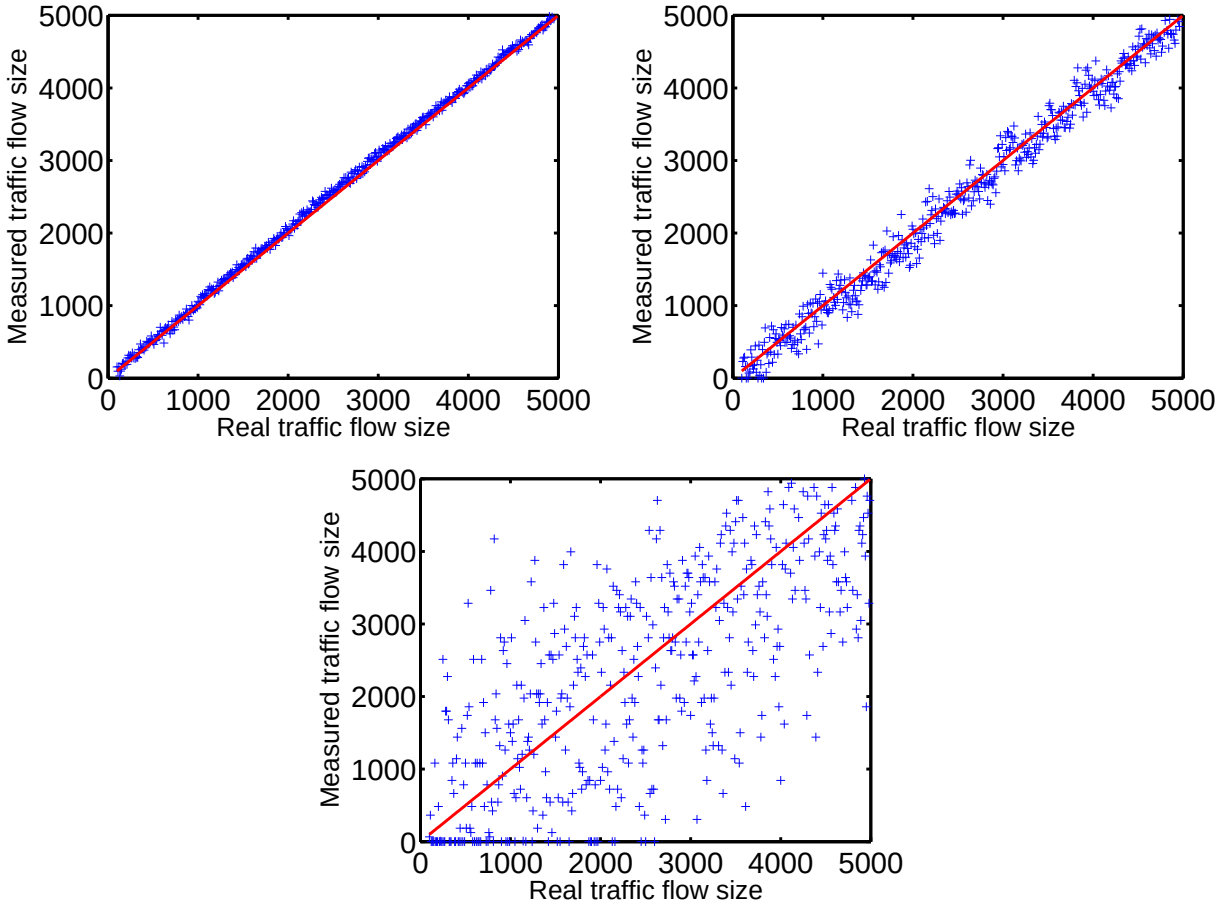


Figure 5-5. Measurement accuracy of the two-point traffic flow measurement scheme based on fixed-length bit array masking. The x-axis shows the real two-point traffic flow size, and the y-axis shows measured two-point traffic flow size. $s = 2$, $n_x = 10,000$, $n_{xy} = [0.01n_x, 0.5n_x]$. *First Plot*: $n_y = n_x$; *Second Plot*: $n_y = 10n_x$; *Third Plot*: $n_y = 50n_x$.

$n_y = n_x$, $n_y = 10n_x$, and $n_y = 50n_x$, respectively. Each plot shows the measured traffic volume $\hat{n}_{xy}$ (y-axis) with respect to the real traffic volume n_{xy} (x-axis). The equality line $y = x$ is also drawn for reference. Clearly, the closer a point is to the equality line, the better the measurement result it represents.

From the first plot of Figure 5-5 and 5-6, one can observe that both schemes achieve perfect performance when $n_y = n_x$. The reason for their comparable performance here is that our extension scheme is almost the same as [47] when $n_y = n_x = n$ (hence $m_y = m_x = m = \bar{f} \times n$). However, when RSUs with different traffic volume are involved, the

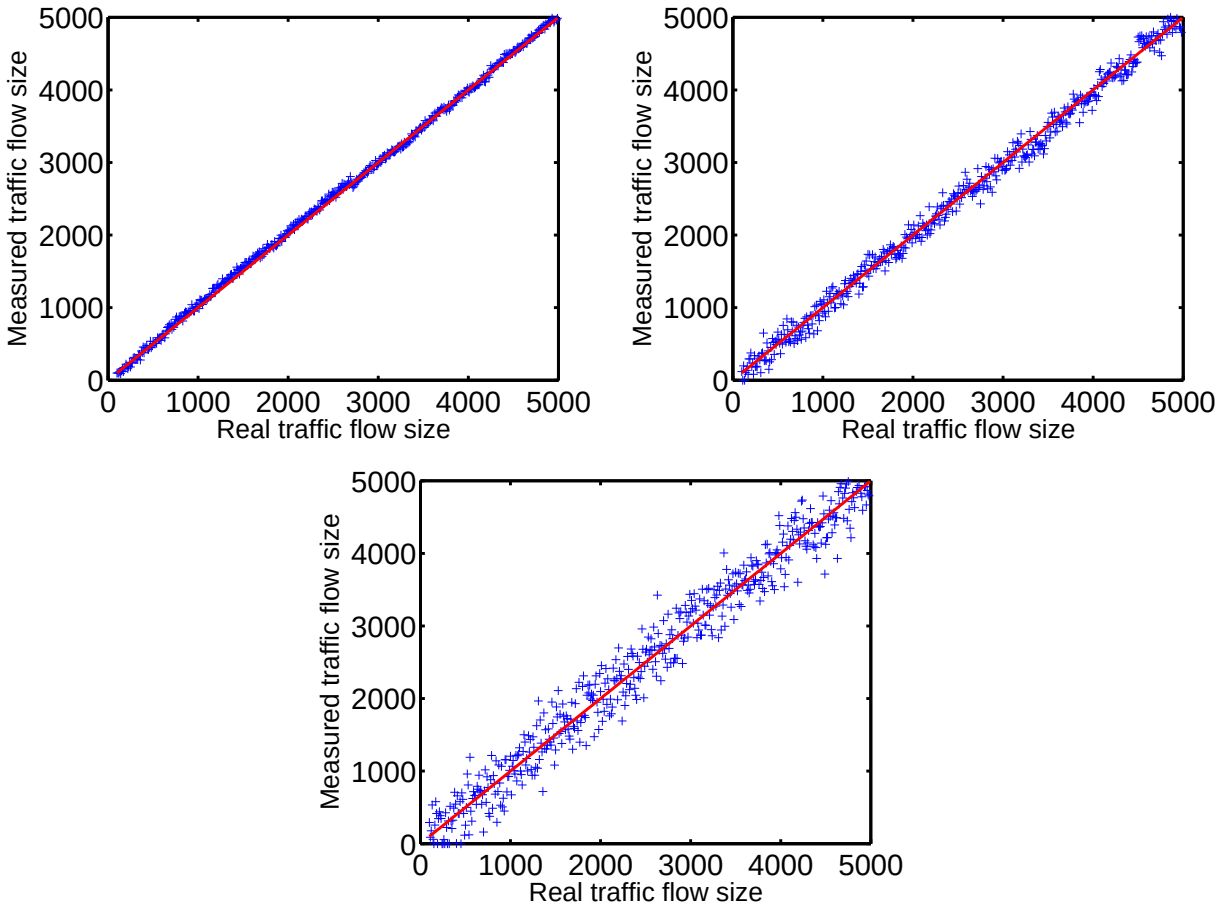


Figure 5-6. Measurement accuracy of the two-point traffic flow measurement scheme based on variable-length bit array masking. The x-axis shows the real two-point traffic flow size, and the y-axis shows measured two-point traffic flow size. $s = 2$, $n_x = 10,000$, $n_{xy} = [0.01n_x, 0.5n_x]$. *First Plot*: $n_y = n_x$; *Second Plot*: $n_y = 10n_x$; *Third Plot*: $n_y = 50n_x$.

measurement accuracy of [47] decreases dramatically. In particular, when $n_y = 50n_x$, the results of [47] are quite inaccurate (the measured results almost scatter everywhere in the third plot of Figure 5-5). On the contrary, the performance of our extension scheme stays accurate (the measured traffic volume closely follow their real values in Figure 5-6). This superior performance originates from our novel design of variable-length bit arrays and the "unfolding" technique, which eliminates the "unbalanced load factor" problem that [47] suffers.

5.6 Summary

In this chapter, we propose our fourth novel scheme for privacy-preserving two-point traffic flow measurement in CPRS, which targets at removing the similar traffic assumption made by the previous scheme [47] to fit in more realistic situations where different RSUs observe varied amount of traffic. This novel scheme tackles the efficiency, privacy, and accuracy problems encountered by all previous solutions. It utilizes variable-length bit arrays to encode traffic data reported by vehicles, and a novel "unfolding" technique to support two-point traffic flow measurement based on those variable-length bit arrays. The novel scheme achieves better privacy for vehicles, more accurate measurement results, and comparable computation overhead, compared with the previous best scheme [47], which is based on fixed-length bit arrays. We demonstrate its applicability through both mathematical and numerical analysis. The simulation results also show the superior performance of the extension scheme.

CHAPTER 6
PRIVACY-PRESERVING THREE-POINT TRAFFIC MEASUREMENT

6.1 From Two-Point Traffic Measurement to Multi-Point Traffic Measurement

In the previous three chapters, we have proposed four novel schemes for privacy-preserving two-point traffic measurement in CPRS, which can be used to automatically collect and efficiently measure the two-point traffic flows passing two arbitrary RSUs while preserving the privacy of vehicles. In particular, the fourth scheme [49] proposed in the previous chapter tackles the efficiency, privacy, and accuracy problems encountered by all previous solutions. It utilizes variable-length bit arrays to encode traffic data reported by vehicles, and a novel "unfolding" technique to support two-point traffic measurement based on those variable-length bit arrays. The novel scheme achieves better privacy for vehicles, more accurate measurement results, and comparable computation overhead, compared with the previous best scheme [47], which is based on fixed-length bit arrays. We demonstrate its applicability through both mathematical and numerical analysis as well as extensive simulation results. To serve for a broader spectrum of applications in vehicular networks and transportation engineering, we are motivated to generalize our design to address the more challenging problem of privacy-preserving multi-point traffic measurement.

In this chapter, we will show how to extend our idea of variable-length bit array masking to address the important problem of privacy-preserving three-point traffic measurement, which observes the potential of further generalization to deal with the problem of privacy-preserving multi-point traffic measurement. Intuitively, if we can "unfold" two variable-length bit arrays to obtain statistical results related to the two-point traffic volume, we should also be able to "unfold" three or more variable-length bit arrays to get a statistical estimator for the multi-point traffic volume. The measurement process should be similar: Vehicles report random indices from their logical bit arrays to mark RSUs' variable-length bit arrays, and the central server performs "unfolding" and bitwise OR operations on three or more bit arrays to obtain statistical results related to the multi-point traffic volume. If an MLE estimator can also be

mathematically derived from those statistical results, it should be easy for the central server to compute the multi-point traffic volume.

In the remaining of this chapter, we will follow the above thinking to develop a novel privacy-preserving three-point traffic measurement scheme. We first explain the two measurement phases of the proposed scheme, and validate the MLE estimator used to measure three-point traffic volume, then analyze its performance. In the next chapter, we will discuss how to further extend our design to address the more general problem of privacy-preserving multi-point traffic measurement.

6.2 Privacy-Preserving Three-Point Traffic Measurement Scheme

In this section, we present our novel scheme for privacy-preserving three-point traffic measurement, which is an extension of our previous two-point traffic measurement scheme. The basic idea is similar: different RSUs will use different-sized bit arrays to collect "masked" information (random bit indices in RSUs' bit arrays) from the passing vehicles, and the central server will measure the three-point traffic flow sizes based on the collected bit arrays, utilizing an "unfolding" technique and the statistical MLE method. There are two measurement phases, online coding and offline decoding, which are explained in the following.

6.2.1 Online Coding Phase

The online coding phase of our three-point scheme works exactly the same as our two-point scheme in [49]. Each RSU R_x maintains a counter n_x to record the total number of passing vehicles, and a bit array B_x with length $m_x = 2^{\lceil \log_2(\bar{n}_x \times \bar{f}) \rceil}$ to collect vehicles' "masked" data, where $\bar{n}_x$ is the expected single-point traffic volume in R_x, and $\bar{f}$ is a system wide load factor, whose value is the same for all RSUs. At the beginning of each measurement period, n_x and all bits in B_x are set to zeros. For privacy protection, each vehicle v also has a logical bit array LB_v, which consists of s bits randomly selected from the largest bit array B_* among all RSUs. The bit indices in B_* are $H(v \oplus K_v \oplus X[0])$, ..., $H(v \oplus K_v \oplus X[s-1])$. Some frequently-used notations can be found in Table 6-1.

Table 6-1. Frequently-used Notations.

Notation	Meaning
R, R_x, R_y, R_z	RSUs
n_x, n_y, n_z	single-point traffic volume of RSU R_x, R_y, R_z
n_{xy}, n_{xz}, n_{yz}	two-point traffic volume between two RSUs
n_{xyz}	three-point traffic volume among three RSUs
$\hat{n}_{xy}, \hat{n}_{xz}, \hat{n}_{yz}$	MLE estimator of the two-point traffic volume n_{xy}, n_{xz}, n_{yz}
$\hat{n}_{xyz}$	MLE estimator of the three-point traffic volume n_{xyz}
B_x, B_y, B_z	bit arrays of RSUs R_x, R_y, R_z
B_{xy}, B_{xz}, B_{yz}	"unfolding" and "bitwise OR" result of two bit arrays
B_{xyz}	"unfolding" and "bitwise OR" result of three bit arrays
m_x, m_y, m_z	sizes of bit arrays B_x, B_y, B_z
m_{xy}, m_{xz}, m_{yz}	sizes of bit arrays B_{xy}, B_{xz}, B_{yz}
m_{xyz}	size of the bit array B_{xyz}
U_x, U_y, U_z	number of zeros in bit arrays B_x, B_y, B_z
U_{xy}, U_{xz}, U_{yz}	number of zeros in bit arrays B_{xy}, B_{xz}, B_{yz}
U_{xyz}	number of zeros in the bit array B_{xyz}
V_x, V_y, V_z	ratio of zeros in bit arrays B_x, B_y, B_z
V_{xy}, V_{xz}, V_{yz}	ratio of zeros in bit arrays B_{xy}, B_{xz}, B_{yz}
V_{xyz}	ratio of zeros in the bit array B_{xyz}
LB_v	the logical bit array of vehicle v
s	size of the logical bit array of every vehicle
f, f_x, f_y, f_z	load factor, ratio of an RSU's bit array size over its traffic volume, $f_x = \frac{m_x}{n_x}$
$\bar{f}$	fixed load factor for all RSUs in the extension scheme [49]
m	fixed bit array size for all RSUs in the previous scheme[47], $m_i = m, \forall R_i$

In the online coding phase, vehicles and RSUs cooperate to automatically collect "masked" traffic data. When a vehicle v receives a query from an RSU R_x, whose bit array is B_x with size m_x, it first verifies if R_x is from the trustworthy authority via its certificate. Once R_x is authenticated, v randomly selects a bit from LB_v by computing an index $b = H(v \oplus K_v \oplus X[H(R_x \oplus t) \bmod s])$, where t is the current time stamp, then generates an index $b_x = b \bmod m_x$ in the range of $[0, m_x)$, and sends b_x to R_x. Upon receiving the index b_x, R_x will increase its counter n_x by 1, and set the b_xth bit in B_x to 1.

6.2.2 Offline Decoding Phase

At the end of each measurement period, similar to our two-point scheme, all RSUs will send their counters and bit arrays to the central server, which first updates the history single-point traffic data for the RSUs to take into account the current measurement period.

After that, the server will measure the three-point traffic volume among three arbitrary RSUs based on the reported counters and bit arrays, which does incur a little bit more work than the two-point measurement (due to the third involving RSU). However, the measurement process is still similar, and the computation overhead is also comparable to the two-point case.

Before describing the measurement process, we first define some notations (also summarized in Table 6-1). We denote the set of vehicles that pass RSU R_x, R_y, R_z as S_x, S_y, S_z with cardinality $|S_x| = n_x$, $|S_y| = n_y$, $|S_z| = n_z$, respectively. Clearly, the set of vehicles that pass through the set of three RSUs $\{R_x, R_y, R_z\}$ is $S_x \cap S_y \cap S_z$. Denote its cardinality as n_{xyz}, i.e., $n_{xyz} = |S_x \cap S_y \cap S_z|$, which is the value that we want to measure. Also, the set of vehicles passing both R_x and R_y is $S_x \cap S_y$, whose size is denoted as n_{xy}, i.e., $n_{xy} = |S_x \cap S_y|$. Similarly, we have $n_{xz} = |S_x \cap S_z|$, and $n_{yz} = |S_y \cap S_z|$. In addition, we denote the size of the bit array B_x, B_y, B_z stored in RSU R_x, R_y, R_z as m_x, m_y, m_z, respectively. Without loss of generality, we assume that $m_x \leq m_y \leq m_z$.

Given above notations, the central server measures n_{xyz} by first performing the following four steps of "unfolding" and bitwise OR operations, and then computing the MLE estimator in (6–5).

Step 1: The server unfolds B_x to the same size of B_y, and takes a bitwise OR operation on the unfolded bit array and B_y to obtain a new bit array B_{xy} of size m_y. More specifically,

$$B_{xy}[i] = B_x[i \bmod m_x] \vee B_y[i], \ \forall i \in [0, m_y). \tag{6–1}$$

Step 2: The server unfolds B_x to the same size of B_z, and takes a bitwise OR operation on the unfolded bit array and B_z to obtain a new bit array B_{xz} of size m_z. More specifically,

$$B_{xz}[i] = B_x[i \bmod m_x] \vee B_z[i], \ \forall i \in [0, m_z). \tag{6–2}$$

Step 3: The server unfolds B_y to the same size of B_z, and takes a bitwise OR operation on the unfolded bit array and B_z to obtain a new bit array B_{yz} of size m_z. More specifically,

$$B_{yz}[i] = B_y[i \bmod m_y] \vee B_z[i], \ \forall i \in [0, m_z). \tag{6–3}$$

Step 4: The server unfolds B_x and B_y to the same size of B_z, and takes a bitwise OR operation on the two unfolded bit arrays and B_z to obtain a new bit array B_{xyz} of size m_z. More specifically,

$$B_{xyz}[i] = B_x[i \bmod m_x] \vee B_y[i \bmod m_y] \vee B_z[i], \ \forall i \in [0, m_z). \tag{6–4}$$

Given B_x, B_y, B_z, B_{xy}, B_{xz}, B_{yz}, and B_{xyz}, the MLE formula that the central server uses to estimate the three-point traffic volume of RSUs R_x, R_y, and R_z is:

$$\hat{n}_{xyz} = \frac{W}{\ln\left(1 - \frac{1}{m_z}\right) + \ln(C_3) - \ln(C_4) - 2\ln(C_5)}, \tag{6–5}$$

where W is a function of zero ratios in the bit arrays

$$W = \ln V_{xyz} + \ln V_x + \ln V_y + \ln V_z - \ln V_{xy} - \ln V_{xz} - \ln V_{yz}, \tag{6–6}$$

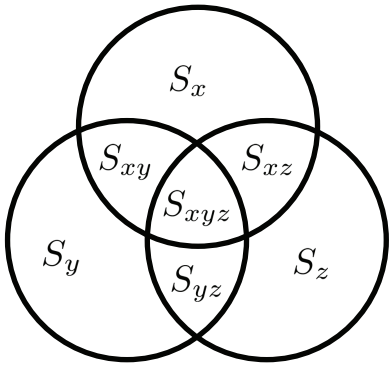
and C_3, C_4, C_5 are constants whose values are:

$$\begin{aligned} C_3 &= \frac{1}{s} \times \left(1 - \frac{s-1}{s} \times \frac{1}{m_z}\right) \\ &+ \left(1 - \frac{1}{s}\right)\left(1 - \frac{1}{m_y}\right)\left(1 - \frac{s-2}{s} \times \frac{1}{m_z}\right), \end{aligned} \tag{6–7}$$

$$C_4 = 1 - \frac{s-1}{s} \times \frac{1}{m_y}, \tag{6–8}$$

$$C_5 = 1 - \frac{s-1}{s} \times \frac{1}{m_z}. \tag{6–9}$$

In (6–6), V_{xyz}, V_x, V_y, V_z, V_{xy}, V_{xz}, V_{yz} are random variables (R.V.) which represent the fraction of bits whose values are zeros in the bit arrays B_{xyz}, B_x, B_y, B_z, B_{xy}, B_{xz}, B_{yz}, correspondingly. Their values can be easily found by counting the number of zeros in the bit arrays, denoted by U_{xyz}, U_x, U_y, U_z, U_{xy}, U_{xz}, U_{yz} (note that they are also R.V.s) respectively, and dividing them by the corresponding bit array size. For example, $V_{xyz} = \frac{U_{xyz}}{m_z}$, $V_x = \frac{U_x}{m_x}$, and $V_{xy} = \frac{U_{xy}}{m_y}$, etc.



$$S_{xyz} = S_x \cap S_y \cap S_z \qquad S_{xy} = S_x \cap S_y - S_z$$
$$S_{xz} = S_x \cap S_z - S_y \qquad S_{yz} = S_y \cap S_z - S_x$$

Figure 6-1. Venn diagram for the set of vehicles, $S_x \cup S_y \cup S_z$.

6.2.3 Derivation of the MLE Estimator $\hat{n}_{xyz}$

Now we follow the MLE method to derive $\hat{n}_{xyz}$ given by (6–5). The derivation process is similar to the two-point case: We first derive the probability $q(n_{xyz})$ for an arbitrary bit in B_{xyz} to be '0', and use $q(n_{xyz})$ to establish the likelihood function $\mathcal{L}$ to observe U_{xyz} '0' bits in B_{xyz}. Finally, maximizing $\mathcal{L}$ with respect to n_{xyz} will give the MLE estimator, $\hat{n}_{xyz}$.

Consider an arbitrary bit b in B_{xyz}. Let A_b be the event that the bth bit in B_{xyz} remains '0', then $q(n_{xyz})$ is the probability for A_b to occur. Observed from Figure 6-1, the set of all vehicles passing R_x and/or R_y and/or R_z (i.e., $S_x \cup S_y \cup S_z$) can be partitioned into seven sets: $S_x \cap S_y \cap S_z$, $S_x \cap S_y - S_z$, $S_x \cap S_z - S_y$, $S_y \cap S_z - S_x$, $S_x - S_y - S_z$, $S_y - S_x - S_z$, and $S_z - S_x - S_y$. Consider the vehicles in each partition. It is clear that event A_b is equivalent to the combination of the following seven events:

(I) *Event H_1: For vehicles passing R_x, R_y, and R_z (i.e., in the set $S_x \cap S_y \cap S_z$), none of them have chosen bit $(b \bmod m_x)$ in B_x or bit $(b \bmod m_y)$ in B_y or bit b in B_z.* Otherwise, bit b in B_{xyz} will be '1' according to (6–4). There are n_{xyz} vehicles in the set $S_x \cap S_y \cap S_z$, and Figure 6-2 shows the decision tree for each individual car $v \in S_x \cap S_y \cap S_z$ to not set those bits. For R_x, v should choose $b_1 \bmod m_x \neq b \bmod m_x$, and the probability is clearly $1 - \frac{1}{m_x}$ (root node in Figure 6-2).

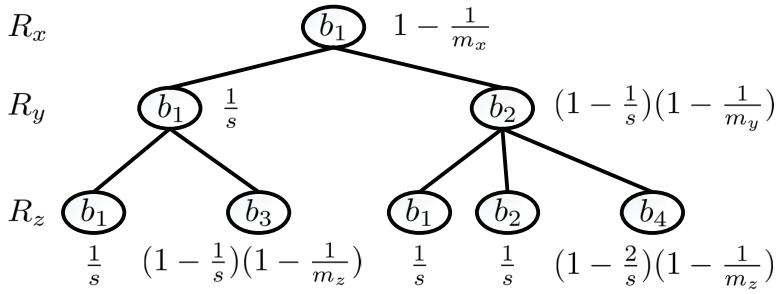


Figure 6-2. Decision tree for an arbitrary bit b in B_{xyz} to remain '0' after a car v passing by all three RSUs R_x, R_y and R_z (i.e., $v \in S_x \cap S_y \cap S_z$) sets bits in the three bit arrays (B_x, B_y, and B_z). The number inside each node represents the index that v chooses for the corresponding RSU, and the math formula next to the node represents the probability for v to choose that index, given the condition that all ancestor nodes have been chosen.

Given its selection of b_1 in RSU R_x, v has two choices in R_y: First, as shown in the left node of the second level in Figure 6-2, with a probability of $\frac{1}{s}$, v selects the same bit b_1 in R_y (hence will not set bit $b \bmod m_y$ in B_y); Second, as shown in the right node of the same level, with a probability of $1-\frac{1}{s}$, v chooses a separate bit b_2 randomly from its logical bit array LB_v, and the conditional probability for $b_2 \bmod m_y \neq b \bmod m_y$ is $1-\frac{1}{m_y}$.

Now we examine the choices for v to not set bit b in B_z of RSU R_z given its previous selections at R_x and R_y (the five nodes in the bottom level of Figure 6-2). Under its first choice at R_y, in order to not set bit b in B_z, v can either choose the same bit b_1 with probability of $\frac{1}{s}$ (node #1), or select a separate bit b_3 randomly from LB_v with a probability $1-\frac{1}{s}$, and the conditional probability for $b_3 \neq b$ is $1-\frac{1}{m_z}$ (node #2). Under its second choice at R_y, v can have three choices to not set bit b in B_z: (1) With a probability of $\frac{1}{s}$, v chooses b_1 in R_z (node #3); (2) With a probability of $\frac{1}{s}$, v chooses b_2 in R_z (node #4); (3) With a probability of $1-\frac{2}{s}$, v chooses a separate bit b_4 randomly from LB_v, and the conditional probability for $b_4 \neq b$ is $1-\frac{1}{m_z}$ (node #5).

Note that the probabilities in above analysis are all conditional probabilities given that the ancestor nodes have been chosen. To sum up, the probability of H_1 is

$$\begin{aligned} Q_1 &= \left\{\left(1-\frac{1}{m_x}\right)\left[\frac{1}{s}\times\left(1-\frac{s-1}{s}\times\frac{1}{m_z}\right)\right.\right. \\ &\quad \left.\left.+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m_y}\right)\left(1-\frac{s-2}{s}\times\frac{1}{m_z}\right)\right]\right\}^{n_{xyz}} \\ &= \left(1-\frac{1}{m_x}\right)^{n_{xyz}} C_3^{n_{xyz}}. \end{aligned} \tag{6–10}$$

(II) *Event H_2: For vehicles passing only R_x and R_y (i.e., in the set $S_x \cap S_y - S_z$), none of them have chosen bit $(b \bmod m_x)$ in B_x or bit $(b \bmod m_y)$ in B_y.* We analyze the probability of each individual vehicle to not set those two bits at R_x and R_y, which is exactly the same as that for Event E_1 of our previous two-point analysis in Section 5.2.3. Since there are $n_{xy}-n_{xyz}$ cars in the set $S_x \cap S_y - S_z$, the probability of H_2 is

$$\begin{aligned} Q_2 &= \left\{\left(1-\frac{1}{m_x}\right)\left[\frac{1}{s}+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m_y}\right)\right]\right\}^{n_{xy}-n_{xyz}} \\ &= \left(1-\frac{1}{m_x}\right)^{n_{xy}-n_{xyz}} C_4^{n_{xy}-n_{xyz}}. \end{aligned} \tag{6–11}$$

(III) *Event H_3: For vehicles passing only R_x and R_z (i.e., in the set $S_x \cap S_z - S_y$), none of them have chosen bit $(b \bmod m_x)$ in B_x or bit b in B_z.* There are $n_{xz} - n_{xyz}$ cars in the set $S_x \cap S_z - S_y$. Similar to the analysis of Event H_2 and E_1 in Section 5.2.3, we obtain the probability of H_3:

$$\begin{aligned} Q_3 &= \left\{\left(1-\frac{1}{m_x}\right)\left[\frac{1}{s}+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m_z}\right)\right]\right\}^{n_{xz}-n_{xyz}} \\ &= \left(1-\frac{1}{m_x}\right)^{n_{xz}-n_{xyz}} C_5^{n_{xz}-n_{xyz}}. \end{aligned} \tag{6–12}$$

(IV) *Event H_4: For vehicles passing only R_y and R_z (i.e., in the set $S_y \cap S_z - S_x$), none of them have chosen bit $(b \bmod m_y)$ in B_y or bit b in B_z.* There are $n_{yz} - n_{xyz}$ cars in the set $S_y \cap S_z - S_x$. Similar to the analysis of Event H_2 and E_1 in Section 5.2.3, we obtain the

probability of H_4:

$$\begin{aligned} Q_4 &= \left\{\left(1-\frac{1}{m_y}\right)\left[\frac{1}{s}+\left(1-\frac{1}{s}\right)\left(1-\frac{1}{m_z}\right)\right]\right\}^{n_{yz}-n_{xyz}} \\ &= \left(1-\frac{1}{m_y}\right)^{n_{yz}-n_{xyz}} C_5^{n_{yz}-n_{xyz}}. \end{aligned} \tag{6–13}$$

(V) *Event H_5: For vehicles passing only R_x (i.e., in the set $S_x - S_y - S_z$), none of them have chosen bit $(b \bmod m_x)$ in B_x.* There are $n_x - n_{xy} - n_{xz} + n_{xyz}$ cars in the set $S_x - S_y - S_z$, and each of them has a probability of $1 - \frac{1}{m_x}$ to not set bit $(b \bmod m_x)$ in B_x. Therefore, the probability of H_5 is

$$Q_5 = \left(1-\frac{1}{m_x}\right)^{n_x-n_{xy}-n_{xz}+n_{xyz}}. \tag{6–14}$$

(VI) *Event H_6: For vehicles passing only R_y (i.e., in the set $S_y - S_x - S_z$), none of them have chosen bit $(b \bmod m_y)$ in B_y.* There are $n_y - n_{xy} - n_{yz} + n_{xyz}$ cars in the set $S_y - S_x - S_z$, and each of them has a probability of $1 - \frac{1}{m_y}$ to not set bit $(b \bmod m_y)$ in B_y. So the probability of H_6 is

$$Q_6 = \left(1-\frac{1}{m_y}\right)^{n_y-n_{xy}-n_{yz}+n_{xyz}}. \tag{6–15}$$

(VII) *Event H_7: For vehicles passing only R_z (i.e., in the set $S_z - S_x - S_y$), none of them have chosen bit b in B_z.* There are $n_z - n_{xz} - n_{yz} + n_{xyz}$ cars in the set $S_z - S_x - S_y$, and each of them has a probability of $1 - \frac{1}{m_z}$ to not set bit b in B_z. Therefore, the probability of H_7 is

$$Q_7 = \left(1-\frac{1}{m_z}\right)^{n_z-n_{xz}-n_{yz}+n_{xyz}}. \tag{6–16}$$

Combining above analysis, we get the probability $q(n_{xyz})$ for bit b in B_{xyz} to remain '0'

$$\begin{aligned} q(n_{xyz}) &= Q_1 \times Q_2 \times Q_3 \times Q_4 \times Q_5 \times Q_6 \times Q_7 \\ &= C_3^{n_{xyz}} \times C_4^{n_{xy}-n_{xyz}} \times C_5^{n_{xz}+n_{yz}-2n_{xyz}} \\ &\quad \times \left(1-\frac{1}{m_x}\right)^{n_x} \times \left(1-\frac{1}{m_y}\right)^{n_y-n_{xy}} \\ &\quad \times \left(1-\frac{1}{m_z}\right)^{n_z-n_{xz}-n_{yz}+n_{xyz}} \end{aligned} \tag{6–17}$$

Similar to the two-point analysis, we know that for any bit in B_z, the probability for it to remain '0' after n_z vehicles each choosing a random bit from B_z is

$$q(n_z) = \left(1-\frac{1}{m_z}\right)^{n_z}, \tag{6–18}$$

and the expected values for V_z and V_{xyz} are

$$E(V_z) = E\left(\frac{U_z}{m_z}\right) = \frac{m_z \times q(n_z)}{m_z} = q(n_z), \tag{6–19}$$

$$E(V_{xyz}) = E\left(\frac{U_{xyz}}{m_z}\right) = \frac{m_z \times q(n_{xyz})}{m_z} = q(n_{xyz}). \tag{6–20}$$

In addition, similar to (5–19), we can obtain

$$\left(\frac{1-\frac{s-1}{s}\times\frac{1}{m_z}}{1-\frac{1}{m_z}}\right)^{n_{xz}} = \frac{V_{xz}}{V_x \times V_z}, \tag{6–21}$$

$$\left(\frac{1-\frac{s-1}{s}\times\frac{1}{m_z}}{1-\frac{1}{m_z}}\right)^{n_{yz}} = \frac{V_{yz}}{V_y \times V_z}. \tag{6–22}$$

Substituting (5–10), (5–11), (5–12), (5–13), (6–18), (6–19), (5–19), (6–21), (6–22), and (6–20) to (6–17), and replacing $E(V_x)$, $E(V_y)$, $E(V_z)$, and $E(V_{xyz})$, with their instance values V_x, V_y, V_z, and V_{xyz}, respectively, we have

$$V_{xyz} = \frac{V_{xy} \times V_{xz} \times V_{yz}}{V_x \times V_y \times V_z} \times \left[\frac{\left(1-\frac{1}{m_z}\right) \times C_3}{C_4 \times C_5^2}\right]^{n_{xyz}}. \tag{6–23}$$

Finally, solving (6–23) gives the MLE estimator $\hat{n}_{xyz}$ as described in (6–5).

6.2.4 Computation Overhead

We now analyze the computation overhead for each group of entity. Note that the online coding phase works exactly the same as our two-point scheme, so the computation overhead for the vehicles and RSUs of our three-point scheme is exactly the same as the two-point scheme. For both schemes, when a vehicle v passes an RSU R_x, v only needs to compute two hashes to obtain an index of a random bit, and R_x only needs to set 1 bit in its bit array B_x. So the computation overhead for each vehicle per RSU as well as for each RSU per passing vehicle are both $O(1)$.

Our three-point scheme and two-point scheme diverge from the offline decoding phase, where the central server performs a little bit more task for three-point traffic measurement: it takes four "unfolding" and bitwise OR operations (Section 6.2.2) instead of one such operation (Section 5.2.2). Similar to our two-point analysis in Section 5.2.4, in our three-point scheme, the "unfolding" and bitwise OR operation in step 1 costs $O(m_y)$ time, and step 2, 3, and 4 each costs $O(m_z)$ time, leading to an overall computation overhead of $O(m_z)$, where m_z is the size of the largest bit array among the three RSUs. One can see that our three-point traffic measurement scheme is also very efficient.

6.2.5 Preserved Privacy

Since the way RSUs collect information from passing vehicles in our three-point scheme is no different from our two-point scheme, the preserved privacy is also the same. Clearly, the first-level privacy is preserved. In addition, for both schemes, the second-level privacy p, satisfying the requirement that the probability for any "trace" of any vehicle not to be identified must be at least p, is actually the conditional probability that states to what degree observing a same bit to be set in both bit arrays of two RSUs does not represent a common vehicle passing by both RSUs (i.e., a piece of a vehicle's trace). The reason is that the only information a vehicle v ever reports to an RSU is a bit index drawn from the same common pool uniformly at random, and the adversary can only attempt to identify the trace of a vehicle

through the observation of the bits that are chosen by the vehicles to be set as '1' in both RSUs. Therefore, the second-level privacy of our three-point scheme is also given by (5–46), with same outstanding conclusions as given in Section 5.4 of the two-point scheme.

6.3 Simulation

We conduct two sets of simulations to evaluate the measurement accuracy of our three-point scheme. Note if we set $m_x = m_y = m_z = m$ in (6–5), we can get the MLE formula for $\hat{n}_{xyz}$ under the setting of fixed bit array size m for all RSUs. Since we have compared our two-point schemes with two different settings, fixed bit array size m as in [47] v.s. fixed load factor f as in [49], we also evaluate our three-point scheme under the two different settings, fixed m v.s. fixed f.

The first set of simulations is to observe the accuracy of our three-point scheme when the single-point traffic volume of three RSUs are comparable, which means the two settings, fixed m and fixed f, are now equivalent. The simulations are controlled by the following parameters: n_x, n_y, n_z, n_{xyz}, s, and m (f). Their values are chosen as follows: $n_x = n_y = n_z = n$, where $n = 50,000$, $100,000$, or $500,000$, and n_{xyz} varies from $0.01n$ to $0.5n$, with a step size of $0.001n$; $s = 2, 5, 10$, and $m_x = m_y = m_z = m$ ($f_x = f_y = f_z = f$) is chosen to achieve the optimal privacy p according to (5–46).

Fig. 6-3, Fig. 6-4, and Fig. 6-5 show our simulation results when $n = 50,000$, $100,000$, and $500,000$, respectively. One can see that our three-point scheme is quite accurate under $s = 2$ (the measured three-point traffic volume $\hat{n}_{xyz}$ closely follows its real value n_{xyz} in the first plot of the three figures). With the increment of s, the measurement results slightly diverge from their real values (refer to the last plot of the three figures), which means larger values of s will bring in less accurate measurement results. This conclusion is similar to what we get from the two-point traffic measurement scheme in [47]. Intuitively, if a vehicle v has a larger logical bit array, the chance for it to report the same bit index to different RSUs decreases, which means the common information collected by different RSUs is reduced. Therefore, the accuracy will also be affected for both the two-point and the three-point

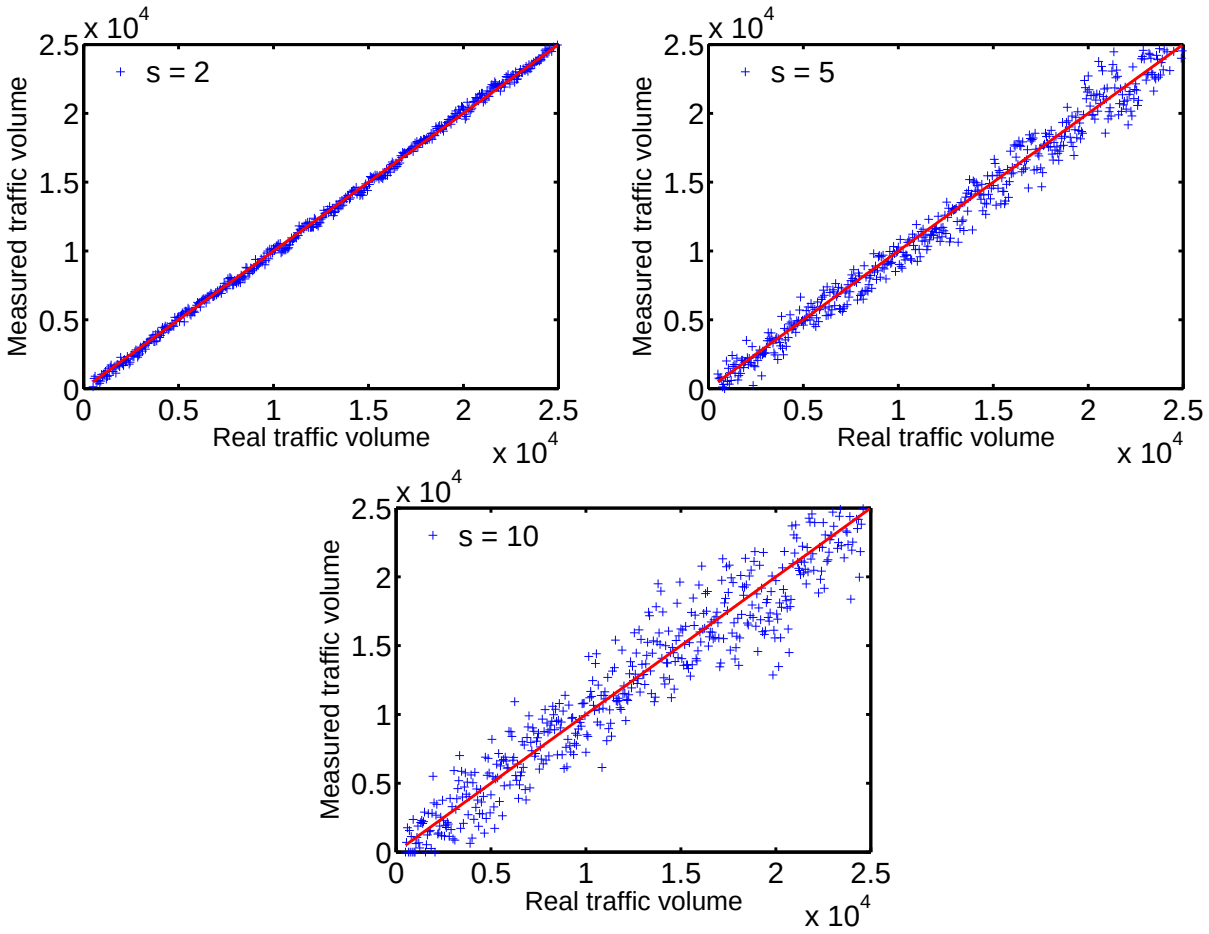


Figure 6-3. Measurement accuracy with optimal privacy, $n_x = n_y = n_z = n = 50,000$, $n_{xyz} = [0.01n, 0.5n]$. The x-axis shows real three-point traffic volume, and the y-axis shows the measured three-point traffic volume. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

measurement. One can also observe that the measurement accuracy of our three-point scheme improves along with the increment of n (compare each plot of Fig. 6-3 with Fig. 6-5), which is a natural phenomenon since our estimator is derived from the statistical MLE method.

The second set of simulations is to observe the measurement accuracy of our three-point scheme when the single-point traffic volume of three RSUs may differ. Under the circumstances where RSUs' traffic volume are not the same, will the two settings, fixed m and fixed f, begin to show differences as we expected? If so, how will the gap between RSUs' single-point traffic volume influence the performance of our scheme under the two different settings? These are the questions to investigate.

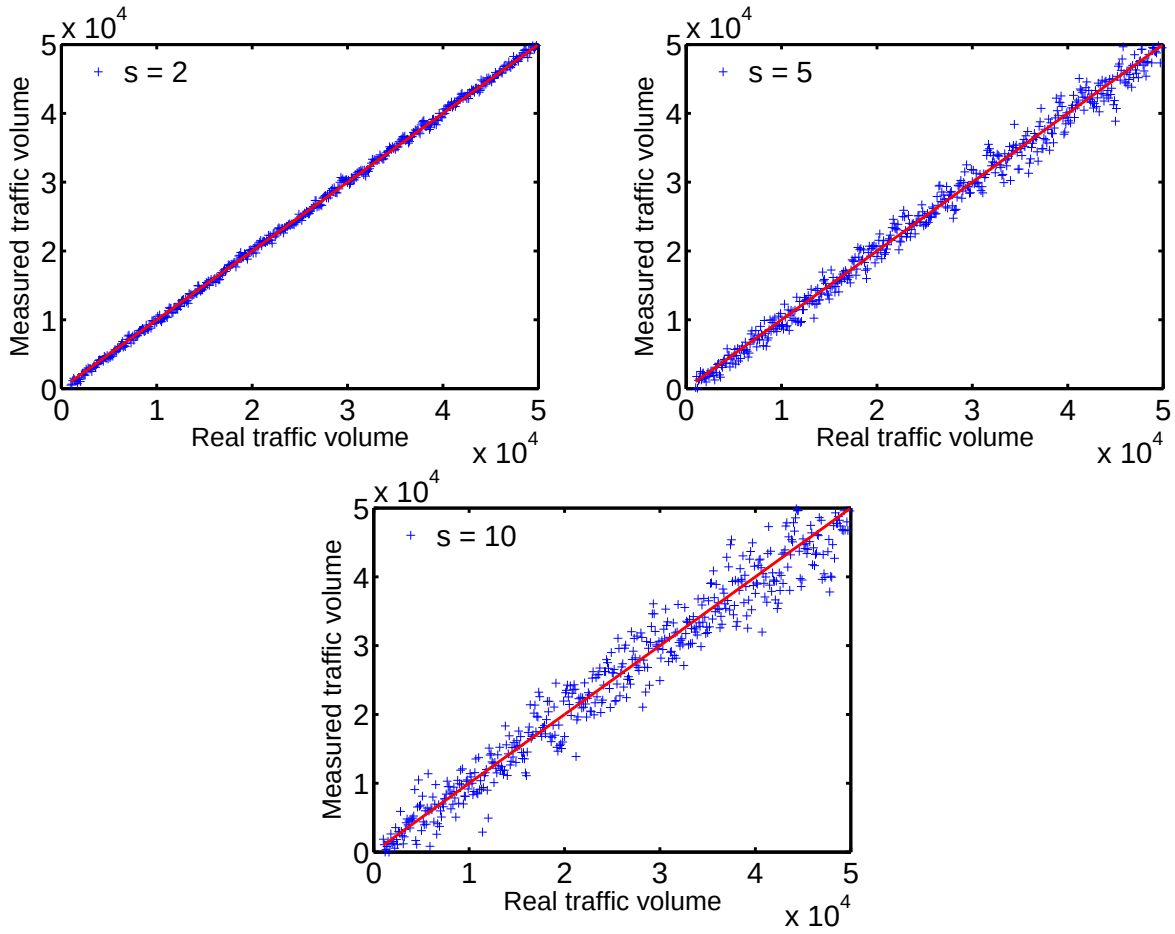


Figure 6-4. Measurement accuracy with optimal privacy, $n_x = n_y = n_z = n = 100,000$, $n_{xyz} = [0.01n, 0.5n]$. The x-axis shows real three-point traffic volume, and the y-axis shows the measured three-point traffic volume. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

Bearing these questions in mind, the second set of simulations are controlled by the following parameters: n_x, n_y, n_z, n_{xyz}, s, f, and m. Their values are chosen as follows: $n_x = 10,000$, $n_z = n_y = n_x$ or $n_z = 4n_y = 16n_x$ or $n_z = 8n_y = 64n_x$, n_{xyz} varies from $0.01n_x$ to $0.5n_x$, with step size of $0.001n_x$. s is set to $2, 5, 10$. m is the fixed bit array size for all RSUs under the first setting, and f is the fixed load factor for all RSUs under the second setting. The values of m and f are chosen to guarantee a minimum privacy of at least 0.5 under the two settings, respectively.

Fig. 6-6 shows the simulation results for our three-point scheme with fixed bit array size m, and Fig. 6-7 shows the results for our three-point scheme with fixed load factor f, both

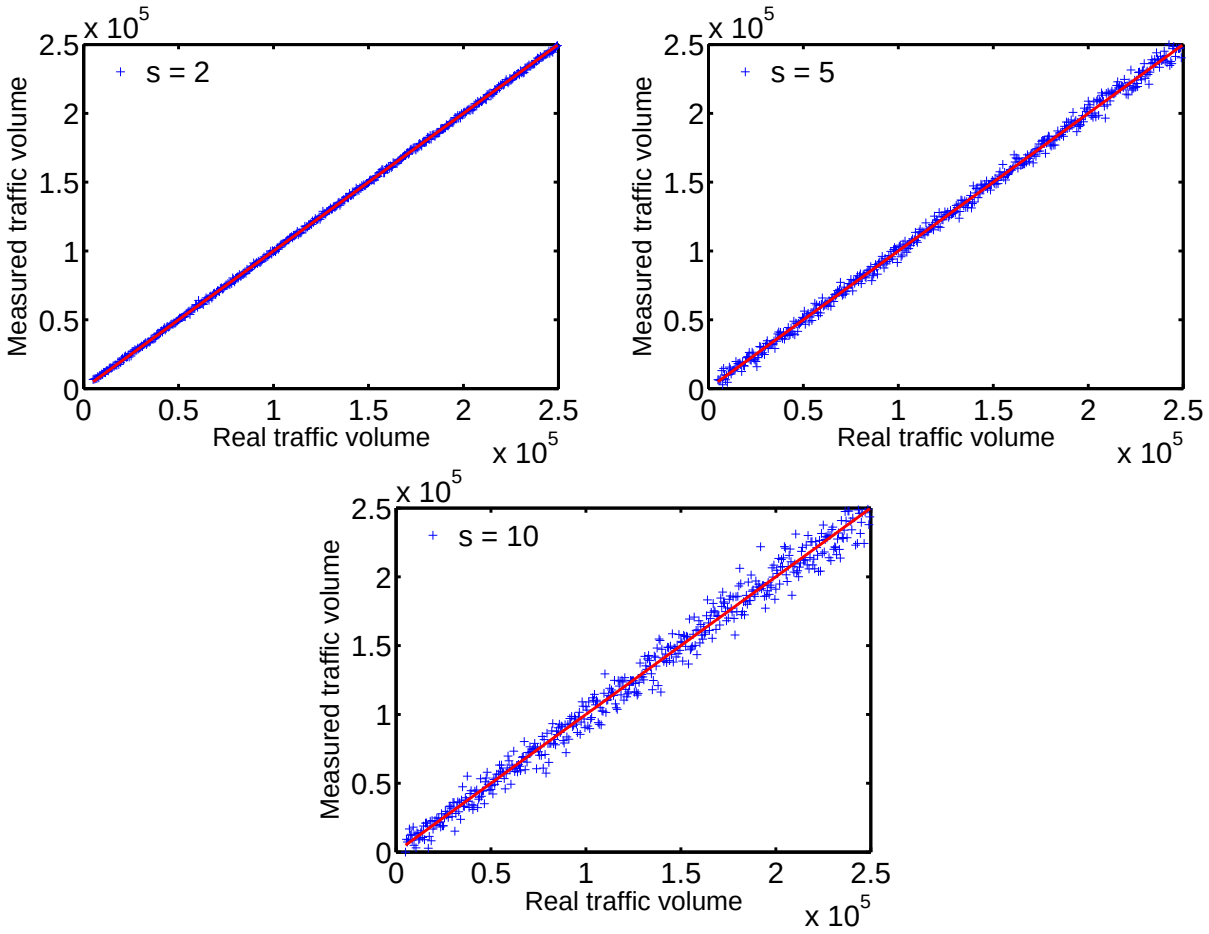


Figure 6-5. Measurement accuracy with optimal privacy, $n_x = n_y = n_z = n = 500,000$, $n_{xyz} = [0.01n, 0.5n]$. The x-axis shows real three-point traffic volume, and the y-axis shows the measured three-point traffic volume. The three plots are controlled by s. *First Plot*: $s = 2$; *Second Plot*: $s = 5$; *Third Plot*: $s = 10$.

under $s = 2$. Since the comparison results for $s = 5$ and $s = 10$ are quite similar, here we omit them. From the two figures, one can observe two key trends: (1) When the single-point traffic volume for the three RSUs are comparable, i.e., $n_z = n_y = n_x$, our three-point scheme under the two settings, fixed m and fixed f, indeed achieves equivalent accuracy (first plot of Fig. 6-6 and Fig. 6-7); (2) When the single-point traffic volume vary for different RSUs, our three-point scheme achieves far better accuracy under the fixed f setting than the fixed m setting, and the performance difference enlarges with the widening of the gap among the three RSUs' single-point traffic volume (the second and third plot of Fig. 6-6 v.s. Fig. 6-7). The two

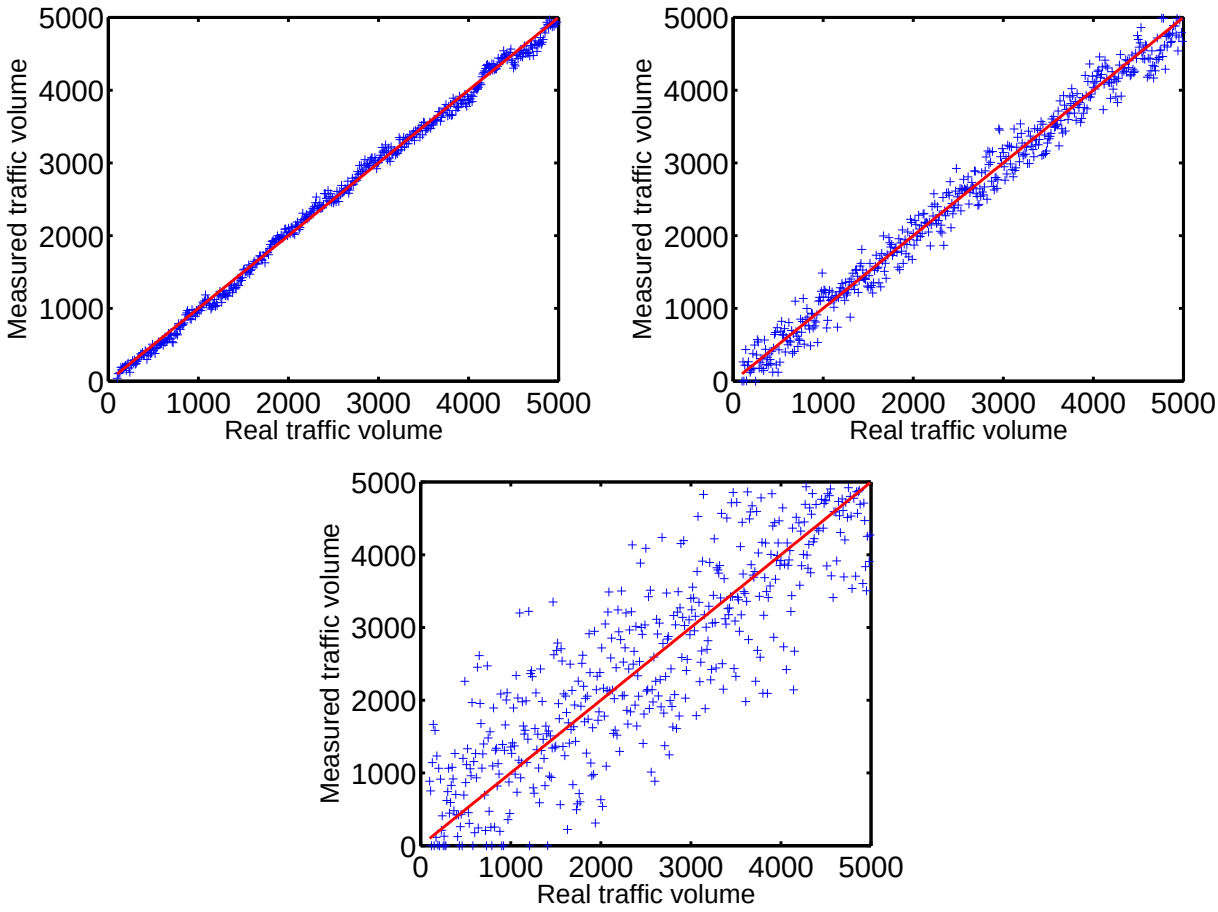


Figure 6-6. Measurement accuracy of our three-point scheme with fixed bit array size m. The x-axis shows real three-point traffic volume, and the y-axis shows measured three-point traffic volume. $s = 2$, $n_x = 10,000$, $n_{xyz} = [0.01n_x, 0.5n_x]$. The three plots are controlled by the ratio of n_y and n_z over n_x. *First Plot*: $n_z = n_y = n_x$; *Second Plot*: $n_z = 4n_y = 16n_x$; *Third Plot*: $n_z = 8n_y = 64n_x$.

trends observed from the measurement results of our three-point scheme also coincide with those shown in our two-point scheme.

6.4 Summary

In this chapter, we focus on addressing the problem of privacy-preserving three-point traffic measurement in CPRS, whose goal is to automatically collect and efficiently measure the traffic passing three arbitrary RSUs while preserving the privacy of vehicles. As far as we know, this is the first study of the privacy-preserving multi-point traffic measurement problem which measures traffic passing through more than two locations while preserving vehicles' privacy

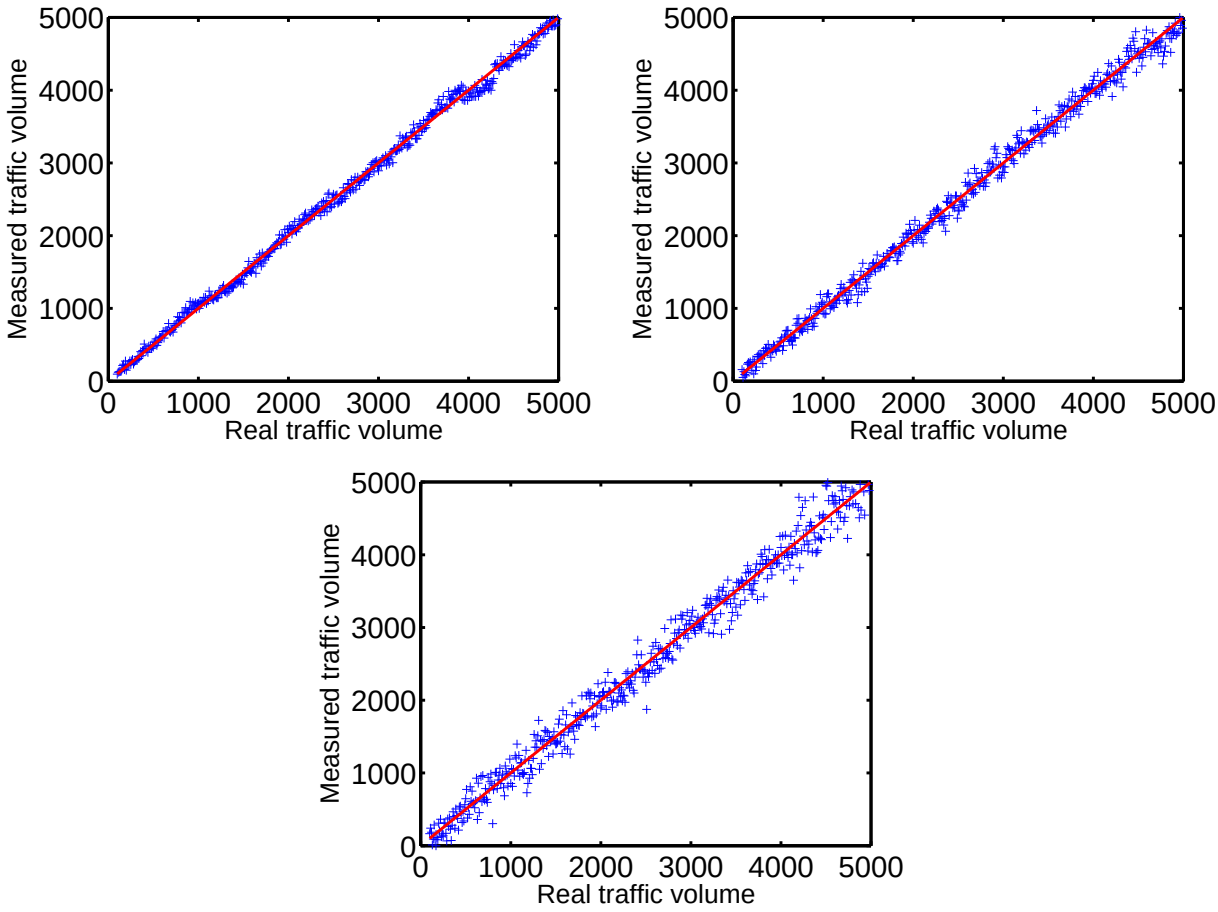


Figure 6-7. Measurement accuracy of our three-point scheme with fixed load factor f. The x-axis shows real three-point traffic volume, and the y-axis shows measured three-point traffic volume. $s = 2$, $n_x = 10,000$, $n_{xyz} = [0.01n_x, 0.5n_x]$. The three plots are controlled by the ratio of n_y and n_z over n_x. *First Plot*: $n_z = n_y = n_x$; *Second Plot*: $n_z = 4n_y = 16n_x$; *Third Plot*: $n_z = 8n_y = 64n_x$.

in a road system. In this work [50], we extend and generalize over our previous best scheme on privacy-preserving two-point traffic measurement [47], and propose a novel scheme for privacy-preserving three-point traffic measurement, which also utilizes variable-length bit array masking to fit in the real-life situations where different RSUs observe varied amount of traffic. In this chapter, we present our novel three-point traffic measurement scheme, and demonstrate its applicability, efficiency, and scalability through mathematical and numerical analysis as well as extensive simulations. During our course of research, we also observe the potential for our design of variable-length bit array masking to be further extended to solve the more general

problem of privacy-preserving multi-point traffic measurement. In the next chapter, we will investigate that possibility.

CHAPTER 7
PRIVACY-PRESERVING MULTI-POINT TRAFFIC MEASUREMENT

In the previous two chapters, we have proposed two schemes for privacy-preserving two-point and three-point traffic measurement based on variable-length bit array masking, which can efficiently measure the traffic volume among an arbitrary set of two or three RSUs, and well preserve vehicles' privacy. During the course of our research, we observe the potential for our design of variable-length bit array masking to be further generalized to measure traffic covering $d > 2$ locations. In this chapter, we will investigate the feasibility of this generalization, and propose a general framework for the problem of privacy-preserving multi-point traffic measurement, which can efficiently measure the traffic volume among an arbitrary of d locations and preserve vehicles' privacy. We will also discuss the performance of the general framework as d increases.

7.1 General Framework

Similar to our two-point and three-point scheme based on variable-length bit array masking as presented in the previous two chapters, our general framework to measure d-point traffic volume also includes two phases: online coding phase for RSUs to collect de-identified vehicle information through variable-length bit arrays, and offline decoding phase for the central server to compute the d-point traffic volume among an arbitrary set of d RSUs based on the variable-length bit arrays. The online coding phase works exactly the same as our two-point (Section 5.2.1) and three-point (Section 6.2.1) scheme, which we omit to avoid duplicate description.

The offline decoding phase is also similar. At the end of each measurement period, all RSUs will send their counters and bit arrays to the central server. To compute the d-point traffic volume among an arbitrary set of d RSUs, denoted as $\{R_1, ..., R_d\}$, the central server will perform a series of "unfolding" and bitwise OR operations in between the bit arrays of the d RSUs to generate a series of statistical results (more specifically, the zero ratios of the resulting bit arrays) that are related to the d-point traffic volume. Again, if an MLE estimator

can be derived based on these statistical results, the central server can easily compute the d-point traffic volume. Therefore, the key is to establish the relationship between the zero ratios of the bitwise ORed bit arrays and the d-point traffic volume.

Before deriving this relationship, we first define some notations. We denote the set of d RSUs as $\mathcal{S}_d$, i.e., $\mathcal{S}_d = \{R_1, ..., R_d\}$. Without loss of generality, we assume $m_1 \leq m_2 \leq ... \leq m_d$, where m_i is the size of the bit array B_i in R_i, $1 \leq i \leq d$. For an arbitrary set $\mathcal{S} \subset \mathcal{S}_d$ of RSUs, we unfold their bit arrays to the same size of the largest bit array among $\mathcal{S}$, and perform a bitwise OR operation over the unfolded bit arrays to obtain a new bit array $B_{\mathcal{S}}$, whose zero ratio is $V_{\mathcal{S}}$. Denote the set of vehicles that pass by all RSUs in $\mathcal{S}$ as $\mathcal{V}_{\mathcal{S}}$ with cardinality $\mathcal{N}_{\mathcal{S}} = |\mathcal{V}_{\mathcal{S}}|$. Clearly, we want to measure $\mathcal{N}_{\mathcal{S}_d}$.

Given an arbitrary bit b in $B_{\mathcal{S}}$, the probability for b to be '0' after an arbitrary vehicle $v \in \mathcal{V}_{\mathcal{S}}$ marks bits for all RSUs in $\mathcal{S}$ is denoted as $P_{\mathcal{S}}$. Similar to our two-point and three-point scheme, we can derive the overall probability $q(\mathcal{N}_{\mathcal{S}_d})$ for an arbitrary bit b in $B_{\mathcal{S}_d}$ to be '0' after online coding as

$$\begin{aligned} q(\mathcal{N}_{\mathcal{S}_d}) = P_{\mathcal{S}_d}^{\mathcal{N}_{\mathcal{S}_d}} \times \prod_{1\leq i\leq d} P_{\mathcal{S}_d-\{R_i\}}^{\mathcal{N}_{\mathcal{S}_d-\{R_i\}}-\mathcal{N}_{\mathcal{S}_d}} \times \\ \prod_{1\leq i<j\leq d} P_{\mathcal{S}_d-\{R_i,R_j\}}^{\mathcal{N}_{\mathcal{S}_d-\{R_i,R_j\}}-\mathcal{N}_{\mathcal{S}_d-\{R_i\}}-\mathcal{N}_{\mathcal{S}_d-\{R_j\}}+\mathcal{N}_{\mathcal{S}_d}} \times \cdots \times \\ \prod_{1\leq i\leq d} P_{\{R_i\}}^{\mathcal{N}_{\{R_i\}}-\sum_{1\leq j\leq d}^{j\neq i}\mathcal{N}_{\{R_i,R_j\}}+\cdots+(-1)^{d-1}\mathcal{N}_{\mathcal{S}_d}}, \end{aligned} \tag{7–1}$$

where each term above captures the probability for bit b in $B_{\mathcal{S}_d}$ to be '0' after the set of vehicles passing only l ($d \geq l \geq 1$) RSUs in $\mathcal{S}_d$ mark bits in the bit arrays, and the superscript in each term denotes the corresponding vehicle set cardinality derived from inclusion-exclusion principle.

Given above analysis, we present Algorithm 7.1 to iteratively derive the MLE estimator $\hat{\mathcal{N}}_{\mathcal{S}_d}$, whose correctness can be easily proved through mathematical induction, which we omit.

In Algorithm 7.1, the inputs P_1, P_2, and P_3 are probability formulas given in (5–6), (5–7), and (5–8) in our two-point MLE derivation, with the notations n_x, n_y, and n_{xy} changed to

$\mathcal{N}_{\{\mathcal{R}_1\}}$, $\mathcal{N}_{\{\mathcal{R}_2\}}$, and $\mathcal{N}_{\{\mathcal{R}_1,\mathcal{R}_2\}}$, respectively. We first initialize the probability set $\mathcal{I}_{P_2}$ and the vehicle cardinality set $\mathcal{I}_{\mathcal{N}_2}$ from the two-point derivation, which serves as the base case of our iterative algorithm. Then the for-loop works iteratively, where the iteration j derives $\mathcal{I}_{P_{j+1}}$ and $\mathcal{I}_{\mathcal{N}_{j+1}}$ based on $\mathcal{I}_{P_j}$ and $\mathcal{I}_{\mathcal{N}_j}$ obtained from the previous iteration. Note that $\mathcal{I}_{\mathcal{N}_{j+1}}$ includes the MLE estimator $\hat{\mathcal{N}}_{\mathcal{S}_{j+1}}$ as a function $\mathcal{F}_{j+1}(\{V^*_{\mathcal{S}_{j+1}}\})$ of the zero ratios, where the set $\{V^*_{\mathcal{S}_{j+1}}\}$ contains the zero ratio $V_{\mathcal{S}}$ of $B_{\mathcal{S}}$ for all $\mathcal{S} \subset \mathcal{S}_{j+1}$, $\mathcal{S} \neq \emptyset$. Therefore, when the for-loop completes, we will obtain the MLE estimator $\hat{\mathcal{N}}_{\mathcal{S}_d}$ as a function $\mathcal{F}_d(\{V^*_{\mathcal{S}_d}\})$ of the zero ratios in the corresponding bitwise ORed bit arrays.

Algorithm 7.1. *Iterative Algorithm to Derive the MLE estimator* $\hat{\mathcal{N}}_{\mathcal{S}_d}$

1: **Inputs:** d, P_1, P_2, P_3, $\{m_i\}_{1\leq i\leq d}$, $\{\mathcal{N}_{\{\mathcal{R}_i\}}\}_{1\leq i\leq d}$, $\hat{\mathcal{N}}_{\mathcal{S}_2}$

2: **Initialize:** $P_{\mathcal{S}_2} \leftarrow P_1$, $P_{\{\mathcal{R}_1\}} \leftarrow P_2$, $P_{\{\mathcal{R}_2\}} \leftarrow P_3$, $\mathcal{I}_{P_2} \leftarrow \{P_{\mathcal{S}_2}, P_{\{\mathcal{R}_1\}}, P_{\{\mathcal{R}_2\}}\}$

$\mathcal{N}_{\mathcal{S}_2} \leftarrow \hat{\mathcal{N}}_{\mathcal{S}_2}$, $\mathcal{I}_{\mathcal{N}_2} \leftarrow \{\mathcal{N}_{\mathcal{S}_2}, \mathcal{N}_{\{\mathcal{R}_1\}}, \mathcal{N}_{\{\mathcal{R}_2\}}\}$

3: **for** $j \leftarrow 2$ *to* $d-1$ **do**

4: **Step 1:** *Use decision tree as Figure 5-2 and Figure 6-2 to obtain* $P_{\mathcal{S}_{j+1}}$

5: **Step 2:** *Use* $P_{\mathcal{S}_{j+1}}$ *and* $\mathcal{I}_{P_j} = \{P_{\mathcal{S}_j}\} \bigcup \{P_{\mathcal{S}_j - \{R_i\}}\}_{1\leq i\leq j} \bigcup \cdots \bigcup \{P_{\{R_i\}}\}_{1\leq i\leq j}$

to update $\mathcal{I}_{P_{j+1}} = \{P_{\mathcal{S}_{j+1}}\} \bigcup \{P_{\mathcal{S}_{j+1} - \{R_i\}}\}_{1\leq i\leq j+1} \bigcup \cdots \bigcup \{P_{\{R_i\}}\}_{1\leq i\leq j+1}$

6: **Step 3:** *Use* $\mathcal{I}_{\mathcal{N}_j} = \{\mathcal{N}_{\mathcal{S}_j}\} \bigcup \{\mathcal{N}_{\mathcal{S}_j - \{R_i\}}\}_{1\leq i\leq j} \bigcup \cdots \bigcup \{\mathcal{N}_{\{R_i\}}\}_{1\leq i\leq j}$

to update $\mathcal{I}_{\mathcal{N}_{j+1}} - \{\mathcal{N}_{\mathcal{S}_{j+1}}\} = \{\mathcal{N}_{\mathcal{S}_{j+1} - \{R_i\}}\}_{1\leq i\leq j+1} \bigcup \cdots \bigcup \{\mathcal{N}_{\{R_i\}}\}_{1\leq i\leq j+1}$

7: **Step 4:** *Use* $\mathcal{I}_{P_{j+1}}$, $\mathcal{I}_{\mathcal{N}_{j+1}} - \{\mathcal{N}_{\mathcal{S}_{j+1}}\}$, *and (7–1), and replace* $q(\mathcal{N}_{\mathcal{S}_{j+1}}) = E(V_{\mathcal{S}_{j+1}})$

by its instance value $V_{\mathcal{S}_{j+1}}$, *to get the MLE estimator* $\hat{\mathcal{N}}_{\mathcal{S}_{j+1}} = \mathcal{F}_{j+1}(\{V^*_{\mathcal{S}_{j+1}}\})$

8: **Step 5:** $\mathcal{N}_{\mathcal{S}_{j+1}} \leftarrow \hat{\mathcal{N}}_{\mathcal{S}_{j+1}}$, $\mathcal{I}_{\mathcal{N}_{j+1}} \leftarrow \mathcal{I}_{\mathcal{N}_{j+1}} - \{\mathcal{N}_{\mathcal{S}_{j+1}}\} \bigcup \{\mathcal{N}_{\mathcal{S}_{j+1}}\}$

9: **end for**

7.2 Discussion

We conclude with a quick discussion about the performance of our general d-point ($d > 1$) traffic measurement scheme. Clearly, since RSUs collect de-identified information from passing vehicles in the same way as our two-point and three-point scheme, the preserved privacy is also the same. Also, the computation overhead for vehicles and RSUs remains $O(1)$. However, as d increases, the computation overhead for the central server to measure d-point traffic

volume grows exponentially. Given d bit arrays of d RSUs, the central server needs to perform unfolding and bitwise OR on every l ($2 \leq l \leq d$) bit arrays to generate $2^d - d - 1$ new bit arrays, and compute the zero ratios in them and d original bit arrays, which costs an overall of $O(2^d \times m_d)$ time.

In addition, as d increases, the measurement accuracy of our general scheme is expected to decrease. The reason is that, for each iteration j of the MLE derivation, an instance value of zero ratio $V_{\mathcal{S}_{j+1}}$ replaces its expected value $q(\mathcal{N}_{\mathcal{S}_{j+1}}) = E(V_{\mathcal{S}_{j+1}})$ to get the MLE estimator $\hat{\mathcal{N}}_{\mathcal{S}_{j+1}}$, which introduces a certain level of inaccuracy. This inaccuracy will accumulate as d increases. When d exceeds some value, say 10, our d-point scheme may not work well as our current two-point and three-point scheme. However, in reality, the d-point traffic of interest usually has small values of d, such as 2, 3, or 4. Therefore, our general scheme is still sufficient to serve for most applications.

CHAPTER 8
CONCLUSION

In this dissertation, we focus on the important problem of privacy-preserving multi-point traffic volume measurement in intelligent cyber-physical road systems (CPRS), which complements the state of art mainly focused on single-point traffic volume measurement. We propose several novel schemes to allow transportation authorities to automatically collect and efficiently measure the aggregate multi-point traffic volume data from CPRS without learning information about individual vehicles.

In the dissertation, we first propose four novel schemes to address the problem of privacy-preserving two-point traffic measurement, with varying degrees of efficiency, accuracy, and privacy. Our first two schemes protect vehicles' identities through keyed signatures based on a family of commutative one-way hash functions, and they can achieve exact measurement results. To further improve the measurement efficiency and achieve better privacy for vehicles, we utilize a novel compact data structure, shared bit arrays, to propose a third measurement scheme. It is much more efficient, and protects not only vehicles' identities but also their travelling trajectory. The scheme can gracefully control the tradeoff between vehicles' privacy and measurement accuracy under the assumption that different locations observe similar amount of traffic. To remove this similar traffic assumption to fit in more realistic situations, we propose our fourth measurement scheme, which is based on variable-length bit array masking, a novel "unfolding" technique, and the rigorous statistical MLE method. Our fourth scheme achieves better privacy for vehicles, more accurate measurement results, and comparable computation overhead, compared with the previous best scheme.

To suit for a broader spectrum of applications in vehicular networks and transportation engineering, we naturally extend our idea of variable-length bit array masking to solve the problem of privacy-preserving three-point traffic measurement, and eventually present a framework to solve the general problem of privacy-preserving multi-point traffic measurement. We demonstrate the feasibility, scalability, and superior performance of our solutions through

mathematical proofs, numerical analysis, as well as extensive simulations. The research results in this dissertation have potential applications beyond vehicular networks, such as privacy-preserving traffic estimation in a subway system with tagged toll cards. It is also possible for them to be used for estimating the movement patterns of mobile users in a corporate wireless network.

REFERENCES

[1] USDOT, "Traffic Monitoring Guide," 2013. [Online]. Available: http://www.fhwa.dot.gov/policyinformation/tmguide

[2] D. Mohamad, K. Sinha, T. Kuczek, and C. Scholer, "Annual Average Daily Traffic Prediction Model for County Roads," *Journal of the Transportation Research Board*, vol. 1617/1998, pp. 69–77, 1998.

[3] W. H. Lam and J. Xu, "Estimation of AADT from Short Period Counts in Hong Kong – A Comparison Between Neural Network Method and Regression Analysis," *Journal of Advanced Transportation*, vol. 34, no. 2, pp. 249–268, 2000.

[4] J. Eom, M. Park, T.-Y. Heo, and L. Huntsinger, "Improving the Prediction of Annual Average Daily Traffic for Nonfreeway Facilities by Applying a Spatial Statistical Method," *Journal of the Transportation Research Board*, vol. 1968/2006, pp. 20–29, 2006.

[5] M. Castro-Neto, Y. Jeong, M. K. Jeong, and L. D. Han, "AADT Prediction using Support Vector Regression with Data-Dependent Parameters," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2979–2986, March 2009.

[6] B. Yang, S.-G. Wang, and Y. Bao, "Efficient Local AADT Estimation via SCAD Variable Selection Based on Regression Models," *Control and Decision Conference*, pp. 1898–1902, May 2011.

[7] I. Tsapakis, W. H. Schneider, and A. P. Nichols, "A Bayesian Analysis of the Effect of Estimating Annual Average Daily Traffic for Heavy-Duty Trucks using Training and Validation Data-Sets," *Transportation Planning and Technology*, vol. 36, no. 2, pp. 201–217, March 2013.

[8] J. Eriksson, H. Balakrishnan, and S. Madden, "Cabernet: Vehicular Content Delivery Using WiFi," *Proc. of MOBICOM*, pp. 199–210, 2008.

[9] U. Lee, J. Lee, J.-S. Park, and M. Gerla, "FleaNet: A Virtual Market Place on Vehicular Networks," *IEEE Trans. on Vehicular Technology*, vol. 59, no. 1, pp. 344–355, 2010.

[10] Y. L. Morgan, "Notes on DSRC & WAVE Standards Suite: Its Architecture, Design, and Characteristics," *IEEE Comm. Surveys & Tutorials*, vol. 12, no. 4, pp. 504–518, 2010.

[11] A. Amanna, "Overview of IntelliDrive / Vehicle Infrastructure Integration (VII)," *VirginiaTech Transportation Institute*, 2009.

[12] USDOT, "United States Department of Transportation," 2015. [Online]. Available: http://www.dot.gov/

[13] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected Vehicles: Solutions and Challenges," *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 289–299, 2014.

[14] M. Pan, P. Li, and Y. Fang, "Cooperative Communication Aware Link Scheduling for Cognitive Vehicular Ad-hoc Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 30, no. 4, pp. 760–768, 2012.

[15] N. Lu, N. Zhang, N. Cheng, X. Shen, J. W. Mark, and F. Bai, "Vehicles Meet Infrastructure: Towards Capacity-Cost Tradeoffs for Vehicular Access Networks," *IEEE Trans. on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1266–1277, 2013.

[16] J. Sun, C. Zhang, Y. Zhang, and Y. Fang, "An Identity-based Security System for User Privacy in Vehicular Ad Hoc Networks," *IEEE Trans. on Parallel and Distributed Systems (TPDS)*, vol. 21, no. 9, pp. 1227–1239, 2010.

[17] Y. Zhu, Y. Wu, and B. Li, "Vehicular Ad Hoc Networks and Trajectory-Based Routing," *Internet of Things*, pp. 143–167, 2014.

[18] X. Zhu, S. Jiang, L. Wang, and H. Li, "Efficient Privacy-Preserving Authentication for Vehicular Ad Hoc Networks," *IEEE Trans. on Vehicular Technology*, vol. 63, no. 2, pp. 907–919, 2014.

[19] R. Du, C. Chen, B. Yang, N. Lu, X. Guan, and X. Shen, "Effective Urban Traffic Monitoring by Vehicular Sensor Networks," *IEEE Trans. on Vehicular Technology*, vol. 64, no. 1, pp. 273–286, 2015.

[20] Y. Lou and Y. Yin, "A Decomposition Scheme for Estimating Dynamic Origin-destination Flows on Actuation-controlled Signalized Arterials," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 5, pp. 643–655, 2010.

[21] Google, "Google Maps Now Includes Real-Time Traffic Data," 2012. [Online]. Available: http://mashable.com/2012/03/29/google-maps-traffic-data/

[22] T. Jeske, "Floating Car Data from Smartphones: What Google and Waze Know About You and How Hackers Can Control Traffic," *Proc. of the BlackHat Europe*, 2013.

[23] Y. Jian, S. Chen, Z. Zhang, and L. Zhang, "A Novel Scheme for Protecting Receiver's Location Privacy in Wireless Sensor Networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 10, pp. 3769–3779, October 2008.

[24] M. Zhang, V. Khanapure, S. Chen, and X. Xiao, "Memory Efficient Protocols for Detecting Node Replication Attacks in Wireless Sensor Networks," *Proc. of IEEE ICNP*, pp. 284–293, October 2009.

[25] M. Zhang, S. Chen, and Y. Jian, "MAC-layer Time Fairness across Multiple Wireless LANs," *Proc. of IEEE INFOCOM*, pp. 1370–1378, March 2010.

[26] Y. Jian, M. Zhang, and S. Chen, "Achieving MAC-Layer Fairness in CSMA/CA Networks," *IEEE/ACM Trans. on Networking*, vol. 19, no. 5, pp. 1472–1484, October 2011.

[27] L. Zhang, W. Luo, S. Chen, and Y. Jian, "End-to-End Maxmin Fairness in Multihop Wireless Networks: Theory and Protocol," *Journal of Parallel and Distributed Computing*, vol. 72, no. 3, pp. 462–474, March 2012.

[28] M. Yoon, T. Li, S. Chen, and J. kwon Peir, "Fit a Spread Estimator in Small Memory," *Proc. of IEEE INFOCOM*, pp. 504–512, April 2009.

[29] ——, "Fit a Compact Spread Estimator in Small High-Speed Memory," *IEEE/ACM Trans. on Networking*, vol. 19, no. 5, pp. 1253–1264, October 2011.

[30] T. Li and S. Chen, *Traffic Measurement on the Internet*. Springer, 2012.

[31] T. Li, S. Chen, and Y. Ling, "Per-Flow Traffic Measurement through Randomized Counter Sharing," *IEEE/ACM Trans. on Networking*, vol. 20, no. 5, pp. 1622–1634, October 2012.

[32] T. Li, S. Chen, W. Luo, M. Zhang, and Y. Qiao, "Spreader Classification Based on Optimal Dynamic Bit Sharing," *IEEE/ACM Trans. on Networking*, vol. 21, no. 3, pp. 817–830, 2013.

[33] Q. Xiao, Y. Qiao, Z. Mo, and S. Chen, "Estimating the Persistent Spreads in High-speed Networks," *Proc. of IEEE ICNP*, pp. 131–142, October 2014.

[34] Q. Xiao, S. Chen, M. Chen, and Y. Ying, "Hyper-Compact Virtual Estimators for Big Network Data Based on Register Sharing," *Proc. of ACM SIGMETRICS*, 2015.

[35] Q. Xiao, M. Chen, S. Chen, and Y. Zhou, "Temporally or Spatially Dispersed Joint RFID Estimation Using Snapshots of Variable Lengths," *Proc. of ACM Mobihoc*, June 2015.

[36] M. Yoon and S. Chen, "Real-Time Detection of Invisible Spreaders," *Proc. of IEEE Globecom*, pp. 2109–2113, November 2008.

[37] T. Li, W. Luo, Z. Mo, and S. Chen, "Privacy-preserving RFID Authentication based on Cryptographical Encoding," *Proc. of IEEE INFOCOM*, pp. 2174–2182, March 2012.

[38] M. Chen, S. Chen, and Q. Xiao, "Pandaka: A Lightweight Cipher for RFID Systems," *Proc. of IEEE INFOCOM*, pp. 172–180, April 2014.

[39] S. Wu, S. Chen, D. Burr, and L. Zhang, "New Technologies for Full Privacy Protection in Data Collection and Analysis," *Proc. of 2014 Joint Statistical Meetings*, August 2014.

[40] Z. Mo, Y. Zhou, and S. Chen, "A Dynamic Proof of Retrievability (PoR) Scheme with O(logn) Complexity," *Proc. of IEEE ICC*, pp. 912–916, June 2012.

[41] ——, "An Efficient Dynamic Proof of Retrievability Scheme," *ZTE Communications, Special Issue on Big Data*, vol. 11, no. 2, pp. 24–29, June 2013.

[42] Z. Mo, Y. Qiao, and S. Chen, "Two-Party Fine-Grained Assured Deletion of Outsourced Data in Cloud Systems," *Proc. of IEEE ICDCS*, pp. 308–317, June 2014.

[43] Z. Mo, Q. Xiao, Y. Zhou, and S. Chen, "On Deletion of Outsourced Data in Cloud Computing," *Proc. of IEEE CIOUD*, pp. 344–351, June 2014.

[44] Z. Mo, Y. Zhou, S. Chen, and X. Chengzhong, "Enabling Non-repudiable Data Possession Verification in Cloud Storage Systems," *Proc. of IEEE CIOUD*, pp. 232–239, June 2014.

[45] Y. Zhou, S. Chen, Z. Mo, and Y. Yin, "Privacy Preserving Origin-Destination Flow Measurement in Vehicular Cyber-Physical Systems," *Proc. of IEEE International Conference on Cyber-Physical Systems, Networks, and Applications (CPSNA)*, pp. 32–37, August 2013.

[46] Y. Zhou, Q. Xiao, Z. Mo, S. Chen, and Y. Yin, "Privacy-Preserving Point-to-Point Transportation Traffic Measurement through Bit Array Masking in Intelligent Cyber-Physical Road Systems," *Proc. of IEEE International Conference on Cyber, Physical and Social Computing (CPSCom)*, pp. 826–833, August 2013.

[47] Y. Zhou, Z. Mo, Q. Xiao, S. Chen, and Y. Yin, "Privacy-Preserving Transportation Traffic Measurement in Intelligent Cyber-Physical Road Systems," *IEEE Trans. on Vehicular Technology*, 2015.

[48] G. Casella and R. L. Berger, "Statistical Inference," *2nd edition, Duxbury Press*, 2002.

[49] Y. Zhou, S. Chen, Z. Mo, and Q. Xiao, "Point-to-Point Traffic Volume Measurement through Variable-Length Bit Array Masking in Vehicular Cyber-Physical Systems," *Proc. of IEEE ICDCS*, pp. 51–60, June 2015.

[50] Y. Zhou, S. Chen, Y. Zhou, M. Chen, and Q. Xiao, "Privacy-Preserving Multi-Point Traffic Volume Measurement through Vehicle to Infrastructure Communications," *IEEE Trans. on Vehicular Technology*, 2015.

[51] SpoofMAC, "Spoof your MAC address," 2015. [Online]. Available: https://github.com/feross/SpoofMAC

[52] J. Petit, F. Schaub, M. Feiri, and F. Kargl, "Pseudonym Schemes in Vehicular Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 17, pp. 228–255, 2015.

[53] R. Lu, X. Lin, T. H. Luan, X. Liang, and X. Shen, "Pseudonym Changing at Social Spots: An Effective Strategy for Location Privacy in VANETs," *IEEE Trans. on Vehicular Technology*, vol. 61, pp. 86–96, 2011.

[54] Z. Ma, F. Kargl, and M. Weber, "Measuring long-term location privacy in vehicular communication systems," *Computer Communications*, pp. 1414–1427, 2010.

[55] A. J. Menezes, P. C. Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996.

[56] J. Benaloh and M. D. Mare, "One-way Accumulators: a Decentralized Alternative to Digital Signatures," *Advances in Cryptology-EUROCRYPT93*, pp. 274–285, 1993.

[57] C. Kaufman, R. Perlman, and M. Speciner, *Network Security, Private Communication in a Public World*, 2nd ed. Prentice Hall, 2002.

[58] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[59] A. Shamir, "On the generation of cryptographically strong pseudorandom sequences," *ACM Trans. on Computer System*, vol. 1, no. 1, pp. 38–44, 1983.

[60] W. K. Newey and D. McFadden, "Large Sample Estimation and Hypothesis Testing," *Dan. Handbook of Econometrics*, vol. 4, pp. 2111–2245, 1994.

[61] W. Bryc, "The normal distribution: characterizations with applications," *Springer-Verlag*, vol. 100, 1995.

[62] T. Li, S. Chen, and Y. Qiao, "Origin-Destination Flow Measurement in High-Speed Networks," *Proc. of IEEE INFOCOM, mini-conference*, pp. 2526–2530, March 2012.

[63] NYSDOT, "Traffic Volume Report," 2012. [Online]. Available: https://www.dot.ny.gov/divisions/engineering/technical-services/highway-data-services/traffic-data

[64] L. J. LeBlanc, E. K. Morlok, and W. P. Pierskalla, "An Efficient Approach to Solving the Road Network Equilibrium Traffic Assignment Problem," *Transportation Research*, vol. 9, pp. 309–318, February 1975.

[65] M. C. Ferris and J.-S. Pang, "Engineering and Economic Applications of Complementarity Problems," *SIAM Review*, vol. 39, no. 4, pp. 669–713, December 1997.

[66] H. Yang and H.-J. Huang, *Mathematical and Economic Theory of Road Pricing*. Elsevier, November 2005.

[67] S. H. Putman, *Integrated Urban Models Volume 2: New Research and Applications of Optimization and Dynamics*. Routledge Revival, April 2014, vol. 2.

BIOGRAPHICAL SKETCH

Yian Zhou received her Ph.D. degree in computer engineering from the University of Florida in the fall of 2015. She received her B.S. degree in computer science and her B.S. degree in economics from the Peking University of China in 2010.

Her research interests include traffic flow measurement, cyber-physical systems, big network data, security and privacy, and cloud computing.