Data Aggregation and Dissemination in Vehicular Ad-Hoc Networks

Khaled Ibrahim
Old Dominion University

DATA AGGREGATION AND DISSEMINATION IN VEHICULAR AD-HOC NETWORKS

by

Khaled Ibrahim
B.S. June 1999, Alexandria University, Egypt
M.S. January 2004, Alexandria University, Egypt

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

COMPUTER SCIENCE

OLD DOMINION UNIVERSITY
May 2011

Approved by:

_____

Michele C. Weigle (Director)

_____

Kurt Maly

_____

Hussien Abdel-Wahab

_____

Stephan Olariu

_____

Tamer Nadeem

ABSTRACT

DATA AGGREGATION AND DISSEMINATION IN VEHICULAR AD-HOC NETWORKS

Khaled Ibrahim
Old Dominion University, 2011
Director: Dr. Michele C. Weigle

Vehicular Ad-Hoc Networks (VANETs) are a fast growing technology that many governments and automobile manufacturers are investing in to provide not only safer and more secure roads, but also informational and entertainment-based applications for drivers. The applications developed for VANETs can be classified into multiple categories (safety, informational, entertainment). Most VANET applications, regardless of their category, depend on having certain vehicular data (vehicular speed, X position and Y position) available. Although these applications appear to use the same vehicular data, the characteristics of this data (*i.e.*, amount, accuracy, and update rate) will vary based on the application category. For example, safety applications need an accurate version of the vehicular data with high frequency, but over short distances. Informational applications relax the data frequency constraint as they need the vehicular data to be reasonably accurate with less frequency, but over longer distances. If each of these applications shares the vehicular data with *only* its peers using its own mechanism, this behavior will not only introduce redundant functionalities (sending, receiving, processing, etc.) for handling the same data, but also wastefully consume the bandwidth by broadcasting the same data multiple times. Despite the differences in the data characteristics needed by each application, this data can be still shared.

Vehicular networks introduce the potential for many co-existing applications. If we do not address the problem of data redundancy early, it may hinder the deployment and usefulness of many of these applications. Therefore, we developed a framework, cluster-based accurate syntactic compression of aggregated data in VANETs (CASCADE), for efficiently aggregating and disseminating commonly-used vehicular data. CASCADE is architected as a layer that provides applications with a customized version of the vehicular data, based on parameters that each application registers with

CASCADE. Additionally, the framework performs the common data handling functionalities (sending, receiving, aggregating, etc.) needed by the applications.

This dissertation makes the following contributions:

- a lossless data compression technique based on differential coding that is tailored for the characteristics of vehicular data

- a syntactic data aggregation mechanism that can represent the vehicular data in a 1.5 km area in one IEEE 802.11 frame

- a light-weight position verification technique that quickly detects false data with very low false positives

- a probabilistic data dissemination technique that alleviates the spatial broadcast storm problem and effectively uses the bandwidth to disseminate data to distant areas in a short amount of time in addition to having less redundancy and more coverage than other techniques.

- a mechanism for recovering from the communication discontinuity problem in short time based on the traffic density in the opposite direction

- an investigation of the possible data structures for representing the vehicular data in a searchable format

- a parametric mechanism for matching the vehicular data and providing a customized version of the data that satisfies certain characteristics based on the parameter value

CASCADE through its four major components, local view, extended view, data security and data dissemination, provides an efficient solution for the problem of scalability for VANET applications.

ACKNOWLEDGMENTS

Completing a PhD is truly a long journey that would not be possible to end successfully without the help and support of many people to whom I would like to express my gratitude. First and foremost, my advisor, Dr. Michele C. Weigle, to whom I owe so much. Dr. Weigle has been advising and guiding me since day one in the research phase of this journey. During thesis writing phase, she has been very passionate to review every single detail to get this thesis in its current shape. Also, I will never forget her support, encouragement and sincere advice when I decided to move to Microsoft.

Next, I would like to thank all my PhD committee members, Prof. Kurt Maly, Prof. Hussien Abdel-wahab, Prof. Stephan Olariu, and Dr. Tamer Nadeem for their valuable feedback and suggestions that helped in improving the dissertation.

I would to thank my parents and all family members who energized and supported me to start this journey. Lastly, and most importantly, my wife to whom my warmest appreciations and thanks go for her sacrifices, support, encouraging, patience and many other things throughout this long journey. Finally, my sons, Abdelrahman and Abdullah, whom their smiles were my energy in this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

.

Over the past few years, the technology needed to support vehicular ad-hoc networks (VANETs) has evolved quickly and has been adopted and supported by many automobile manufacturers (*e.g.* Honda, Toyota, etc.) and the US Department of Transportation (DoT). This industrial and governmental support has resulted in many ideas for applications targeting VANETs as their platform. These applications can be classified into various categories (*e.g.* safety, informational and entertainment). Regardless of their category, most of these applications require some common vehicular data, *i.e.* speed and position (X,Y), to be able to function properly. The data characteristics (accuracy, update frequency and volume) that each of these applications needs varies according to the application category. So, having each of these applications share the same set of data with its peers independently of the other applications will introduce a huge amount of redundant data into the media. This redundancy will wastefully consume the limited bandwidth dedicated to VANETs. Therefore, we propose a framework that can efficiently share common vehicular data among these applications and, at the same time, satisfy the data characteristics needed by each of them. The proposed framework consists of the following components: local view, extended view, data security and data dissemination.

Thesis Statement: The CASCADE framework can efficiently and securely provide each class of VANET application with a customized version of the vehicular data for the traffic ahead through enhanced compression, aggregation, and dissemination techniques. Efficiency will be evaluated by message reception rate, visibility, dissemination delay, and percentage of redundant messages sent.

I.1 VANET BASICS

A VANET is a special kind of network in which the vehicles represent the network nodes. The vehicles can communicate with each other in addition to communicating with available supporting infrastructure along the road. Vehicle-to-vehicle (V2V) communication helps in sharing the information within the VANET, while

---

This dissertation follows the style of *The IEEE Transactions*

vehicle-to-infrastructure (V2I) communication extends the VANET to include other designated networks (*e.g.* a centralized network or the Internet) and allows sharing of information with them.

VANET applications can be divided into three main categories: safety, informational, and entertainment. Safety applications, which are the main focus of the US DoT's efforts in vehicular networks, include collision warning [1, 2] and merge assistance [3]. Informational applications include notification of upcoming traffic conditions [4] and roadway hazards [5], as well as gathering and disseminating weather information [6]. The third type of applications, entertainment, includes Internet access [7], multimedia streaming [8], and P2P file sharing [9, 10].

To realize communication in VANETs, the Federal Communications Commission (FCC) dedicated 75 MHz of the frequency spectrum in the range 5.850 GHz to 5.925 GHz to be used for V2V and V2I communication. The 5.9 GHz spectrum was termed Dedicated Short Range Communication (DSRC) [11] and uses IEEE 802.11p [12]. VANET applications must share the allocated bandwidth, making it a scarce resource that should be managed very carefully. Inefficient data dissemination wastes a large amount of bandwidth that if saved would allow more vehicular applications to co-exist in addition to allowing the vehicular data to be disseminated further.

Many VANET applications require each vehicle to share its data (*e.g.* speed and location) with its neighbors through broadcasting a message containing such data. Sending these messages to farther distances will waste the bandwidth and may cause a broadcast storm problem based on the traffic density [13]. So, to share the data with vehicles at farther distances efficiently, many data aggregation techniques have been proposed.

Data aggregation has been proposed in VANETs to solve the bandwidth utilization problem. The basic idea is to gather information about many vehicles into a single frame. Data aggregation techniques can be classified as *syntactic* or *semantic* [14]. Syntactic aggregation uses a technique to compress or encode the data from multiple vehicles in order to fit the data into a single frame. This results in lower overhead than sending each message individually. In semantic aggregation, data from individual vehicles is summarized. For instance, instead of reporting the exact position of five vehicles, only the fact that five vehicles exist is reported. The trade-off is a much smaller message in exchange for a loss of precise data.

Efficient data dissemination in VANETs is an important problem that needs to be handled carefully. The importance of this problem has attracted many researchers to study it and, as a consequence, many data dissemination techniques have been proposed. The proposed techniques can be categorized according to the communication method used V2I, V2V, and hybrid of the two. In the infrastructure-based techniques, vehicles mainly communicate through infrastructure units that can be road side units (RSUs) [7, 15–17], embedded sensor belts in the road pavement [4], or cellular networks. V2I approaches depend upon a pervasive infrastructure that may not become a reality due to its high cost. V2V techniques for data dissemination can depend on data routing [18–20] or on broadcasting. In the majority of VANET applications (especially safety applications), the exchanged messages have no specific destination but have all the surrounding vehicles as the targeted destinations. This makes broadcast the most suitable method for dissemination. The third category of dissemination techniques combines both V2I and V2V based on their availability [21, 22].

The MAC layer architecture in IEEE 802.11 has two methods for accessing the medium, distributed coordination function (DCF) and point coordination function (PCF). PCF is only useful in the case of infrastructure network configuration, so it is not applicable for VANETs. So, the fundamental medium access method in VANETs is DCF. DCF relies on carrier sense multiple access with collision avoidance (CSMA/CA) for coordinating the medium access. The carrier sense in CSMA/CA can be either performed using physical mechanisms within the physical layer or virtually by the MAC layer using the RTS/CTS mechanism. RTS/CTS (Request to Send/ Clear to Send) is simply a medium reservation mechanism in which the node that has data to send (the source) will first send an RTS frame indicating the reservation time and identifying the destination node. Once the destination receives the RTS frame, it sends a CTS frame that when received by the source triggers sending the data frames. Meanwhile, all the other nodes within the transmission range should stay silent till the reservation time expires. In VANETs, it is not appropriate to use RTS/CTS as the carrier sense mechanism for CSMA/CA due to the following reasons:

- For most VANET applications, the message size that needs to be communicated is usually small, so using RTS/CTS to reserve the channel will reduce the system throughput because the RTS/CTS frames are considered overhead.

- Using RTS/CTS implies that there is a single intended recipient (who is supposed to send the CTS frame), while in most VANET applications all the vehicles within the transmission range are the intended recipients.

- In certain VANET applications, especially the safety related applications, the messages being communicated are very time critical so that using the RTS/CTS to reserve the channel before sending the message is not applicable.

This implies that in VANETs CSMA/CA should depend on the physical layer for sensing the media and reporting whether it is idle or not. As data broadcast is the common method for data sharing in VANETs and hence there will be no an ACK frames sent, DCF will operate as follows. Whenever a node has data to send, it first should determine that the medium is idle for DIFS[1] interval. If no interruption happens during the DIFS period, then it defers the data sending for random amount of time (random backoff). The logic for having the random backoff is to minimize the collision possibility in case of having more than one node waiting to send data. If no interruption happens during the random deferral time, then the node should proceed and send its data. In case of detecting an interruption during the DIFS or the random deferral time, the count down pauses until the channel is idle for a DIFS and then the countdown resumes.

To carry vehicle data beyond the original sender's transmission range, data re-broadcast will be needed. Flooding is the typical method for data broadcasting. Basic flooding is defined as whenever a node receives a frame, it will re-broadcast it. This may result in redundant re-broadcasts because the neighbors of the re-broadcasting node may have already received the original frame. Moreover, when multiple nodes in the same vicinity receive a frame, they will all re-broadcast it, causing severe contention on the channel. In addition to the previous two problems, collisions are highly probable due to the lack of RTS/CTS and collision detection mechanisms. So, blindly flooding will waste bandwidth by sending redundant frames that will probably collide. These three problems are collectively known as the *broadcast storm problem* [13]. So, there should be a more efficient method for handling re-broadcast of information to enhance bandwidth utilization and avoid the broadcast storm problem.

---

[1]DIFS is the DCF interframe space (DCF IFS), which is the time interval between frames in case of DCF

Security is also a very challenging problem in VANETs. Most of the presented techniques for data exchange in VANETs try to secure the exchanged data using Public Key Infrastructure (PKI) with digital signatures. The trusted authority in the PKI system is responsible for distributing key pairs on nodes so that each node should have pair of keys, public key and private key. The public key is known to everyone, while the private is *only* known to the owning node. Upon sending a message, to prevent any intermediate node from changing the original message content without being detected, the sender should calculate the message signature by calculating the message hash then encrypting the hash using its private key. The message signature is then attached to the original message in addition to the public key and the trusted authority certification for that key. Upon receiving the message, the receiving node should verify that the message content has not been tampered with, so it first verifies the public key authenticity using the trusted authority certification. Then it decrypts the message signature to get the original message hash. To verify that the message content has not been changed, it recalculates the message hash and compares it to the original message hash. If they match, this means the message content has not been changed. In almost all data dissemination techniques in VANETs, each vehicle is responsible for informing the others about its information, especially its location information. The receiving vehicles can verify that the information was actually sent by the sender and has not been tampered with, however this received data can be false. Disseminating false information is one of the common attacks in VANETs and is difficult to detect. Handling such an attack can be provided as part of the data dissemination technique to be used by all applications, or it can be delegated so that each application handles security in its own way.

I.2   MOTIVATION

VANET technology is growing quickly both in research and in the realization of research ideas into real applications. The energy behind this growth is coming from the huge support and fast adoption from government, represented by the US DoT, and industry, represented by many automobile manufacturers *e.g.* Honda, Toyota, etc. Both the US DoT and the automobile manufacturers depend on VANET technology to provide not only more safe and secure roads but also more comfortable and entertaining driving. This interest has resulted in many proposed vehicular applications that can be classified into three categories (safety, informational, and entertainment).

Most of the VANET applications, regardless of their category, depend on having certain vehicular data (vehicular speed, X position and Y position) available to perform the required functionalities. Although this common vehicular data (vehicular speed, X position and Y position) is required by most of these applications, each application, based on its category, needs certain characteristics to be satisfied in this data. These characteristics are the data accuracy, data refresh or update rate, and data amount. For example, for the safety applications to operate appropriately, they need to receive very frequently a highly accurate version of the vehicular data over short distances. While the informational applications can relax the frequency of the received data constraint, as they still need the received vehicular data to be reasonably accurate with less frequency, but over longer distances.

If each application shares this common vehicular data with its peers using its own mechanism to guarantee that the needed data characteristics will be satisfied, there is the potential for a tremendous amount of redundant data to be broadcast. Each individual application would broadcast the same vehicular data that the other applications broadcast. This vehicular data could be shared among all those applications despite the differences in its characteristics. Also, as each application will use a different mechanism for handling the data, there will be redundant implementations for the same functionalities. With the potential growth rate for the vehicular applications, if we do not find a solution that can efficiently share the vehicular data among all applications, it will hinder applications' performance and functionality and also limit the number of applications that can co-exist and share the medium simultaneously.

A solution for such problems is to insert a layer that implements the common data handling functionalities (data sending, receiving, aggregating, compressing, etc.) on top of which the vehicular applications could be implemented. Each application registers the data characteristics it needs with this layer. This layer should handle all the data-related functionalities on behalf of the registered applications and provide each of them with a customized version of the vehicular data satisfying those characteristics. Hence, the new layer will avoid wastefully consuming the bandwidth with redundant data and will simplify the implementation of the applications. Therefore, we designed and developed a framework that implements the functionalities in this new layer. This framework will be responsible for efficiently aggregating and disseminating the vehicular data most commonly used by vehicular applications. Having
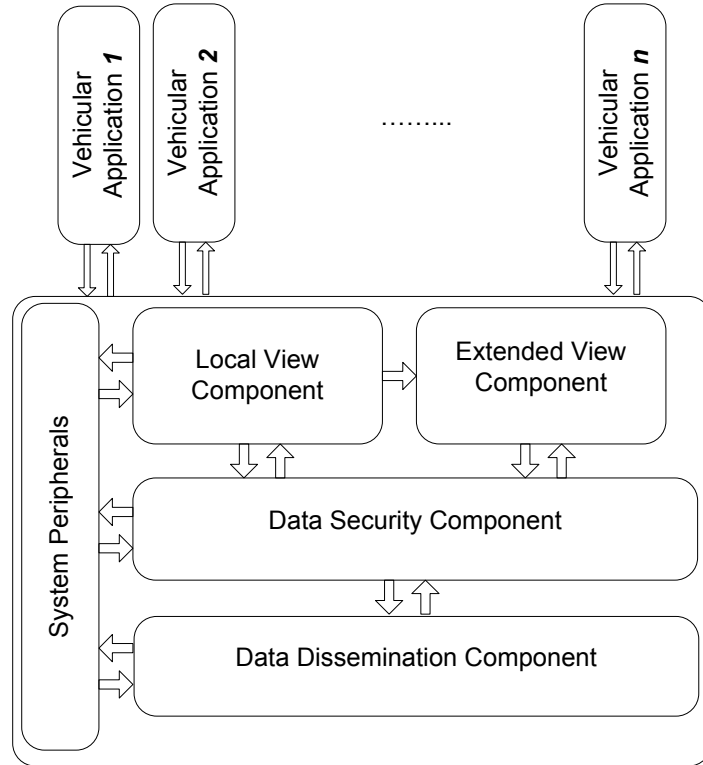
FIG. 1: Architecture of CASCADE Framework

such a framework will solve the scalability problem in VANET applications.

I.3 FRAMEWORK COMPONENTS

The framework that we are proposing to solve the problem explained in section I.2 consists of four main components: local view, extended view, data security and data dissemination. Figure 1 shows the framework architecture and the details of each component will be explained in the chapter corresponding to that component. We collectively call the framework components CASCADE (*Cluster-based Accurate Syntactic Compression of Aggregated Data in VANETs*). In CASCADE, the road ahead of each vehicle is divided into clusters, as in Figure 2, and the vehicles' data in each cluster is compressed using an adaption of differential coding. The compressed data from each cluster is aggregated so that it fits into a single MAC frame to form what is called the aggregated frame. Then each vehicle broadcasts its aggregated frame to help the other vehicles extend their views about the traffic conditions ahead.

Each vehicle in our proposed architecture is responsible for broadcasting its information (*i.e.*, location, speed, etc.) in what we call a primary frame, in addition to its aggregated frame. To send this information to distant vehicles, these frames should be re-broadcasted multiple times due to the limited transmission range, which is 300 m for DSRC [11]. Unless the data dissemination component handles the re-broadcasting of frames efficiently, it will lead to a broadcast storm. To avoid this, the data dissemination component in CASCADE uses a probabilistic broadcasting technique that adapts itself according to the vehicular density to utilize the bandwidth efficiently.

I.3.1 Assumptions

We assume that each vehicle in the system is equipped with a GPS receiver for obtaining location and time and a navigation system that can map GPS coordinates to a particular roadway and offer routes. The GPS precision is in the order of meters, which may result in inaccurate position estimation. So, we recommend using differential correction technology (DGPS), which reduces the error in estimating positions to the order of centimeters [23]. Each vehicle is also equipped with a communications device using Dedicated Short Range Communications (DSRC) [11]. Each vehicle is also pre-assigned a public/private key pair and the public key's certificate, used for authentication. To address privacy concerns, each vehicle may also use multiple pseudonyms to disguise its public keys [24, 25]. In our design of CASCADE, we assume a four-lane highway with 4 m wide lanes and ignore vehicles traveling in the opposite direction, although these vehicles may be used to disseminate reports during sparse traffic conditions.

I.3.2 CASCADE

The CASCADE framework consists of four main components: local view, extended view, data security and data dissemination.

The local view component is one of the basic building blocks in the CASCADE framework architecture. This component is responsible for building and maintaining an accurate view about the traffic ahead for a short distance, 1.5 km. This view is called *the local view*, because it is built and maintained locally inside each vehicle, Figure 2, using the primary frames received from the data dissemination component after being verified by the data security component.
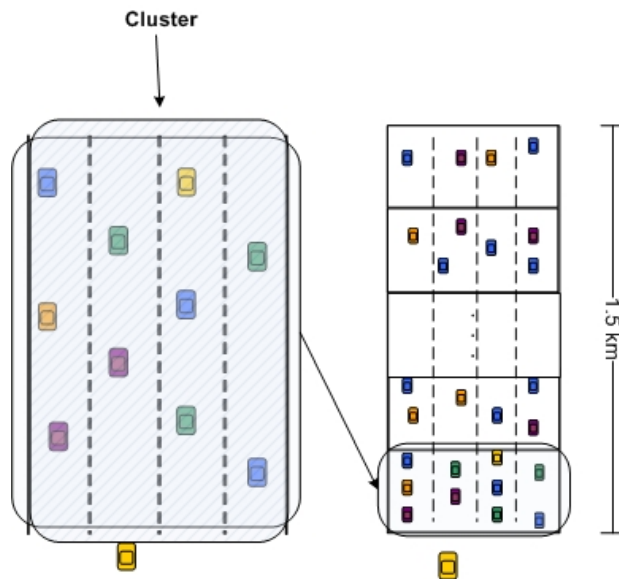
FIG. 2: Vehicle's Local View

The primary frames are the frames that each vehicle broadcasts frequently, and they encapsulate the vehicular data (vehicular speed, X position and Y position, etc.) for the broadcasting vehicle. These frames are received by the other vehicles and are used to build and maintain their local views. Also, this component is responsible for compressing and aggregating all the vehicular data in the local view to compose the aggregated frame. The aggregated frames are broadcasted by each vehicle, with less frequency than the primary frames and used by the other vehicles to build and maintain their extended view.

The extended view component is responsible for building and maintaining a view for the traffic ahead, but as opposed to the local view, it represents longer distances, up to 26 km. This component uses the aggregated frames broadcasted by the other vehicles to extend the vehicle's local view, as illustrated in Figure 3. The extended view component provides a customized version of the extended view to each application according to its pre-defined customization parameter. This component decompresses and reconstructs the original view from each aggregated frame it receives from the other vehicles. Then it compares the decompressed data to the local view data and, based on the comparison results and the customization parameter for each application, it decides whether to accept this aggregated frame or reject it. The matching technique in this component has two possible implementations, one using a graph
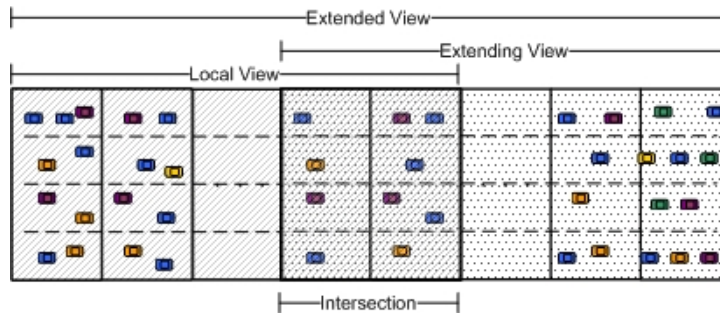
FIG. 3: Vehicle's Local View and Extended View

data structure for representing the vehicular data and the other using a KD-Tree it can switch between them based on the application settings.

The data security component is responsible for verifying the correctness of the received primary frames by other vehicles, in addition to adding the related security fields in both the primary frames and aggregated frames before broadcasting them. The security component in CASCADE is supported with a light-weight defense technique that handles the position verification problem. Our technique supplements the received signal strength (RSSI) mechanism with a laser rangefinder to reduce the inaccuracy incurred in the RSSI. The proposed defense technique consists of two main modules, the *detection module* and the *quarantine module*. The detection module is responsible for checking the consistency of received primary frames, while the quarantine module in a suspected vehicle is responsible for temporarily suspending the CASCADE application from sending any frames. The proposed technique can detect any malicious vehicle sending false position information very quickly with very few false positives, even under a low penetration rate (*i.e.*, low percentage of vehicles equipped with communications devices and CASCADE). We show that the incurred overhead due to using the position verification technique with CASCADE is negligible.

The data dissemination component is responsible for receiving the primary and aggregated frames broadcasted by the other vehicles and passing them to the security component to be verified before any further processing occurs. Also, it is responsible for disseminating both kinds of frames in the system efficiently, and in order to do that, it uses a probabilistic broadcasting technique that we developed called *probabilistic IVG (p-IVG)*. Additionally, the data dissemination component is responsible for continuously checking the traffic connectivity and handles any discontinuity using

the *On-Demand Vehicular Gap-Bridging (OD-V-GB)* technique that we developed.

CASCADE will be evaluated using a VANET-specific simulator that we developed, ASH (Application-aware SWANS with Highway mobility). ASH is an extension of the scalable network simulator SWANS [26] and provides the needed two-way communication between the vehicular mobility model and the networking model. To support highway scenarios, we included in ASH a customizable highway topology, allowing entrances, exits, and variable number of lanes. ASH includes implementations of the existing IDM car-following [27] and MOBIL lane-changing models [28], as well as the Inter-Vehicle Geocast [29] data dissemination protocol. To facilitate diverse simulations, ASH enhances the node model in SWANS to allow for the presence of non-participating vehicles and obstacles, along with participating vehicles and road-side infrastructure. These additions to the scalable SWANS simulator allow for realistic VANET simulations of important safety and traffic information applications.

I.4   CONTRIBUTIONS

The major contribution in this thesis is the design and development of the CASCADE framework that can provide each application category with a customized version of the traffic data ahead based on the application's settings without adding extra overhead to the medium. CASCADE relieves the applications from handling the vehicular data transmission and receiving so that they can be more focused on their business logic. Also, CASCADE solves the scalability problem for the vehicular applications as increasing the number of vehicular applications will not add much overhead to the network. CASCADE successfully achieved its functionalities through a well-architected set of components. Here, we list the framework components along with the contributions in each:

Local View Component Through this component the vehicle can have an accurate short view, $1.5\,\mathrm{km}$, of the traffic ahead. Within this component we investigated the possible data compression techniques that fit the vehicular data characteristics, and we found that differential coding compression provides a good compression ratio. Also, we determined the optimal cluster size that will maximize the local view length and still keep the maximum aggregated frame size to be less than $2312\,\mathrm{bytes}$, the IEEE 802.11 maximum frame size.

Extended View Component The contributions in this component include finding

suitable data structures, graph and KD-Tree, for representing the vehicular data and comparing them efficiently. In addition, this component can use the other vehicles' aggregated frames to build an extended view for the traffic ahead, up to 26 km, that can be customized based on the applications' needs for data accuracy, refresh rate, and view length.

Data Security Component In this component, we developed a light-weight position verification technique that can guarantee passing only verified primary frames to the local view component with high accuracy.

Data Dissemination Component During the work in this component, we were the first to discover the *spatial broadcast storm problem* that happens in dense traffic when using the IVG technique for data dissemination. We developed and designed an enhanced version of IVG, *probabilistic IVG (p-IVG)*, that alleviates the *spatial broadcast storm problem* and disseminates the vehicular data to reach distant areas efficiently even in dense traffic. *p*-IVG has shown good results with respect to many metrics (*e.g.* reception rate, visibility, etc.) when compared to both IVG and flooding when using it within the CASCADE context and even outside of it. Also, we developed and designed a new technique, *On-Demand Vehicular Gap-Bridging (OD-V-GB)*, to detect and alleviate the traffic discontinuity problem. The performance of OD-V-GB has been evaluated within the CASCADE context, and it can detect and recover from the traffic discontinuity problem quickly.

I.5   THESIS ORGANIZATION

In Chapter II we briefly present related work pertaining to data compression and aggregation, position verification, data dissemination and VANET simulators. Chapter III presents the details for the local view component including the data compression and aggregation techniques. The extended view component is presented in Chapter IV with all the details about the alternative data structures, graph and KD-Tree for representing the vehicular data. Also, we present in this chapter how to compare the vehicular data and customize the decision whether the compared data is matching or not based on a pre-defined application specific parameters. Chapter V describes the position verification technique utilized by CASCADE to tighten its security. Then in Chapter VI we present the data dissemination technique, *p-IVG*, that will be used by

CASCADE to disseminate its various data frames. This chapter also discusses the traffic discontinuity problem and our solution, OD-V-GB. As a proof of concept, in Chapter VIII we show how CASCADE can be used in a vehicular advertising system and the benefits that the system will gain from using CASCADE. Also we briefly explain how to build a vehicular merge assistant system based on CASCADE. In Chapter IX we summarize the thesis work and present the directions for future work. Finally, we present details of our developed simulator ASH in Appendix A.

CHAPTER II

RELATED WORK

In this chapter we present the approaches and solutions proposed by others to handle data dissemination, data compression and aggregation, and position verification problems in VANETs. Although there are many solutions that can handle these problems in MANETs, they cannot be directly applied to VANETs because VANETs have some very different characteristics from MANETs, namely higher mobility speeds and restricted network topologies and vehicle movement. Also, we present previous work on VANET simulators and the differences between those developed simulators and our simulator ASH.

II.1 DATA COMPRESSION AND AGGREGATION IN VANETS

Data aggregation has received much attention in the wireless sensor network community [30–32], but many of the approaches either assume a static network or require several rounds of communication between nodes to provide security. Both of these requirements are impractical for VANETs.

There has been recent work on data aggregation techniques specifically designed for VANETs. Picconi *et al.* [14] classified aggregation techniques as either *syntactic* or *semantic*. Syntactic aggregation uses a technique to compress or encode the data from multiple vehicles in order to fit the data into a single frame. This results in lower overhead than sending each message individually. In semantic aggregation, the data from individual vehicles is summarized. For instance, instead of reporting the exact position of five vehicles, only the fact that five vehicles exist is reported. The trade-off is a much smaller message in exchange for a loss of precise data.

Nadeem *et al.* [33] present the *TrafficView* system, which uses semantic aggregation. The authors present two techniques for aggregation: *ratio-based* and *cost-based.* In the ratio-based technique, the roadway in front of a vehicle is divided into regions. Data is aggregated based on ratios that have been pre-assigned to each region. Regions farther away from a vehicle are assigned larger aggregation ratios, because precise detail may not be needed over a long range. The resulting view of traffic conditions is, thus, customized for each particular vehicle. For this reason, the produced view may not be useful for other vehicles unless they use the same aggregation

ratios. In the cost-based aggregation technique, data is aggregated based on a cost function that depends on the position of the aggregating vehicle. For this reason, the produced view of the traffic is not useful to any other vehicle unless it is close to the aggregating vehicle.

Lochert *et al.* [34] present a probabilistic technique for aggregating the disseminated data in VANET applications. The proposed technique does not aggregate the actual values but uses a modified Flajolet-Martin sketch as a probabilistic approximation for the values. This technique can be applied to aggregate the data in any non-accuracy-sensitive application (*e.g.*, estimating the number of available parking spaces), but it cannot be used in our CASCADE framework, which requires the actual vehicle information to be disseminated and re-aggregated to reach distant vehicles.

Yu *et al.* [35] present an aggregation technique called Catch-Up that aggregates similar reports generated by the vehicles whenever an event occurs *e.g.*, a change in vehicle's density. The technique is based on inserting a delay before forwarding any report in the hopes of receiving similar reports from surrounding vehicles so that these reports can be aggregated into a single report. Since Catch-Up inserts a delay before forwarding messages, it would not be suitable for safety applications, such as collision warning.

Lochert *et al.* [22] describe a hierarchical aggregation technique for vehicle travel times. In this technique each vehicle broadcasts its travel time between two landmarks along its trip. Then these travel times are aggregated hierarchically and broadcasted to provide distant vehicles with an estimate of the travel times along the road segments so that they can avoid congested roads (the roads with larger travel time estimate). In case of a slow driver traveling along the road, the vehicle will report a long travel time between any two landmarks, which will be translated by the other vehicles as congestion between those landmarks even though there may be no congestion on the roadway. As with Catch-Up, this work does not disseminate or aggregate information suitable for safety applications, but is only concerned with reporting traffic conditions.

Saleet *et al.* [36] present a location query protocol that aggregates data in VANETs. The protocol divides the road into segments, and the closest node to the segment center plays the server role. Each vehicle periodically broadcasts its

information, and the server node is responsible for storing this information, aggregating it, and then broadcasting it. The aggregation technique is different than ours in that it does not include any compression mechanisms. Moreover, the main target is to provide a location query facility which is orthogonal to our target applications, which are collision avoidance and congestion notification.

Robinson *et al.* [37] present the concept of using message dispatcher to coordinate all the data exchange requirements of the safety applications running on a vehicle. The message dispatcher accomplishes this by serving as an interface between the safety applications and the communication stack. Safety applications will send data elements to be broadcast to the message dispatcher, which will then summarize these data elements across safety applications and create a single packet comprising the minimum set of the data elements to be transmitted. In CASCADE framework, we extend the concept of message dispatcher to be used not only within the same category of applications, but also across different categories.

II.2   POSITION VERIFICATION IN VANETS

Many studies have been performed for securing vehicular networks using either Public Key Infrastructure (PKI) [24, 25, 38–40] or digital signatures [25, 41–43]. But none of these studies address how to verify the correctness of the data inside the messages.

MANETs suffers from the position verification problem as well as VANETs [44, 45]. Most of the solutions that have been suggested to handle such problem in MANETs are not applicable for VANETs, because the techniques used in these solutions are either depend on limited speed nature for the nodes in MANETs or the freedom to exist in any location in the field. While, it is exactly the opposite in case of VANETs as the vehicles can move with a variety of speed ranges from low to high speeds and the vehicles movement is limited by the road boundary. Due to these differences in characteristics, the solutions proposed to alleviate the position verification problem in MANETs are not applicable for VANETs.

Although the position verification problem has not received the attention in VANETs equal to its importance, there has been some research performed in this area. Leinmüller *et al.* [46, 47] showed that disseminating false position information in VANETs has a much more severe impact on the system in highway scenarios than in city scenarios. Thus, we focus on position information in highway scenarios.

Golle *et al.* [48] used heuristics such as drastic changes in speed or location to

determine if a vehicle is sending false information. The technique tries to correct the information rather than quarantining the malicious node. In addition, the technique assumes that nodes can be distinguished from other nodes, using RSSI, camera, or other devices.

Yan *et al.* [49] proposed a novel technique for verifying the claimed location using an omni-directional radar system, but such a device is not yet available. Their technique incurs more communications overhead than our proposed technique, and some attacks cannot be handled satisfactorily (*e.g.*, a lying vehicle claims to be in a location where another vehicle already exists, so both are blocked).

There has been much recent work on position verification in mobile ad-hoc networks, as well as vehicular networks. Sastry *et al.* [44] proposed a technique to determine if a mobile communicating node exists in the region near where it claims to be. The limitation of this technique is that it cannot verify that the node is in the exact claimed location, which reduces its accuracy. Suen *et al.* [45] used radio signal properties (*e.g.* direction, strength) to determine the transmitter location. Using the radio signal properties alone bounds the technique's accuracy to the RSSI accuracy, which is in meters. In the previous two techniques, accuracy was an issue. Having such low accuracy will increase the false positive rate while our proposed technique inherits the laser rangefinder accuracy, which is in centimeters.

Xiao *et al.* [50] also use radio signal strength to verify nodes' locations. To alleviate the inaccuracy in this technique, they enhance the location estimation by using statistical algorithms and road-side infrastructure. The use of road-side infrastructure increases the deployment cost drastically, while our proposed technique needs no infrastructure and has comparable accuracy.

Recently, Tiffany *et al.* [51] have proposed a new security model to detect messages reporting false events, regardless of the false information source whether it is sent intentionally or due to malfunctioning devices. The model detects and filters out the malicious messages whenever the certainty level of being false exceeds certain threshold. The certainty level is being built by examining the information using 6 information sources (Local Sensors, Sender Reputation, etc.). The driver is notified of verified events only when they are relevant enough to him. Our approach for position verification and Tiffany's approach both align in considering the other vehicles' decisions about the correctness of certain events in building the certainty level about the same event. They also both consider the event relevance while making

the decision about it. Our approach differs from Tiffany's approach in being more specific to verifying the received vehicular positions and speeds, while Tiffany's is more generic. The downside in Tiffany's approach is that it has some dependency on roadside infrastructure while ours does not. Also in Tiffany's, malicious vehicles are allowed to continue in sending messages, while we can permanently block the sending unit in the malicious vehicles so that they cannot send false messages anymore until they are unblocked by central authorities.

Our approach for handling the position verification problem provides a lightweight mechanism for detecting and quarantining the malicious vehicles with high accuracy, in addition to verifying the received vehicular locations before performing any further processing on them.

II.3   DATA DISSEMINATION IN VANETS

As we have mentioned, broadcast is the most appropriate data dissemination technique in VANETs especially for safety-related applications. One of the most common problems in broadcast is the broadcast storm problem [13]. Various solutions have been proposed to handle the broadcast storm problem in mobile ad-hoc networks (MANETs) [13, 52, 53], but most of them are not applicable in case of VANETs. The design of the MANET techniques depends on the node limited speed, but in VANETs the nodes (the vehicles) can move with much higher speeds.

The approaches that have been proposed to alleviate the broadcast storm problem in dense VANETs can be classified as probability-based, timer-based and priority-based approaches. The main idea behind the probability-based techniques is that whenever a vehicle receives a frame, it will re-broadcast it after holding it for a certain waiting time, according to a probability p that depends on how far this vehicle is from the sending vehicle. Distant vehicles will have higher probability p to re-broadcast the received frames than nearby vehicles. This technique is called weighted p-persistence broadcasting and has been proposed by Wisitpongphan *et al.* [54].

There are two versions of the timer-based techniques: slotted time and continuous time. Wisitpongphan *et al.* [54] proposed the slotted 1-persistence broadcasting technique, dividing the waiting time into slots. When each vehicle receives a frame, it is assigned a time slot during which it should re-broadcast the frame with probability 1 if no one else had re-broadcasted it. As the distance increases, the receiving vehicle

is assigned a shorter re-broadcast time slot. Also, the same authors proposed a slotted p-persistence broadcasting technique, in which during the assigned time slot the vehicle will re-broadcast the frame with a pre-determined probability p.

Briesemeister *et al.* [55] proposed a continuous time version of the timer-based technique, called role-based multicast. In this technique, whenever a vehicle receives a frame, it waits for a certain amount of time before re-broadcasting it. The longer the distance from the sender, the shorter the waiting time. This technique was only concerned with the sparsely connected networks to maximize the message reachability and does not handle the broadcast storm problem. Bachir *et al.* [29] proposed Inter-Vehicle Geocast (IVG), based on the same idea presented by Briesemeister *et al.* [55], but it handles the dense network situation.

In general, timer-based techniques have shown superiority over the probability-based techniques in mitigating the broadcast storm problem. In dense VANETs, the timer-based techniques re-broadcast the frame sooner (early re-broadcast) because the re-broadcast waiting time lessens with increasing distance from the sending vehicle, while the waiting time is constant in the probability-based techniques. Also, having an early re-broadcast will reduce the channel contention and redundant re-broadcasts because whoever hears the re-broadcast will cease its own re-broadcast. Within the timer-based techniques, the time continuous version known as IVG, is better than the slotted time techniques, because the slotted versions restrict the re-broadcast to be initiated only at certain times, which increases channel contention. Although IVG mitigates the broadcast storm problem, it suffers from the spatial broadcast storm problem, as it does not utilize the network topology information in the re-broadcast decision.

In both of the timer-based and probability-based techniques the farthest vehicle(s) within transmission range of the original sender will be selected to be the re-broadcaster(s). Based on that, all the vehicles at the boundary of transmission range will re-broadcast the frame at the same time. This will cause a broadcast storm locally at the boundary. Although the generated storm is local, it will affect the overall system performance. We term this the *spatial broadcast storm problem.*

Torrent-Moreno *et al.* [56] proposed a priority-based broadcast scheme in which nodes that have a time-critical message to send will be assigned a higher priority to access the channel. In general, the priority-based techniques categorizes the network nodes into multiple classes with different priorities and schedules frame transmission

accordingly. This technique reduces the contention on the channel access by allowing the higher priority nodes to access the channel before the other nodes, but it does not solve the broadcast storm problem.

Tonguz *et al.* [57] provided a comprehensive framework (DV-CAST) to handle broadcasting in VANETs considering three possible traffic densities (dense, regular, and sparse). In dense traffic, they suggest using one of the timer-based techniques [54], while in sparse traffic they suggest using role-based multicast [55]. They did not suggest a specific technique to be used in case of regular density traffic since they define regular density as a mix of some vehicles sensing dense traffic and some vehicles sensing sparse traffic. So, each vehicle will use the technique appropriate to what it has sensed. Even though DV-CAST appears to be a complete solution, it is still vulnerable to the spatial broadcast storm problem. Our contribution of p-IVG was designed to specifically address the spatial broadcast storm problem.

II.4 VANET SIMULATORS

Without a doubt, the existence of a standard VANET simulator that the entire research community trusts would enhance the research quality and shorten the development cycle of any VANET application. Towards this goal, much effort has been exerted to develop a VANET-specific simulator by integrating a network simulator with a mobility generator, because the simulation of VANET applications not only requires simulating the wireless communication between the vehicles, but also requires simulating the mobility of the vehicles.

Vehicular traffic models are typically classified into three categories based on traffic granularity: macroscopic, mesoscopic, and microscopic. Macroscopic models deal with traffic as flows, while mesoscopic models are concerned with the movement of whole platoons of vehicles. Microscopic models handle the movement of each vehicle in the traffic flow, thus they are the most suitable for VANET applications. Many microscopic models have been developed. The most widespread ones are the SK model [58], the Cellular Automaton (CA) model [59], and the IDM/MOBIL model [60] [61]. Fiore *et al.* [62] evaluated the realism of several mobility models and recommended that only realistic car-following models, such as IDM, be used in VANET simulations. There have been multiple efforts for developing mobility simulators. SUMO [63] is considered one of the pioneers mobility simulators that can generate vehicle mobility traces. CanuMobiSim [64] is another mobility simulator

that can also generate vehicle mobility traces.

There are variety of network simulators that are being used in the network research community. Some researchers prefer open source simulators like GloMoSim [65], OMNet++ [66], and ns-2 [67], while others prefer the commercial simulators, such as QualNet [68] and OPNET [69], to get better support and customization. Although ns-2 is an open source simulator, it is the most widely-used network simulator in the research community [70]. But, ns-2 suffers from problems when simulating large numbers of nodes. SWANS [26] was developed to be a scalable alternative to ns-2 for simulating wireless networks. Based on comparisons [71] between SWANS, GloMoSim and ns-2, SWANS was found to be the most scalable, the most efficient in memory usage, and fastest in runtime. In addition, Kargl *et al.* [71] validated the network model in SWANS against ns-2. They showed that along with better performance, SWANS delivered similar results as ns-2, at least for the network components that were implemented in both.

Unfortunately, in most cases these two components of VANET simulation (wireless network/communication simulation and vehicular mobility simulation) have been decoupled. Both vehicular mobility and wireless communication have large communities concerned with their modeling and simulation, so high quality simulators exist in each of these areas. The problem is in merging the two types of simulators. An ideal VANET simulator would consist of two sub-simulators, a network simulator to simulate the wireless communication between the vehicles and a traffic simulator to simulate the vehicles' mobility. VANET applications that run at the top level of the network simulator can then be categorized according to how these two sub-simulators need to communicate.

Entertainment-related applications, including Internet connectivity [7], multimedia applications, and peer-to-peer applications [9, 10], require only *one-way communication* between the two sub-simulators, from the traffic simulator to the network simulator. The network simulator uses the provided information (position information, speed, acceleration, direction, *etc.*) from the traffic simulator for data routing. Since this kind of application has no effect on driving decisions (*i.e.* vehicle mobility), there is no need to generate the mobility information dynamically. For simplicity, a pre-generated trace file of the vehicles' mobility is often used.

Safety-related and traffic information applications require *two-way communication* between the traffic simulator and network simulator. In these types of applications, the traffic simulator feeds the network simulator with position information, speed, acceleration, direction, *etc.* The VANET application that runs at the top level of the network simulator incorporates this information with surrounding vehicles' information in order to notify the driver of upcoming congestion or a possible collision. Based on this notification, driving decisions (*i.e.* vehicle mobility) may be affected. For example, the driver may choose to change lane or slow down. These mobility decisions need to propagate back to the traffic simulator to be reflected in the vehicle mobility information. In this way, the mobility model becomes *application-aware.*

Simulating one-way communication is straightforward and is already supported by the combination of various traffic mobility and network simulators. On the other hand, the difficulty in simulating two-way communication resides in how to couple independent traffic simulators with network simulators. Recently, many VANET-specific simulators have been developed, but most allow only one-way communication.

There have been several efforts to combine network and mobility simulators to achieve one-way communication. Karnadi *et al.* [72] developed a tool that interfaces with the mobility generator SUMO [63] to generate vehicle mobility traces that can be imported into the network simulators ns-2 [67] or QualNet [68]. CanuMobiSim [64] is a mobility simulator that allows for the export of mobility traces for use in various network simulators. The developers of VanetMobiSim [73] extended CanuMobiSim by incorporating IDM as a traffic model. The authors of STRAW [74] extended SWANS by implementing a street mobility model. Saha and Johnson [75] implemented a simple mobility model directly in ns-2.

Recently, we have seen multiple simulators integrating the network and mobility component by implement both of them in one simulator. GrooveSim [76] can be considered as an example for such simulators, but the network model has not been validated against any other well-known network simulator. The same is true for NCTUns [77]. Further, in both cases, the mobility components cannot be separated from the network components, which makes them difficult to extend further. TraNS [78] couples ns-2 with SUMO and provides for both one-way and two-way communication between them. But, as TraNS uses ns-2 for its network simulator, it inherits the scalability problems that ns-2 suffers from [71]. Plus, using two separate

simulators (ns-2 and SUMO) doubles the overhead of managing the environment settings for each. Arbabi *et al.* [79] present the first implementation of a well-known vehicle mobility model to ns-3. Yan *et al.* [80] provide a comprehensive overview of the state of vehicular network simulators.

II.5 SUMMARY

In this chapter, we briefly presented related work that has been done by other researchers. First, we presented the previous efforts in compressing and aggregating the vehicular data and how our technique is different from them, namely being the only lossless yet accurate and efficient technique among them. Verifying the data correctness that the system is processing is an important aspect in maintaining the system accuracy, so we have presented the related work on VANET security and showed how our proposed technique for verifying the vehicular data is different. Because bandwidth is a scarce resource in both MANETs and VANETs, we have explained the broadcast storm problem that may happen when using the bandwidth inefficiently, the techniques that are trying to alleviate such problem in MANETs and why these techniques are not applicable in VANETs. Also, we briefly explained the related work for data dissemination in VANETs and how our proposed technique (p-IVG) handles the same problem differently and more efficiently. Finally, we presented the work that has been done recently in developing both wireless network and vehicular mobility simulators. We also covered the efforts exerted in integrating these two type of simulators to create a VANET simulator and described how our simulator (ASH) is the first VANET simulator that provides two-way communication between mobility and network simulator components.

CHAPTER III

LOCAL VIEW COMPONENT

In this chapter, we present the details for the local view component which is the core of the CASCADE framework and how this component interacts with the other system components. The main responsibilities of this component are to

- build and maintain an accurate view about the traffic ahead for short distance (1.5 km), and

- compress and aggregate the vehicular data in its view to build the aggregated frames

To fulfill these responsibilities, the local view component should maintain a view that has a certain dimension (1.5 km length and 4 lanes width) with respect to its current location that is being updated using GPS. The other system components, namely the data dissemination component and the data security component, contribute to the success of the local view component in building and maintaining its view accuracy. The data dissemination component receives the primary frames broadcasted by the other vehicles and passes them to the data security component that verifies them and passes the legitimate ones to the local view component. At that time, the local view component extracts the vehicular data from these frames to build and update the local view. To keep the local view current, the component applies an aging technique to purge the obsolete vehicular data from the view. In addition to the local view component's responsibility for building and maintaining the local view, it frequently compresses and aggregates the vehicular data in its view to build the aggregated frames. The local view component passes the aggregated frames to the data security component, which calculates and adds the security fields. Then the data security component passes these frames to the data dissemination component to be broadcasted. To avoid having the aggregated frames fragmented, the maximum size for the aggregated frame should not exceed 2312 bytes, the IEEE 802.11 maximum frame size. To satisfy the aggregated frame maximum size constraint and at the same time achieve the maximum local view length with the minimum overhead on the network medium, we analyzed the relationship between the various system variables and found the optimal cluster size to be 16 m wide and 126 m long.

This chapter is organized as follows, we give an overview about the local view component then we explain the details for how it functions. First, we explain the details for how the vehicular data is represented in the primary records and how they are disseminated, rebroadcasted and purged after aging. Then, we explain the details for how the vehicular data in the local view is compressed and aggregated to form the aggregated records, in addition to how those records are disseminated. Finally, we study the problem of finding the optimal cluster size.

III.1   LOCAL VIEW COMPONENT OVERVIEW

Before describing the details of the system, we present a brief high-level overview. Each vehicle periodically broadcasts its vehicular data (including location, speed, acceleration, and heading), which we call a *primary record*. Received primary records are stored in a local database in each vehicle. Those primary records representing vehicles ahead of the current vehicle comprise the *local view*. The local view, as shown in Figure 4, is divided into *clusters*[1]. Each cluster has a width of 16 m (4 lanes) and a length of 126 m. Selecting the cluster dimensions 16 m x 126 m is based on optimal cluster size analysis, balancing the trade-off between local view length and expected frame size. More detail will be presented in Section III.4. There are 12 rows of clusters in a local view, resulting in a visibility of 1.5 km (exactly 1512 m).

As the local view is longer than the typical DSRC transmission range (about 300 m), primary records may be re-broadcast (to a maximum of 1.5 km behind the original sender).

Each vehicle periodically compresses and aggregates the primary records in its local view into an *aggregated record*. This aggregated record is then broadcast to neighboring vehicles. Received aggregated records may be used to augment the local view by providing information about vehicles beyond the local view, resulting in an *extended view*. Figure 5[2] shows an example of the local view and extended view.

Although our examples feature a straight, rectangular-shaped road, CASCADE is not limited to such geometries. Figure 6 shows a local view mapped onto a curved roadway. As the vehicle enters the curve, more of the vehicles inside and past the

---

[1]The term *cluster* is used here in local sense only. Each vehicle will assign vehicles it knows about into the appropriate cluster based on the vehicles' distances from itself. Thus, there is no need for cluster management or node agreement on which vehicles are in which clusters.

[2]This figure is a copy of Figure 3.
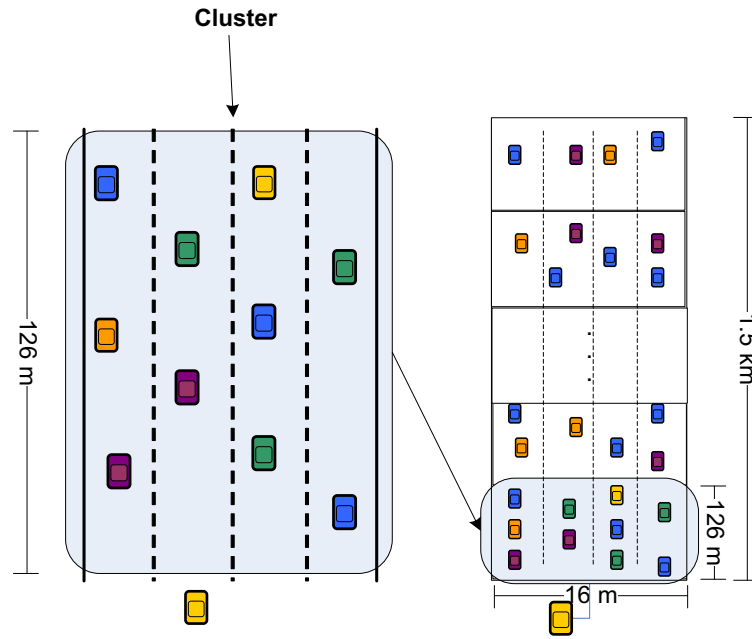
FIG. 4: Vehicle's Local View, Divided into 16 m x 126 m Clusters
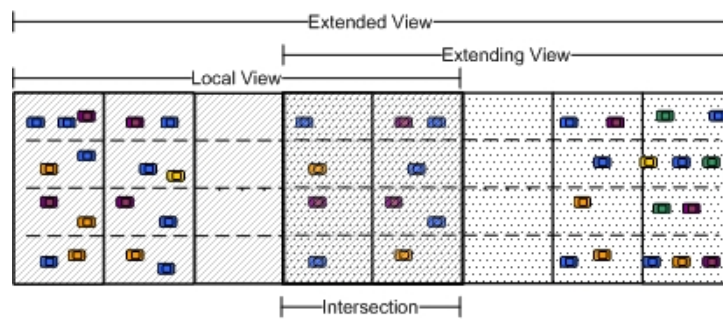
FIG. 5: Comparing Vehicle's Local View and Extended View
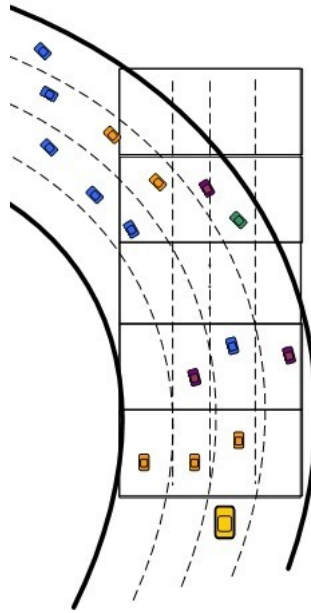
FIG. 6: Vehicle's Local View Mapped onto a Curved Roadway

curve will be added to its local view. CASCADE has a limited capability in handling the curved roads that will be handled in the future work.

III.2 PRIMARY RECORDS

A vehicle's local view is built entirely of received primary records. The primary record contains the basic information for a single vehicle. Each record can be represented in 29 bytes:

- *timestamp* (8 bytes) - time the record was generated

- *location* (16 bytes) - latitude and longitude

- *speed* (1 byte) - in meters/second

- *acceleration* (1 byte) - in meters/second^2

- *heading* (1 byte) - in degrees from North (0-360)

- *altitude* (2 bytes) - in meters above sea level

III.2.1 Initial Dissemination

A vehicle broadcasts a *primary frame* containing its primary record at a random interval between 300-400 ms, which is consistent with message frequency recommendations for informational applications [81]. The primary frame, totaling 1033 bits[3], consists of:

- *type* (1 bit) - primary or aggregated frame

- *sender's location* (16 bytes) - latitude/longitude

- *primary record* (29 bytes)

- *digital signature* (28 bytes)

- *certificate* (56 bytes)

The sender's location in the primary frame is updated each time the primary frame is re-broadcast by another vehicle. The primary record is signed by the original vehicle using ECDSA [82]. The certificate included in the frame contains the original vehicle's public key, signed by the certificate authority. Since the primary record is signed by the original sender, it cannot be tampered with by a re-broadcasting node without detection. In addition, replay attacks are nullified by the presence of the timestamp inside the signed primary record.

A receiving vehicle will record the primary record and use the vehicle's public key as an identifier. Typically, only primary records from vehicles within the receiving vehicle's local view (*i.e.*, vehicles in front of the receiving vehicle) will be stored. But for some applications, such as merging assistance, awareness of vehicles behind or beside the receiving vehicle is important. In these cases, the vehicle would store the primary records of nearby (one cluster' worth, or within 126 m) following vehicles to be used in the application. Again, since these records are from following vehicles, they are not considered part of the local view.

III.2.2 Rebroadcast

In order for primary records to reach vehicles farther than 300 m, the records must be re-broadcast. To limit the number of re-broadcast messages use to propagate the

---

[3]If the underlying link-layer requires a frame size of full bytes, then the primary frame would be padded to 130 bytes.

frames, we use p-IVG, an adaptation of the IVG algorithm, which will be described in Chapter VI.

In addition to using p-IVG to limit the number of re-broadcasts, primary frames have a time-to-live (TTL) value to ensure that only fresh information is disseminated. The TTL of all primary frames in CASCADE is 1 second. If a node receives a frame and the difference between the original sending time and the current time is greater than the primary frame TTL, the frame will be dropped.

III.2.3   Aging

The goal of CASCADE is to present highly accurate information about upcoming traffic conditions. So, it is important that old information is purged from the system. As the local view is concerned only with vehicles in front of the current vehicle, primary records are removed from the local view (but not necessarily from a vehicle's database) once the vehicle has physically passed the vehicle described by the record. Additionally, primary records may also be removed from the local view when no updates have been received in 1 second. With vehicles sending new primary frames 3 times a second, receiving no message about a vehicle for 1 second means that 3 messages in a row were not received, which would indicate that the vehicle corresponding to the old record has likely left the area.

III.3   AGGREGATED RECORDS

Each vehicle builds its local view based on primary records received from other vehicles. In order to extend the view farther, vehicles exchange aggregated records. Here, we describe how primary records are grouped into clusters, how clusters are aggregated, and how the aggregated records are disseminated and used to build the extended view.

III.3.1   Compression

As primary records are received, the vehicles described in those records are grouped into their corresponding clusters, based on their distance from the receiving vehicle. When clustering is done, a vehicle's heading and altitude are taken into account to ensure that vehicles are assigned to the proper cluster.
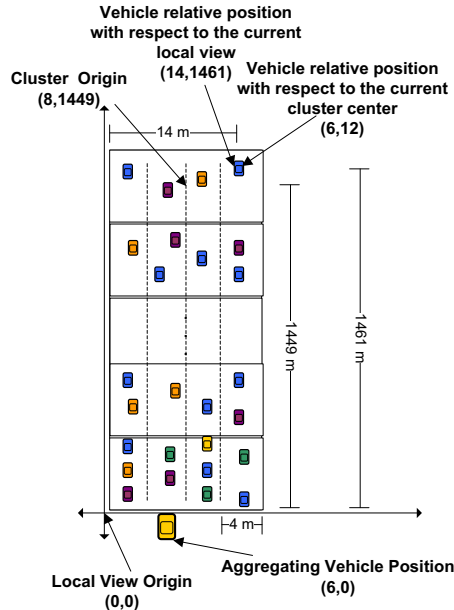
FIG. 7: Converting GPS Coordinates to {X,Y} Coordinates

The *compact record* is used to represent a single vehicle within a cluster. The compression is achieved by using a variation on differential coding that is more efficient in compressing the vehicular data. CASCADE represents only the differences between the vehicle data and overall cluster data. Before the compact record is formed, the median speed of all vehicles in a cluster is calculated, the position of the center of the cluster is calculated, and the position of each vehicle is translated into $\{X, Y\}$ coordinates (in integer meters) with the local view origin as the origin, as shown in Figure 7. For this, we assume that the digital map in the vehicle provides the GPS position of the leftmost lane of the roadway.

Each compact record, totaling 16 bits, contains the following fields:

- ΔX (5 bits) - difference between the vehicle's X coordinate and the X coordinate for the center of its cluster

- ΔY (7 bits) - the difference between the vehicle's Y coordinate and the Y coordinate for the center of its cluster

- ΔS (5 bits) - the difference between vehicle's speed and the median speed of the vehicles in the cluster

- *Speed Indicator (SI) Flag* (2 bits) - indicates if the vehicle's speed is within the

acceptable range for the cluster

Since the base value is the center of the cluster (as in Figure 7), ΔX can have a negative value. With sign-magnitude representation, this means that five bits must be used for ΔX, one for the sign and four for the magnitude (with a maximum width difference of $8\,\text{m}$).

Since the cluster length is $126\,\text{m}$, ΔY can be represented with only 7 bits, using 1 bit for the sign and 6 bits for the difference between the vehicle's position and the cluster center (at most $63\,\text{m}$).

The range of acceptable values for ΔS is $[-15\,\text{m/s},\ 15\,\text{m/s}]$. If the difference is outside of this range, then the ΔS field will be omitted, and the *SI Flag* will be set. The *SI Flag* can take one of three possible values $\{00,\ 01,\ 10\}$:

- `00` - ΔS can be represented in the allowed range [min ΔS, max ΔS]

- `01` - $\Delta S > $ max ΔS, the vehicle is a *speeder*

- `10` - $\Delta S < $ min ΔS, the vehicle is a *lagger*

The *SI Flag* has an important application in collision warning. Many accidents are due to vehicles that are either traveling much faster than surrounding vehicles (*speeders*) or traveling much slower than surrounding vehicles (*laggers*). If drivers can be alerted to these vehicles in advance, they may be able to avoid accidents.

With CASCADE, we achieve a compression ratio of at least 86%. The primary data for each vehicle (location and speed) is represented in 136 bits (17 bytes) while the compact data for each vehicle is represented in at most 19 bits. The compression ratio is even higher if the ΔS field is omitted, as in the case of speeders and laggers.

III.3.2 Aggregation

Once compression has been completed, CASCADE forms an *aggregated cluster record*, which is a concatenation of the compact data records of the vehicles in the cluster. Each aggregated cluster record contains the following fields:

- *cluster flag* (1 bit) - indicates if the cluster contains any vehicles

- *cluster median speed* (8 bits) - the median speed of the vehicles in the cluster in meters/second

- *number of vehicles* (7 bits) - the number of vehicles contained in the cluster

- *compact data records* (19 bits each) - concatenation of all of the compact data records for vehicles in this cluster

If there is a cluster that contains no vehicles, its *cluster flag* is set to 0 and no more information about the cluster is contained in the record.

Since a cluster is four lanes wide and 126 m long, and the average vehicle length is 5 m with a minimum inter-vehicle distance of 2 m, there can be at most 72 vehicles in a cluster, which can be represented in 7 bits.

III.3.3   Initial Dissemination

Every 2 seconds, compression and aggregation is done. Once the aggregated cluster records are constructed, they are concatenated into a single frame and sent via broadcast. The aggregated frame includes the following fields:

- *type* (1 bit) - primary or aggregated frame

- *timestamp* (8 bytes)

- *aggregating vehicle's X-coordinate* (5 bits) - meters from the vehicle's local view origin, assuming 4 lanes of traffic

- *aggregating vehicle's location* (19 bytes)

- *aggregated cluster records* - up to 12 records

- *digital signature* (28 bytes)

- *certificate* (56 bytes)

- *sender's location* (16 bytes) - latitude/longitude

The signature is calculated by the aggregating vehicle over all the fields in the aggregated frame except the certificate which is signed by the certificate authority (CA) and the sender's location, which represents the location of the last vehicle that broadcast the frame.

Note that if traffic is sparse, the aggregated frame will be much smaller than the maximum 2312 bytes, because empty clusters are represented by a single bit.

The cluster aggregated records are arranged according to their place in the view, starting with the bottom-left cluster, moving from left to right, and then increasing in distance from the aggregating vehicle. Since these records are always arranged in the same manner, there is no need for a cluster ID to be included in the frame or record.

III.3.4   Rebroadcast

Aggregated frames are re-broadcast in the same manner as primary frames, using the p-IVG algorithm described in Chapter VI. Aggregated frames originating from vehicles physically behind the receiving vehicle will be dropped, as well aggregated frames that are older than the TTL. All aggregated frames have a TTL of 2 seconds, balancing timeliness of the data and the distance that the aggregated frame could travel. The propagation delay for $300\,$m is about $1\,\mu s$, which we treat as negligible. Assuming a conservative $20\,$ms processing delay at each hop, including medium access delays, a full frame can travel $300\,$m in about $23\,$ms. So, in 2 seconds, an aggregated frame could travel $26\,$km, or about 16 miles. As with primary frames, the re-broadcaster will update the sender's location field in the aggregated frame before transmitting.

III.4   DETERMINING OPTIMAL CLUSTER SIZE

In this section, we explain how we obtained the optimal cluster size of $16\,$m x $126\,$m. We investigate and determine the optimal cluster size in terms of the size of the aggregated frames created and the length of the local view that results. As primary frames contain information about only a single vehicle, they are not affected by the cluster size. In determining the optimal cluster size, we strive to find an appropriate trade-off that will minimize the aggregated frame size and maximize the local view length.

III.4.1   Aggregated Frame Size

The aggregated frame contains the compressed records of all vehicles in the aggregating vehicle's local view and is useful for vehicles behind the aggregating vehicle. The data section of the aggregated frame consists of the concatenation of all of the aggregated cluster records (ACRs) in the local view. Each ACR represents a single

cluster and consists of a header and the compact records (CRs) of all vehicles in the cluster. We will follow a bottom-up approach in investigating the relationship between the aggregated frame size and the cluster size. We first discuss of the size of a CR (describes one vehicle), then the size of an ACR (describes all of the vehicles in a single cluster), and finally the size of the aggregated frame.

Compact Record (CR)

The size of a CR depends upon the size of the fields ΔX, ΔY, ΔS, and $SIFlag$. Neither the size of ΔS (5 bits) nor the $SIFlag$ (2 bits) depend upon the cluster length, so we focus on the size of ΔX and ΔY.

The values ΔX and ΔY for each vehicle are calculated with respect to the cluster center, so the number of bits allocated to each depends on the cluster width W_C and cluster length L_C as shown in Equations 1 and 2, respectively.

$$\Delta X.size = \lceil \log_2(\frac{W_C}{2} + 1) \rceil + 1 \tag{1}$$

$$\Delta Y.size = \lceil \log_2(\frac{L_C}{2} + 1) \rceil + 1 \tag{2}$$

Recall that ΔX and ΔY are expressed using sign-magnitude representation, so an extra bit is added to hold the sign. As the cluster width and length increase, the number of bits allocated for ΔX and ΔY increases, respectively. Equation 3 shows the number of bits needed to represent a CR as a function of the cluster width and cluster length.

$$CR.size = \lceil \log_2(\frac{W_C}{2} + 1) \rceil + \lceil \log_2(\frac{L_C}{2} + 1) \rceil + 9 \tag{3}$$

Aggregated Cluster Record (ACR)

The ACR represents all of the vehicles in a cluster. The maximum number of vehicles V_{max} in a cluster depends on the size of the cluster, the size of a single lane, and the average size of a vehicle. We assume that the width of one lane is 4 m and that the average vehicle length is 5 m, so V_{max} can be expressed as in Equation 4.

$$V_{max} = \lfloor \frac{L_C}{5} \rfloor * \lfloor \frac{W_C}{4} \rfloor \tag{4}$$

The size of the ACR can be broken up into the size of the header part $ACR\_Header.size$ and the size of the data part $ACR\_Data.size$.

Frame Type (1 Bit)	Timestamp (8 Bytes)	The Sender Location		The View's Base Point			Data Section (Variable)	Signature (28 Bytes)	Certificate (56 Bytes)
		(Latitude) (8 Bytes)	(Longitude) (8 Bytes)	X-coordinate (Variable)	(Latitude) (8 Bytes)	(Longitude) (8 Bytes)			

FIG. 8: Aggregated Frame

The header section in the ACR consists of the cluster flag, cluster median speed, and number of vehicles. The sizes of both the cluster flag (1 bit) and the cluster median speed (8 bits) are constant, so they do not depend on the cluster dimensions. The number of bits allocated for the number of vehicles field V_{count} depends on the cluster size dimensions and is shown in Equation 5.

$$V_{count}.size = \lceil \log_2(V_{max} + 1) \rceil \tag{5}$$

Thus, the size of the ACR header can be represented by Equation 6.

$$ACR\_Header.size = \lceil \log_2(V_{max} + 1) \rceil + 9 \tag{6}$$

The data section of the ACR contains the CRs of all vehicles in the cluster. The maximum size of the data section is bounded by the size of the CR multiplied by V_{max} (Equation 7), while the actual size depends upon the number of vehicles in the cluster (Equation 8).

$$Max\_ACR\_Data.size = V_{max} * CR.size \tag{7}$$

$$ACR\_Data.size = V_{count} * CR.size \tag{8}$$

Aggregated Frame (AF)

Now that we have equations for the size of the CR and ACR, we can look at how the size of the aggregated frame is affected by the cluster size. The fields in the aggregated frame are shown in Figure 8. There are only two fields that are variable in size: the X-coordinate of the aggregator and the concatenation of the ACRs, labeled *Data Section* in the figure. The remaining fields contribute 993 bits to the size of the aggregated frame. The total size of the AF in bits can be represented by Equation 9.

$$AF.size = X\_Coord.size + AF\_Data.size + 993 \tag{9}$$

The *X*-coordinate of the aggregator is the position of the aggregator measured as the number of meters from the left edge of the leftmost lane. The number of bits needed to represent this (Equation 10) depends upon the size of each lane (we assume 4 m) and the number of lanes on the roadway C_L.

$$X\_Coord.size = \lceil \log_2(4 * C_L) + 1 \rceil \tag{10}$$

The *Data Section* consists of the concatenation of all of the ACRs in the local view. Each ACR represents a single cluster. If there are no vehicles in the cluster, the ACR is represented by a single bit. We show the size *AF_Data.size* in Equation 11, where *ACR.count* is the number of ACRs that can fit in the frame and v_i is the number of vehicles in cluster i.

$$AF\_Data.size = \sum_{i=1}^{ACR.count} \begin{cases} ACR\_Data_i.size & \text{if } v_i > 0 \\ 1 & \text{if } v_i = 0 \end{cases} \tag{11}$$

The maximum number of ACRs (*ACR.count*) that can be in the frame depends upon the maximum frame size (*MAC_Frame.size*), the size of the frame without the *Data Section* (*AF_nodata.size*), and the size of each ACR (*ACR.size*), as shown in Equation 12.

$$ACR.count = \lfloor \frac{MAC\_Frame.size - AF\_nodata.size}{ACR.size} \rfloor \tag{12}$$

III.4.2 Local View Length Analysis

The goal of our analysis is to find a cluster size that will minimize the aggregated frame size while maximizing the local view length. The length of the local view, which determines how much information about vehicles ahead can be passed to vehicles behind, is dependent upon the size of the aggregated frame. An important constraint on frame size is the maximum IEEE 802.11 frame size of 2312 bytes. The maximum number of clusters in the local view can be determined using Equation 12. As we are concerned with the maximum aggregated frame size, we calculate the size of the ACR in the case of having the maximum possible number of vehicles in each cluster, V_{max}. Equation 13 shows how to determine the local view length, where LC_C is the number of lanes per cluster ($LC_C = W_C/4$ since we assume that each lane is 4 m
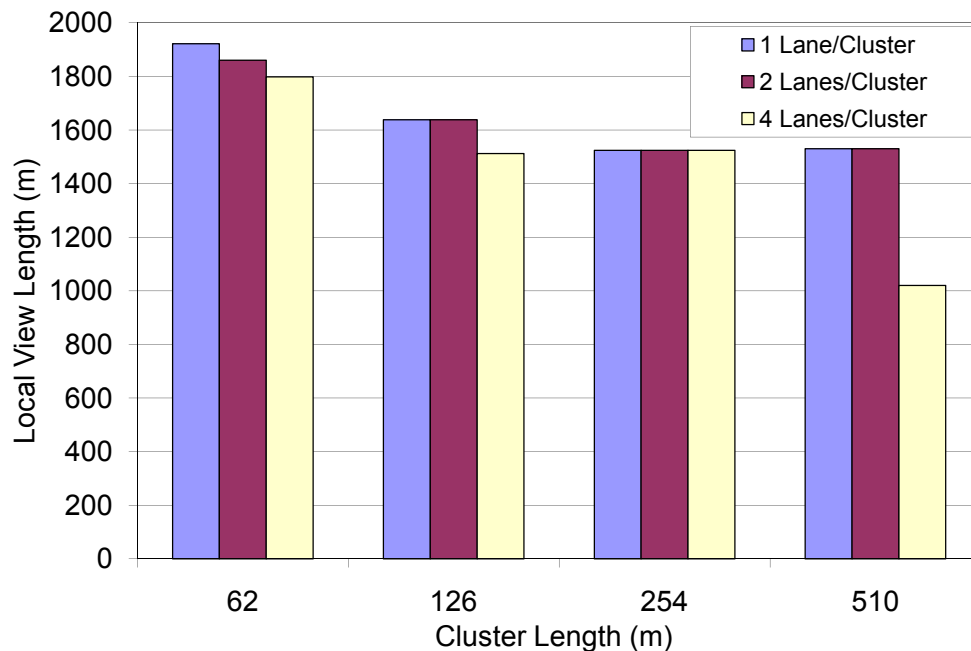
FIG. 9: Local View Length as Cluster Dimensions Change

wide), and L_C is the cluster length.

$$Length = \frac{ACR.count}{LC_C} * L_C \tag{13}$$

In our analysis, we consider four different cluster lengths (62 m, 126 m, 254 m, and 510 m) and three different cluster widths (1 lane, 2 lanes, and 4 lanes). The cluster lengths were chosen to maximize bit usage in the representation of ΔY in the compact record. Using sign-magnitude representation and 6 bits for ΔY, we can represent positions in the range [-31 m, 31 m] from the center of the cluster, for a total distance of 62 m. In the same manner, a length of 126 m can be represented in just 7 bits for ΔY, 254 m in 8 bits, and 510 m in 9 bits.

Figure 9 shows the local view length of each of the cluster dimensions. The clusters with length 62 m have the longest local views. Within this group, the local view length decreases as the cluster width increases. For clusters with length 126 m, both the clusters with one lane width and two lanes width have the same local view length, while the cluster with four lanes width has a shorter local view. All the clusters with 254 m length have the same cluster length regardless of the cluster width. The last group, with cluster length 510 m, behaves similar to the second group, where the clusters with one lane width and two lanes width have the same
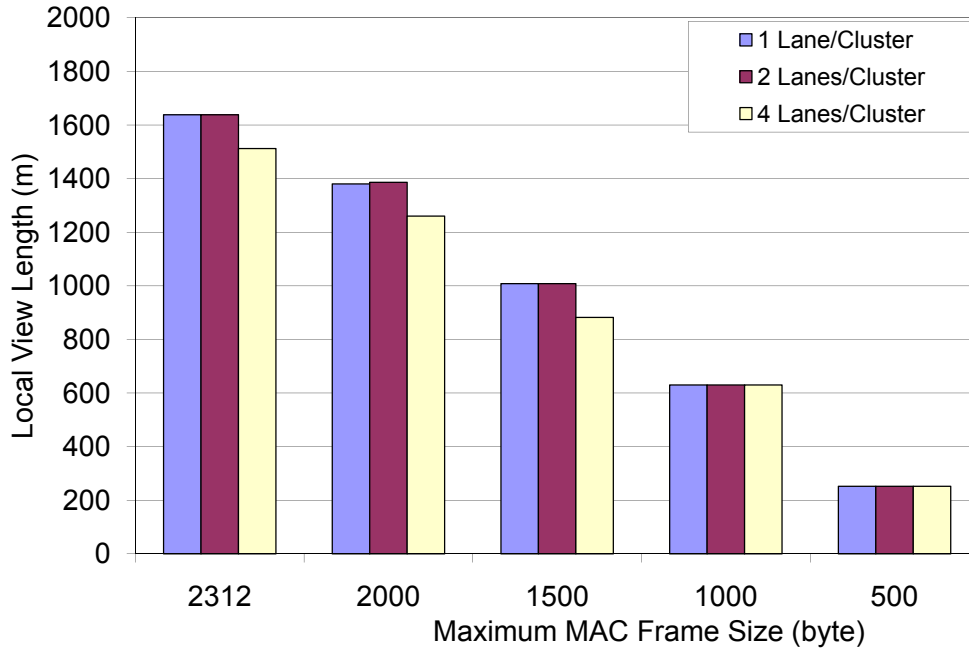
FIG. 10: Local View Length as the Frame Size Changes: Using Cluster Length of 126 m

local view length, and the local view length for the cluster with four lanes width is the shortest of all the other cluster sizes. Since the cluster with 510 m length and four lanes width provides by far the shortest local view, we eliminate it from further analysis.

The frame size that we used in the analysis is the maximum IEEE 802.11 frame size of 2312 bytes. Changing the frame size will result in different maximum local view lengths, considering the same number of lanes per cluster. If the frame size increases to be greater than 2312 bytes, it will definitely result in longer local view length but also it will result in the frame being fragmented, which is not recommended. So the option for increasing the frame size is not viable. Figure 10 shows the impact of decreasing the frame size on the local view length when the cluster length is fixed to length 126 m. There is a tradeoff with using a larger frame size, as it takes longer to transmit and therefore will hold the channel longer.

III.4.3 Aggregated Frame Size Analysis

The cluster dimensions are not the only factors that affect the size of the aggregated frame. Two other factors are the number of vehicles in the local view (ranging
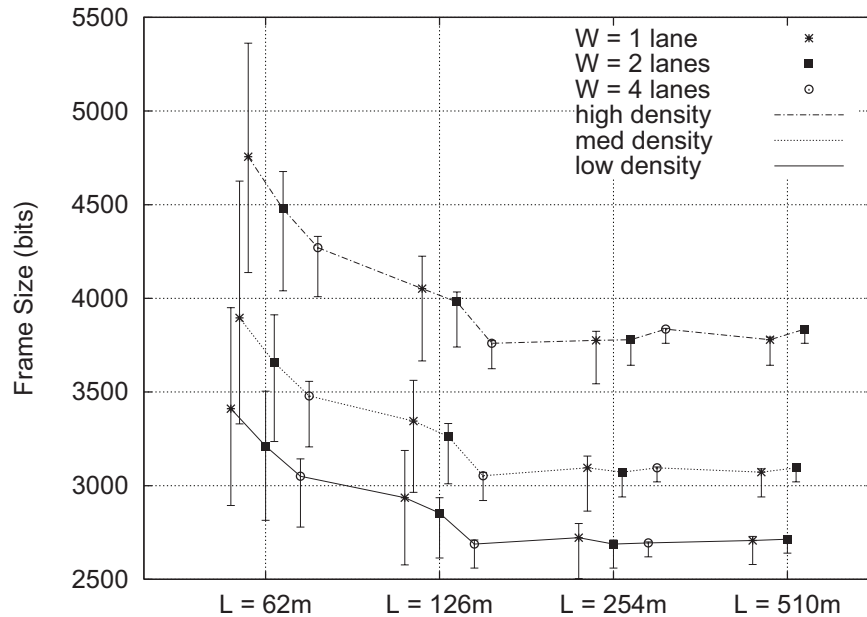
FIG. 11: Minimum, Maximum, and Expected Aggregated Frame Sizes as Cluster Dimensions and Traffic Density Change

between 0 and $ACR.count * V_{max}$) and how these vehicles are distributed over the local view clusters.

We calculate the aggregated frame size for various cluster sizes and also considering different traffic densities. We consider 53 vehicles/km as low density, 66 vehicles/km as medium density, and 90 vehicles/km as high density. For each traffic density, as the cluster dimensions change, the associated local view will change, which implies that the total number of vehicles in the local view, N, will change as well, as shown in Table 1. In the table, M is the number of clusters in the local view, and K is the maximum number of vehicles per cluster. We distribute the vehicles over both the worst-case distribution to find the maximum frame size and the best-case distribution to find the minimum frame size. Figure 11 shows the minimum, maximum, and expected value of the aggregated frame size over different cluster sizes and traffic densities. (The calculation of the expected value of the aggregated frame size is given later in this section) Each vertical line represents the possible frame sizes for the specific cluster dimension. The lowest point on the line is the minimum frame size, the highest point on the line is the maximum frame size, and the symbol in between is the expected value of the frame size. The expected value points for each traffic density are connected by a line to highlight the minimum expected frame size

for that density. As explained in Section III.4.2, the case of cluster length 510 m with a width of four lanes has been omitted.

From Figures 9 and 11, we find that a cluster width of 4 lanes and length of 126 m provides a small frame size along with long local view over various traffic densities.

To study whether increasing the cluster width to have more than 4 lanes will benefit the local view or not, we calculated the local view length for a highway with 5 lanes in a cluster. In this case, the local view length dropped drastically, as shown in Table 2, which implies that increasing the cluster width to more than 4 lanes will provide no benefit. To conclude, having a cluster with width of 4 lanes and length of 126 m has proved to be the optimal among any other cluster dimensions in CASCADE.

Although using the optimal cluster settings (4 lanes width and 126 m length) will provide the best results, CASCADE will have to reconfigure itself on detecting a highway with more than 4 lanes in order to accommodate the extra lanes. In this case, it will continue to use the cluster length of 126 m and the number of lanes in the cluster will be the same as the number of lanes in the highway. For example, if the highway has 6 lanes, the local view component in CASCADE will dynamically reconfigure the cluster dimensions to be 126 m long and 6 lanes wide. And as a consequence, the local view length will not be 1.5 km, and it needs to be calculated as part of the new configuration.

In the ideal situation, a vehicle's aggregated frame will contain data about all vehicles in its local view. But, in some situations where available bandwidth is limited, the vehicle will limit the number of clusters that it includes in the broadcasted frame. To handle these situations, each aggregated frame should have a field, *ClustersCount*, that states the number of clusters (including empty clusters) represented in the frame. Having *ClustersCount* as part of the broadcasted frame will allow the receiving vehicle to determine the maximum length for the constructed view from the received frame.

Calculating The Expected Frame Size

We present the calculation of the expected value of the aggregated frame size. Each frame size corresponds to a different distribution of vehicles over the local view clusters. Some of the vehicle distributions are more frequent than others, so they occur with higher probability. The problem of distributing the vehicles over the local view

TABLE 1: Data for All Cluster Dimensions

Cluster		Local View			Low Density	Medium Density	High Density
Length (m)	Width (lanes)	Length (m)	M	K	N	N	N
62	1	1922	124	8	101	126	172
62	2	1860	60	16	98	122	167
62	4	1798	26	32	95	118	161
126	1	1638	52	18	86	108	147
126	2	1638	26	36	86	108	147
126	4	1512	12	72	80	99	136
254	1	1524	24	36	80	100	137
254	2	1524	12	72	80	100	137
254	4	1524	6	144	80	100	137
510	1	1530	12	72	81	100	137
510	2	1530	6	144	81	100	137
510	4	1020	2	288	54	67	91

TABLE 2: Comparing Local View Lengths in Case of Cluster Width of 4 Lanes and 5 Lanes

Cluster		Local View
L_C (m)	W_C (lanes)	Length (m)
62	4	1798
62	5	1426
126	4	1512
126	5	1260
254	4	1524
254	5	1016
510	4	1020
510	5	1020

clusters is similar to the occupancy problem with limited capacity.

Let us assume that we have M clusters, N vehicles, and the maximum capacity for each cluster cannot exceed K vehicles. The vehicles can be distributed over the clusters as in Equation 14.

$$A_i = (k_1, k_2, \ldots, k_r) \tag{14}$$

where $0 \leq k_i \leq K$ and $r \leq \lceil \frac{N}{K} \rceil$.

A_i is the general form for the possible distributions in which the number of vehicles in the i_{th} cluster is exactly k_i vehicles. The total number of all possible vehicle distributions over the clusters, assuming the clusters are distinguishable, can be

calculated from Equation 15.

$$Count(N, M, K) = \sum_{i=0}^{\lfloor \frac{N}{K+1} \rfloor} (-1)^i \binom{M}{i}$$
$$\binom{N + M - i(K + 1) - 1}{M - i} \tag{15}$$

Each possible distribution A_i will have an associated frame size, but it is possible for a frame size to be associated with more than one distribution. To calculate the expected frame size, we have to calculate the probability of each frame size, thus we have to count all possible distributions that can generate this frame size. If we study Equation 11 from Section III.4.1 carefully, we notice that the frame size depends on the number of vehicles N, the number of clusters M, and the number of empty clusters. As M and N are constant for each possible cluster dimension, the only factor that will affect the frame size will be the number of empty clusters. The frame size FS_j is generated when having j empty cluster(s). The count of all possible distributions that can generate the frame size FS_j can be calculated using Equation 15, after modifying the parameters M, N, and K to be M', N', and K' as in Equation 16.

$$Count_j(N', M', K') = \sum_{i=0}^{\lfloor \frac{N'}{K'+1} \rfloor} (-1)^i \binom{M'}{i}$$
$$\binom{N' + M' - i(K' + 1) - 1}{M' - i} \tag{16}$$

where j is the number of empty clusters $0 \leq j \leq M - \lceil \frac{N}{K} \rceil$, $M' = M - j$, $N' = N - M'$, and $K' = K - 1$.

Using Equation 15 and 16 we can calculate the probability of having frame size FS_j, as in Equation 17.

$$Pr(FS_j) = \frac{Count_j(N', M', K')}{Count(N, M, K)} \tag{17}$$

Hence, the expected frame size will be calculated using Equation 18.

$$E(FS) = \sum_{j=0}^{M - \lceil \frac{N}{K} \rceil} Pr(FS_j) * FS_j \tag{18}$$

Using the previous equations, we can calculate the expected frame size for the possible cluster dimensions with different traffic densities. In fact, this was done to create the graph in Figure 11.

III.5 SUMMARY

In this chapter, we have presented the local view component. The local view presents data gathered from primary records, which are sent in signed frames containing a vehicle's position information. The local view is grouped into clusters, which are then used to compact and aggregate the local view data. Aggregated data from other vehicles can be used to extend a vehicle's view past its 1.5 km local view. Since vehicles' positions and speeds are represented as differences from the cluster data rather than combined with other vehicles' data, the accuracy of the aggregated data in our system is very high. Also, we have presented an analysis of the CASCADE data aggregation technique. In our analysis, we determined that a cluster size 16 m wide and 126 m long would provide the best trade-off between frame size and local view length. Having such an optimal cluster size will reduce the bandwidth consumption and provide better extended driver visibility. The evaluation of both the vehicular data compression technique and data dissemination technique will be presented in Chapter VII.

CHAPTER IV

EXTENDED VIEW COMPONENT

In Chapter III, we explained how each vehicle builds its local view out of the received primary frames and why its length is limited to 1.5 km. Also, we explained how the local view is compressed, aggregated and broadcasted as an aggregated frame. In this chapter, we focus on how each vehicle will use the received aggregated frames to extend its local view and hence increase its knowledge about the traffic ahead beyond the 1.5 km limit.

Upon receiving an aggregated frame, the receiver will validate and decompress it then the original view will be constructed out of this received frame, as described in Chapter III. At this time, the receiving vehicle will have two traffic views, one is its own local view (LV) and one has been constructed out of the received aggregated frame, the received local view (RLV). To extend the local view, LV, using the data in the received local view, RLV, we should first determine the intersection, or overlap, percentage between these two views. The intersection percentage can range from 0% to 100%, as shown in Figure 12. It will be equal to 100% as in Figure 12-c, when the vehicle broadcasting the RLV frame, V_1, is at the same X position as the vehicle receiving that frame, V_2. While, the intersection percentage will be equal to 0%, (Figure 12-a), when the vehicle broadcasting the RLV frame, V_1, is 1.5 km ahead of the vehicle receiving that frame, V_2. Figure 12-b shows an example for the case when the intersection percentage has a value in between 0% and 100%.

The problem that we are addressing in this chapter is that we have received a view, RLV, from another vehicle and we need to decide whether we want to accept it as true or not. The way we do that is to compare the vehicles in the intersecting region. If the matching percentage of the vehicles in the intersecting region passes some threshold, we say the RLV is validated. So, we accept the entire RLV and use it to extend the local view. The challenge is in comparing the vehicles in the different views considering different overlapping percentages between the local view, LV, and the received view, RLV, as shown in Figure 13.

This chapter will be organized as follows. In Section IV.1, we present two possible data structures that can be used to represent the vehicular data in the intersection area. In Section IV.2, we present the function that will be used to compare any

(a) 0% intersection   (b) 30% intersection   (c) 100% intersection
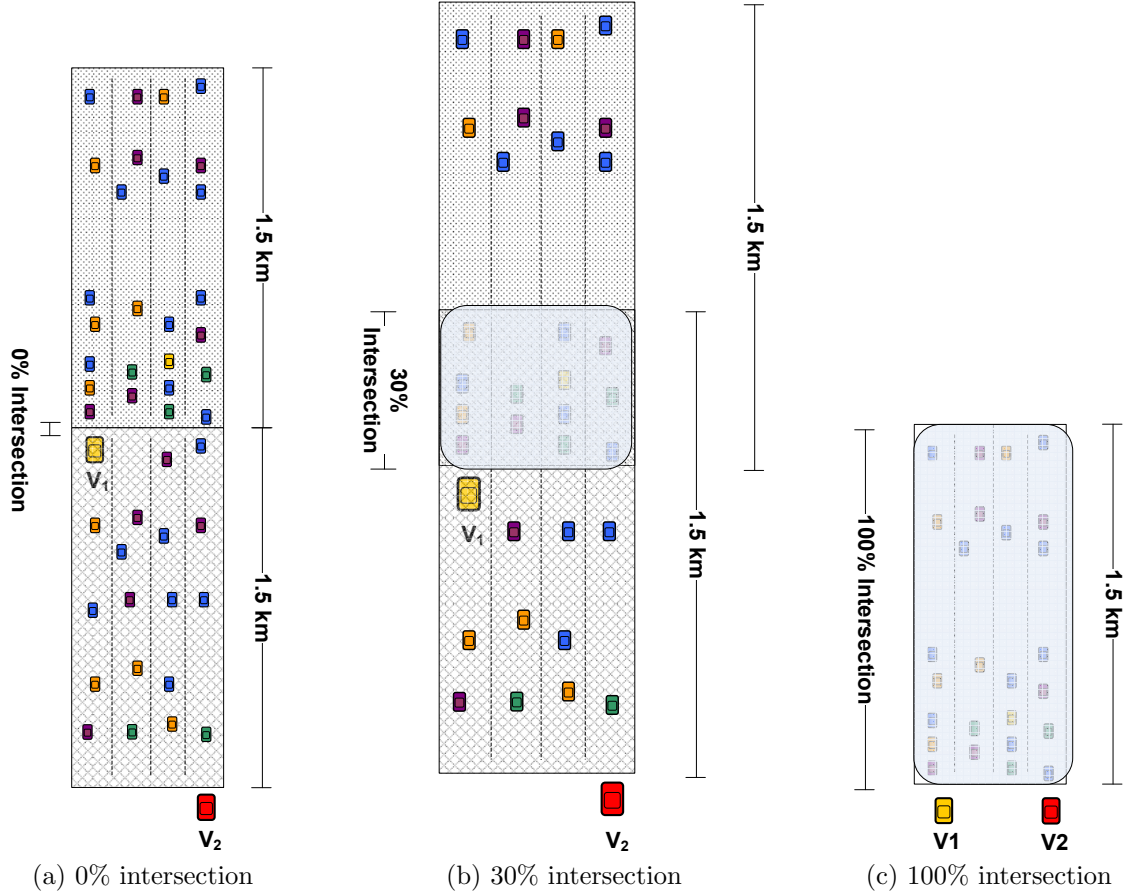
FIG. 12: Different Intersection Percentages Between RLV and LV Views

two vehicles' data[1] to determine how similar they are. Section IV.3 discusses how to compare the vehicular data in the overlapping area and build the best matching list, BML. In Section IV.4, we discuss the possible transitions that will happen for the vehicle's matching state as the matching threshold changes. On comparing two vehicles' data, the threshold that will be used to justify whether these two vehicles are matching or not will be discussed in Section IV.5. Section IV.6 focuses on finding the matching threshold for justifying whether the two views, LV and RLV, are matching or not.

The experiments performed in this chapter assume that each vehicle broadcasts its ID as part of the primary frame. This assumption is valid because these experiments are performed offline during the calibration phase to find the optimal value for some

---

[1]The data that we are concerned with in this context are the vehicle's X position, Y position and speed.
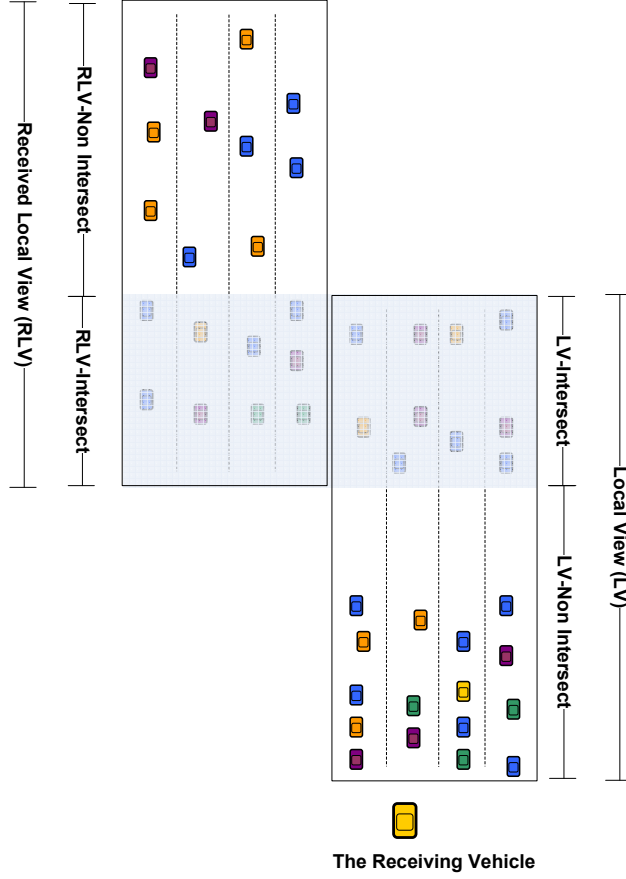
FIG. 13: LV and RLV Different Sections

parameters. In real situations, the vehicular data is broadcasted anonymously for privacy reasons.

IV.1 ALTERNATIVE DATA STRUCTURES FOR VEHICULAR DATA REPRESENTATION

Figure 13 shows an RLV frame received and constructed by the receiving vehicle that intersects with the vehicle's LV. As a result of the intersection, each view will be divided into two sections, intersecting and non-intersecting, so we assign them different names to avoid any ambiguity when referring to them.

The vehicular data that will be used to compare any two vehicles consists of three components: X position, Y position and speed. In this section, we focus on finding a suitable data structure that can used to represent the vehicular data in the LV-Intersect and RLV-Intersect sections. Selecting the appropriate data structure will

directly impact the performance of building the extended view module and hence the overall system performance. There are multiple data structure alternatives that can be used in representing the vehicular data. Out of these alternatives, we chose the graph and the KD-Tree as they are the most appropriate data structures to represent data with multiple components or a multi-dimensional nature. In the following sections IV.1.1 and IV.1.2, we are going to show how the graph and the KD-Tree can be used to represent the vehicular data. Every experiment in this chapter will be run twice, one run using the graph data structure and one run using the KD-Tree data structure. Our main goal here is to find any evidence in the experiment results, mainly the accuracy, that can favor one data structure over the other. Also, *CPU usage*, *memory consumption* and *turn around time*[2] will be tracked along with each experiment.

IV.1.1   Graph Data Structure Representation

There are two sets of vehicular data, LV-Intersect and RLV-Intersect, that need to be represented using the graph data structure. Converting these vehicular data sets into their equivalent graph data structure representations involves two steps. In the first step, each of these two sets will be represented as a disconnected graph so that each vehicle in LV-Intersect and RLV-Intersect will be represented by a vertex in the LV-Intersect graph and RLV-Intersect graph, respectively, as shown in Figure 14. In the second step, we will connect each vertex in one set to all the vertices in the other set using weighted edges (Figure 15). The weight of each edge will be calculated using the function *Vehicular_Data_Comparer* that will be described in more detail in Section IV.2. This edge weight represents how different the two represented vehicles are. The smaller the edge weight, the more similar the two vehicles.

IV.1.2   KD-Tree Data Structure Representation

A KD-Tree is a binary tree that is useful in representing data that has a multi-dimensional nature. Most of the applications that require multi-dimensional search key, *i.e.* nearest neighbor searches and range searches, use the KD-Tree as their main data structure. The algorithm complexity for building a static KD-Tree is $O(n \log^2 n)$, both the insertion and removal of a single point in a balanced KD-Tree

---

[2]The turn around time is the time the matching algorithm will take to decide whether the given two views, LV and RLV, are matching or not.
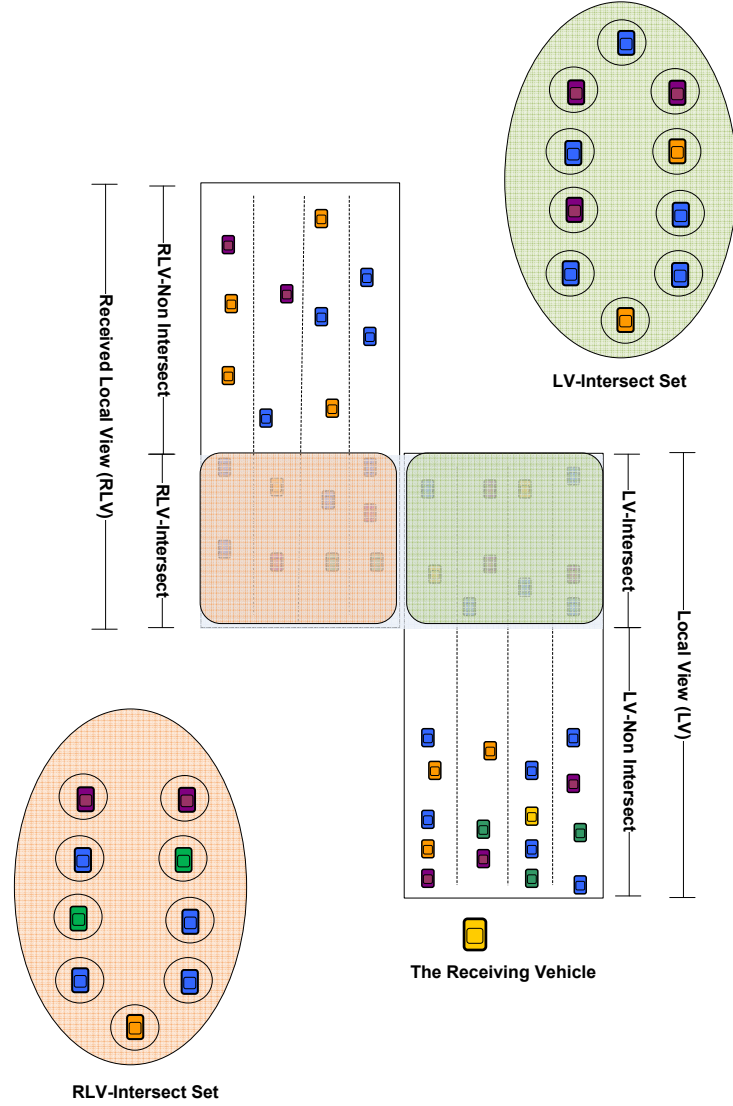
FIG. 14: Graph Representation for LV-Intersect and RLV-Intersect Data: Step 1

take $O(\log n)$ [83]. The vehicular data has multiple components, but in our case we are concerned with only three components: X position, Y position and speed. Each component can be represented in a separate dimension, which makes the KD-Tree a good fit for representing the vehicular data. Figure 16 shows an example of how a KD-Tree can be used to represent vehicular data. In Figure 16-a, we see 7 vehicles driving along a highway segment. The exact vehicular data for these vehicles are in the table on the figure's right. Figure 16-b shows the vehicular data represented in a KD-Tree data structure. The first dimension that we use to split the vehicular data is the X-Position at the KD-Tree root or level 0, so that all the nodes in the root's left
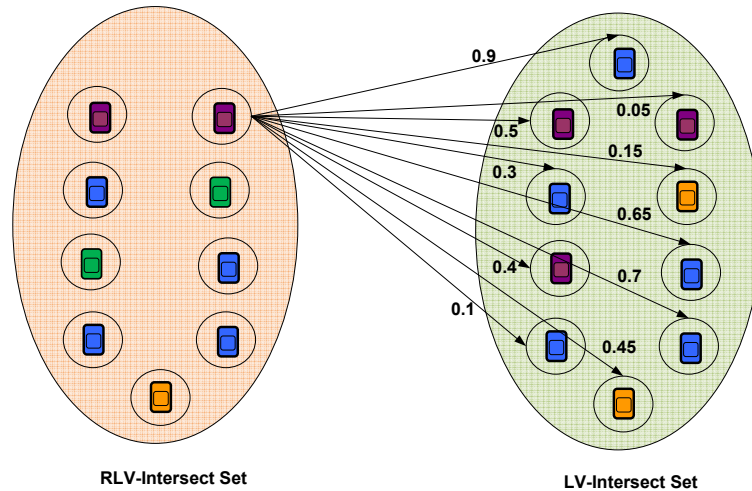
FIG. 15: Graph Representation for LV-Intersect and RLV-Intersect Data: Step 2

branch will have X-Positions less than 9, which is the X-Position for the vehicle at the root node, and all the nodes in the root's right branch will have X-Positions greater than 9. The Y-Position is the second dimension that we use to split the vehicular data at KD-Tree level 1 and at KD-Tree level 2. Speed is the third dimension to split the vehicular data on. At level 3 in the KD-Tree, we start to split again on the X-Position dimension. Figure 17 shows how these three vehicular data dimensions are used to split the space in a 3-dimensional KD-Tree.

Similar to representing the vehicular data as a graph, the KD-Tree representation for the vehicular data consists of two steps or phases, the KD-Tree construction phase and the KD-Tree probing phase. The KD-Tree representation is different than the graph representation, as only the vehicular data in the LV-Intersect region will be represented as a KD-Tree. The vehicular data in the RLV-Intersect will be used to probe the constructed tree. After the KD-Tree construction phase, the vehicular data for each vehicle in the LV-Intersect will be represented as a node in the KD-Tree, as shown in Figure 18. In the probing phase, the vehicular data for each vehicle in the RLV-Intersect will be used to probe the KD-Tree, looking for the best match among the vehicles in the LV-Intersect set. Each of the probed nodes in the KD-Tree will maintain a list to store the probing nodes' data and the corresponding $diff\_percentage$, as shown in Figure 19.

In the graph representation, the $diff\_percentage$ is calculated between each vehicle in the RLV-Intersect graph and all the vehicles in the LV-Intersect graph. While
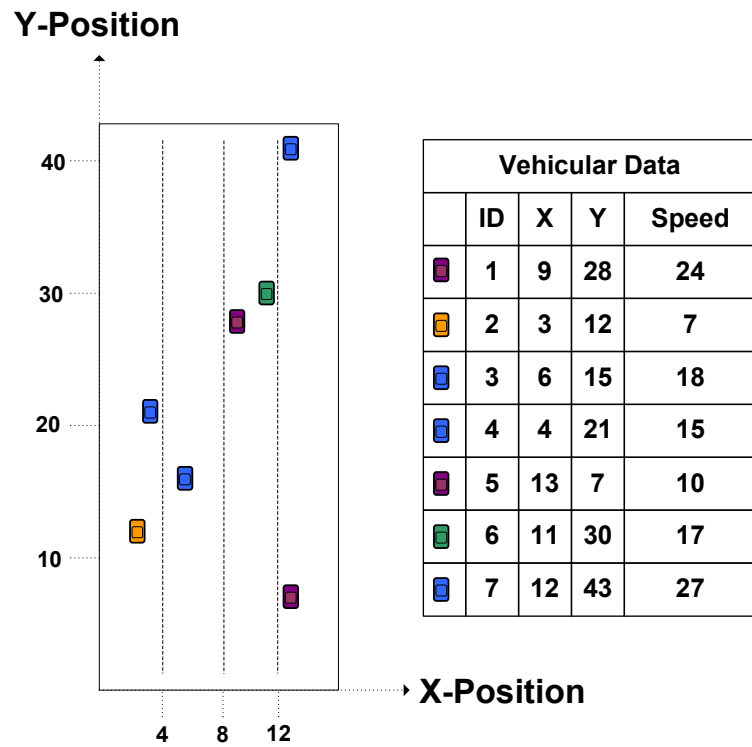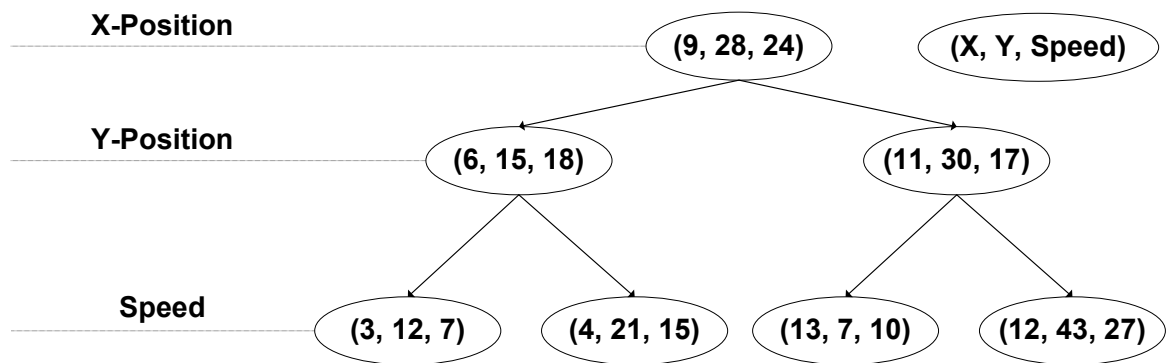
FIG. 16: Representing Vehicular Data in a KD-Tree Data Structure
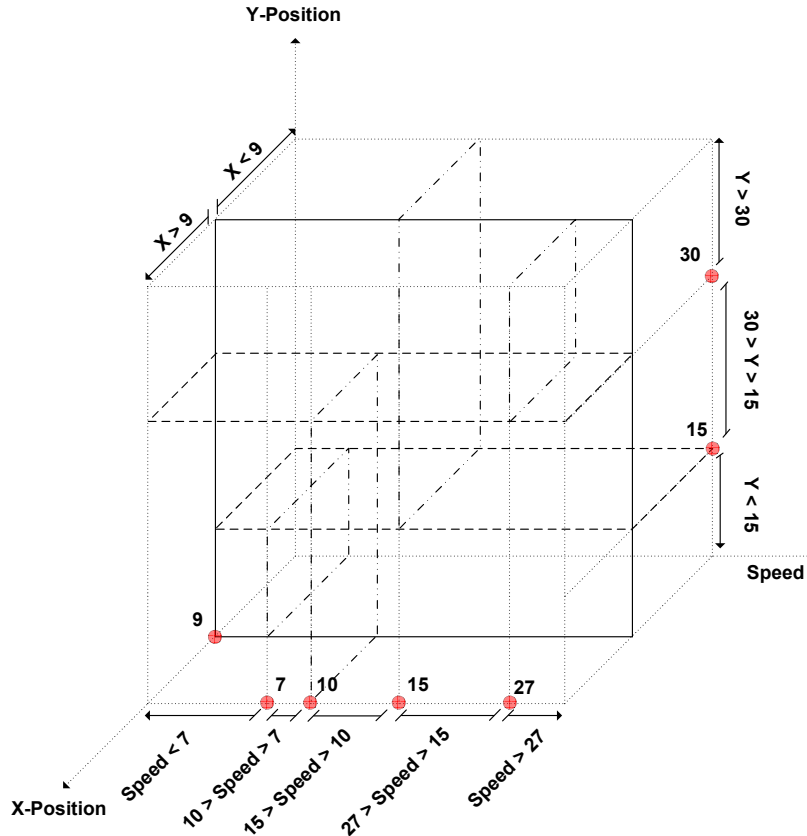
FIG. 17: 3-Dimensional KD-Tree Representation for the Vehicular Data

in the KD-Tree representation, the $diff\_percentage$ is calculated between each vehicle in the RLV-Intersect and only the vehicles in the LV-Intersect that were along the probing path[3] in the KD-Tree.

IV.2   VEHICULAR DATA COMPARISON FUNCTION

The goal of this section is to determine the similarity between two vehicles' data. The solution provided for this problem does not provide a binary answer, *i.e.*, similar or not similar, but rather a percentage that shows how different the two vehicles' data are. For example, if the $diff\_percentage$ is 0%, this means there is no difference between these two vehicles' data, which implies they are an exact match. As the $diff\_percentage$ increases, this shows that there are some differences. When $diff\_percentage$ reaches 100%, this means these two vehicles' data are completely

---

[3]The probing path is the sequence of KD-Tree nodes that each vehicle in the RLV-Intersect will visit during the probing phase.
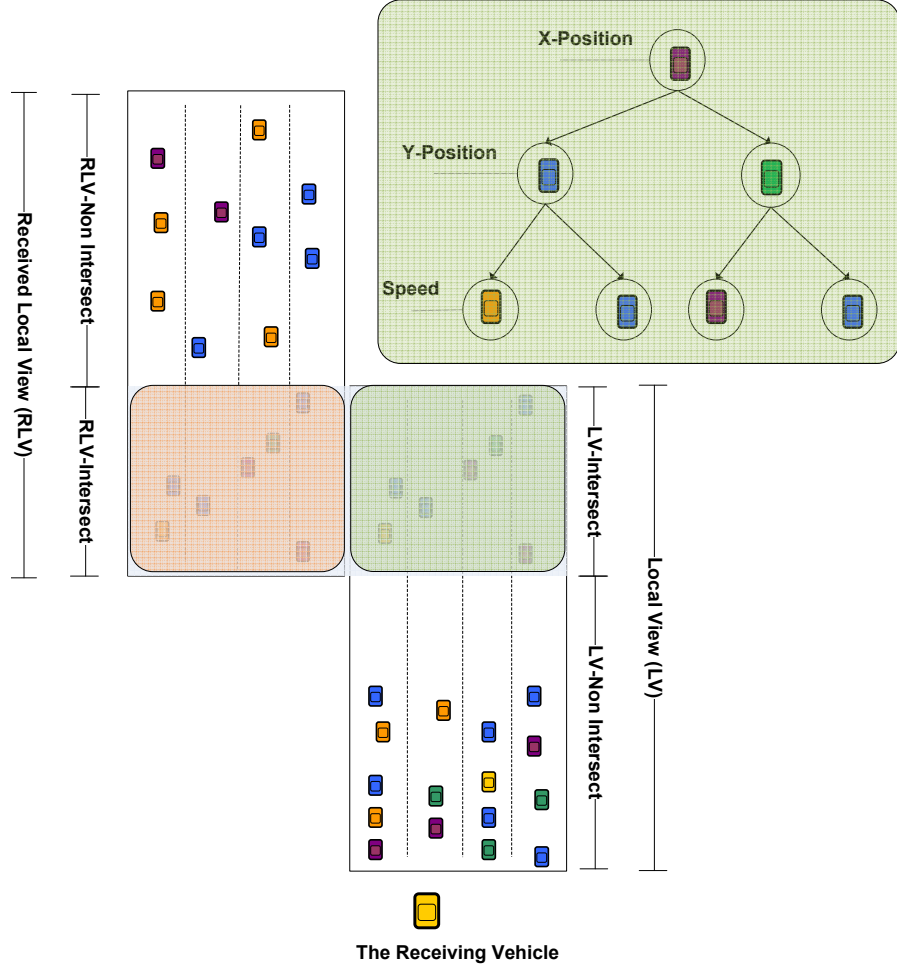
FIG. 18: KD-Tree Representation for LV-Intersect Data: Step 1

different.

$diff\_percentage$ is the overall difference percentage at the vehicular data level, including the differences at each vehicular data component, X position, Y position and speed. To compute the $diff\_percentage$, we should first compute the difference percentage for each component in the vehicular data, then average them. The vehicular data components that we are concerned with are X position, Y position and speed, so the corresponding absolute differences are,

$$\triangle X = |X_{V_1} - X_{V_2}|,$$

$$\triangle Y = |Y_{V_1} - Y_{V_2}|,$$

$$\triangle Speed = |Speed_{V_1} - Speed_{V_2}|,$$
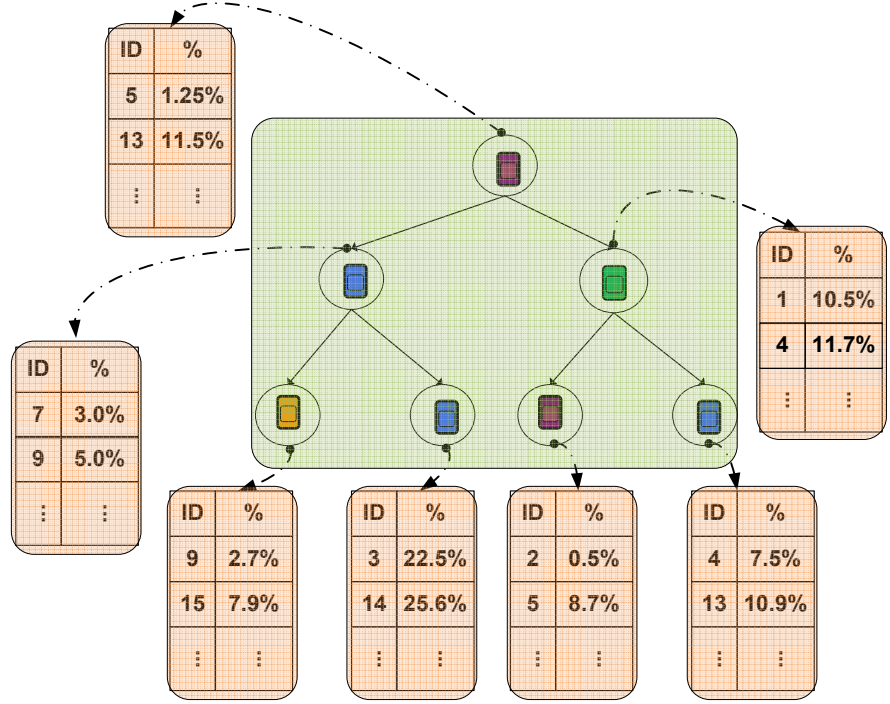
FIG. 19: KD-Tree Representation for LV-Intersect and RLV-Intersect Data: Step 2

where $(X_{V_1}, Y_{V_1}, Speed_{V_1})$ and $(X_{V_2}, Y_{V_2}, Speed_{V_2})$ are the X position, Y position and speed for two vehicles V_1 and V_2, respectively. To compute the difference percentage at the component level, $diff\_percentage_X$, $diff\_percentage_Y$ and $diff\_percentage_{Speed}$, we should find a reference value for each component, which is the maximum possible value for this component. We use the optimal local view length, $1.5\,km$, and width, $16\,m$, as reference values for $diff\_percentage_X$ and $diff\_percentage_Y$, while for $diff\_percentage_{Speed}$, the speed of the vehicle in the LV frame, $Speed_{V_{LV}}$, will be used as the reference value. $Speed_{V_{LV}}$ can be either $Speed_{V_1}$ or $Speed_{V_2}$ depending on which of them belong to the LV frame. The following are the equations used to compute the difference percentage for each competent,

$$diff\_percentage_X = \frac{\triangle X}{1500},$$

$$diff\_percentage_Y = \frac{\triangle Y}{16},$$

$$diff\_percentage_{Speed} = \frac{\triangle Speed}{Speed_{V_{LV}}},$$

and hence the overall difference percentage equation will be,

$$diff\_percentage = \frac{diff\_percentage_X + diff\_percentage_Y + diff\_percentage_{Speed}}{3}$$

TABLE 3: Best Matching List, BML

LV-InterSect Vehicular Data	RLV-InterSect Vehicular Data	$diff\_percentage$
$Vehicle._1$	$Vehicle._1$	1.4%
$Vehicle._2$	$Vehicle._9$	0.8%
$Vehicle._{25}$	$Vehicle._{25}$	20.4%
$Vehicle._4$	$Vehicle._{18}$	70.6%
$\vdots$	$\vdots$	$\vdots$
$Vehicle._n$	$Vehicle._m$	x.y%

All of the above equations are implemented in the *Vehicular_Data_Comparer* function that simply consumes the vehicular data for two vehicles and outputs the overall difference percentage, $diff\_percentage$.

IV.3 CONSTRUCTING THE BEST MATCHING LIST (BML)

After representing the vehicular data in the LV-Intersect and RLV-Intersect using either graph or KD-Tree data structure representation, these vehicular data will be processed to generate a list of vehicle pairs as shown in Table 3. Each pair in this list has two vehicles, one vehicle from the LV-Intersect and the other from the RLV-Intersect and their corresponding $diff\_percentage$.

In case of the graph representation, the best matching list (BML) is constructed through the following steps:

1. Find the edge with the smallest weight (*i.e.*, closest match), then create a new entry in the BML to record the two vehicles, $vehicle_{LV}$ and $vehicle_{RLV}$, that this edge is connecting along with the edge weight, $diff\_percentage$

2. Remove the edge and corresponding two nodes, then clean the graph by removing all the edges that used to connect any of the two removed nodes to any other node in the graph

3. As long as there are more edges go back to Step 1

In case of the KD-Tree representation, the best matching list (BML) is constructed through the following steps:

1. Visit each node in the KD-Tree using depth-first search

2. Search the list associated with the current node for the RLV-Intersect vehicle with the minimum $diff\_percentage$, and save this vehicle as the local minimum for this node

3. At the end of traversing the tree, identify the global minimum entry in the tree (*i.e.*, the RLV-Intersect vehicle with the smallest $diff\_percentage$). Then, create a new entry in the BML to record the two vehicles, $vehicle_{RLV}$ the global minimum entry in the list and $vehicle_{LV}$ the node that had the list with the global minimum

4. Traverse the tree branch where the node with the global minimum exists and remove the $vehicle_{RLV}$ from all the other nodes' lists

5. Delete the node that has the $vehicle_{LV}$ with the global minimum

6. As long as there are more nodes go back to Step 1

IV.4   VEHICLE'S MATCHING STATE TRANSITION DIAGRAM

Each entry in the best matching list (BML) represents a pair of vehicles and their corresponding $diff\_percentage$. This pair can be classified to be in one of the four states shown in Figure 20 based on the comparison result of $diff\_percentage$ to $diff\_percentage\_thr$ and the vehicles' IDs. The following is a detailed explanation of the four states.

- *TruePositive* state: a pair will be in this state if its $diff\_percentage$ is less than $diff\_percentage\_thr$ and the vehicles' IDs are equal.

- *FalsePositive* state: a pair will be in this state if its $diff\_percentage$ is less than $diff\_percentage\_thr$ and the vehicles' IDs are not equal.

- *TrueNegative* state: a pair will be in this state if its $diff\_percentage$ is greater than or equal to $diff\_percentage\_thr$ and the vehicles' IDs are equal.

- *FalseNegative* state: a pair will be in this state if its $diff\_percentage$ is greater than or equal to $diff\_percentage\_thr$ and the vehicles' IDs are not equal.
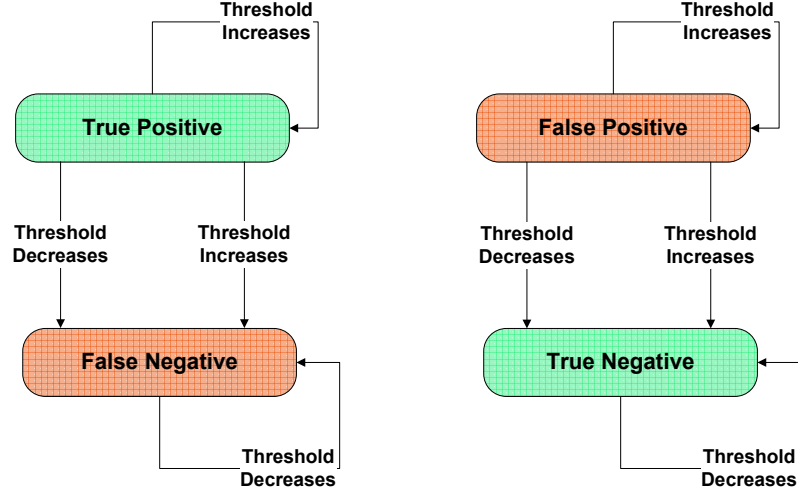
FIG. 20: The State Transition Diagram for the 4 Possible Received Vehicular Data Categories as the *diff_percentage_thr* Varies

As we see in Figure 20, the four states are clustered into two groups, one group has the *TruePositive* and *FalseNegative* states, while the other one has the *FalsePositive* and *TrueNegative*. The only variable that can be changed is the *diff_percentage_thr* and according to its change direction, increase or decrease, a transition may happen from one state to another within the same group.

To explain how a transition between two states can happen, let us assume that we have a pair in *TruePositive* state, which means the *diff_percentage* is less than the current *diff_percentage_thr* and the two vehicles composing this pair have the same ID. So increasing the *diff_percentage_thr*, which means relaxing the matching criteria, will keep this pair in the same state. Enforcing a more conservative matching criteria by decreasing the *diff_percentage_thr* till it becomes equal to or greater than *diff_percentage* will cause a state transition. Although the two vehicles in this pair have the same ID, the pair will be classified as *FalseNegative* because its *diff_percentage* does not satisfy the matching criteria, *diff_percentage < diff_percentage_thr*. Starting with a pair at *FalseNegative* state will have the opposite state transition logic, because decreasing the *diff_percentage_thr* keeps the pair in the same state and increasing the *diff_percentage_thr* till it becomes greater than *diff_percentage* will trigger a state transition, so that the pair will be in the *TruePositive* state. The same state transition logic applies for the two states *FalsePositive* and *TrueNegative* in the other group.

IV.5 DIFFERENCE PERCENTAGE THRESHOLD AT VEHICLE LEVEL

Although the *Vehicular_Data_Comparer* function, explained in Section IV.2, provides us with the $diff\_percentage$ between any two vehicles, it does not tell us whether these two vehicles are matching or not. To be able to make this decision, we should first compare their $diff\_percentage$ against the difference percentage threshold, $diff\_percentage\_thr$, that we are dedicating this section to determine. Using the following equations we can make our decision,

$$diff\_percentage < diff\_percentage\_thr \Rightarrow matching,$$

$$diff\_percentage \geq diff\_percentage\_thr \Rightarrow not\,matching,$$

Section IV.3 ended with a list of vehicle pairs, the BML. Each pair has one vehicle from the LV-Intersect set, while the other vehicle is from RLV-Intersect. The common properties among these pairs is that the $diff\_percentage$ between the two vehicles in any pair is the minimum possible, so that the sum of all pairs' $diff\_percentage$ is minimum. Although each pair has the minimum $diff\_percentage$ between its two vehicles, it does not mean these two vehicles are matching because the $diff\_percentage$ should be checked against the $diff\_percentage\_thr$, as in the previous equations, then each pair can be tagged as matching or not matching.

Based on comparing $diff\_percentage$ for each pair to the $diff\_percentage\_thr$, we classify or tag each pair as either:

- *Matching* (or *Positive*)

Or

- *NotMatching* (or *Negative*)

Note that just because a pair is classified into certain category of the above two categories does not mean that we are 100% positive of the classification. This is because the vehicular data is broadcasted anonymously, without the vehicles unique ID, to protect the vehicles privacy. So it may happen to tag a pair as matching, while the two vehicles in it are different vehicles (have different IDs). Also, it may happen to tag a pair as not matching, while the two vehicles in it are the same vehicle (have same IDs). Based on comparing the vehicles' ID in each pair, we can classify each of the previous two categories even further:

- *TrueMatching* (or *TruePositive*): a pair will be marked as *TrueMatching* if the following two conditions are satisfied

$$diff\_percentage < diff\_percentage\_thr$$

$$\&\&$$

$$RLV.Vehicle._{ID} = LV.Vehicle._{ID}$$

- *FalseMatching* (or *FalsePositive*): a pair will be marked as *FalseMatching* if the following two conditions are satisfied

$$diff\_percentage < diff\_percentage\_thr$$

$$\&\&$$

$$RLV.Vehicle._{ID} \neq LV.Vehicle._{ID}$$

- *TrueNotMatching* (or *TrueNegative*): a pair will be marked as *TrueNotMatching* if the following two conditions are satisfied

$$diff\_percentage \geq diff\_percentage\_thr$$

$$\&\&$$

$$RLV.Vehicle._{ID} \neq LV.Vehicle._{ID}$$

- *FalseNotMatching* (or *FalseNegative*): a pair will be marked as *FalseNotMatching* if the following two conditions are satisfied

$$diff\_percentage \geq diff\_percentage\_thr$$

$$\&\&$$

$$RLV.Vehicle._{ID} = LV.Vehicle._{ID}$$

Table 4 shows an example for a *BML* after classifying each pair into one of the above four categories.

What controls tagging a pair as matching or not is the *diff_percentage_thr*. The smaller the *diff_percentage_thr* value, the more pairs being tagged as not matching and less pair being tagged as matching and vice versa. Having small *diff_percentage_thr* can be translated as a conservative matching criteria, because

TABLE 4: Classifying Each Pair in the BML as $TrueMatching$, $FasleMatching$, $TrueNotMatching$ or $FalseNotMatching$, assuming a $diff\_percentage\_thr = 17.0\%$

LV-InterSect Vehicular Data	RLV-InterSect Vehicular Data	$diff\_percentage\_thr$	
$Vehicle._1$	$Vehicle._1$	1.4%	$TrueMatching$
$Vehicle._2$	$Vehicle._9$	0.8%	$FasleMatching$
$Vehicle._{17}$	$Vehicle._{25}$	20.4%	$TrueNotMatching$
$Vehicle._{18}$	$Vehicle._{18}$	70.6%	$FalseNotMatching$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$Vehicle._n$	$Vehicle._m$	x.y%	tag

in this case, a pair is being marked as matching, only if, the $diff\_percentage$ between the vehicular data for its two vehicles is very small, even smaller than $diff\_percentage\_thr$. While, having large $diff\_percentage\_thr$ can be translated as relaxed matching criteria, because in this case, a pair is being marked as matching even if the $diff\_percentage$ between the vehicular data for its two vehicles is large, as long as the $diff\_percentage$ is less than the $diff\_percentage\_thr$.

Our goal is to find the $diff\_percentage\_thr$ that will maximize both $TruePositive$ and $TrueNegative$, and at the same time, minimize both $FalsePositive$ and $FalseNegative$. This optimization problem can be summarized as the following, find the $diff\_percentage\_thr$ that maximizes

$$\sum(Count(TruePositive) + Count(TrueNegative))$$

and at the same time minimizes

$$\sum(Count(FalsePositive) + Count(FalseNegative))$$

The online solution is not feasible because it requires classifying both the matching and the not matching pairs as either true or false. This classification requires knowing the vehicles unique ID, which is not possible in real cases as the vehicular data is broadcasted anonymously for privacy reasons. So we are going to solve the problem offline after forcing the vehicles to broadcast its unique ID as part of its vehicular data, then we will use the optimal $diff\_percentage\_thr$ determined offline as the typical

diff_percentage_thr for real situations, online. The optimal *diff_percentage_thr* determined offline is not the absolute optimal *diff_percentage_thr* because it depends on some other parameters *i.e.*, vehicular density, message sending rate, etc. If any of these parameters changes, the optimization problem should be resolved to find the corresponding optimal *diff_percentage_thr*. In CASCADE, the only parameter that will change in real situations is the vehicular density, so we should solve the problem for different vehicular densities and determine the corresponding optimal *diff_percentage_thr*. This table should be pre-loaded in each vehicle as part of the CASCADE software. At runtime the appropriate *diff_percentage_thr* should be used according to the surrounding vehicular density.

The following experiment shows how to compute the *diff_percentage_thr* for medium density, 66 vehicles/km.

IV.5.1 Finding the Optimal *diff_percentage_thr* Experimentally

The purpose of this experiment is to solve the optimization problem experimentally by finding the optimal difference percentage threshold, *diff_percentage_thr*, at certain vehicular density.

TABLE 5: Simulation Settings for Optimal *diff_percentage_thr* Experiment

transmission range	300 m
highway length	100 km
max distance traveled	10 km
vehicles generated	500
max speed	30 m/s
simulation runtime	360 seconds
high density	90 vehicles/km
medium density	66 vehicles/km
low density	53 vehicles/km

The same simulation settings as in Table 5[4] have been used for this experiment except that the vehicular density has been fixed to be medium density, 66 vehicles/km. So we set the max transmission range for all the vehicles participating in the experiments to 300 m [84]. The highway configuration is a four-lane divided highway of length 100 km. Also, we set the max speed for any vehicle to be 30 m/s, and they

---

[4]This table is a copy of Table 8.

enter the highway according to a Poisson distribution. The simulation is run for 360 seconds, resulting in a total of 500 vehicles generated. In the 360-second simulation runtime, the maximum distance traveled by any vehicle is 10 km. Moreover, the content of broadcasted frames, primary and aggregated frames, has been changed to include the unique vehicular ID. For primary frames, the broadcasting vehicle will add its unique ID as part of the information in the primary record. Also, compact records in the aggregated frames will include the vehicle unique ID. Including the vehicular unique ID in broadcasted frames is *only* performed in this experiment and does not happen in real situations, to protect the vehicles' privacy.

Algorithm 1 Finding the Optimal $diff\_percentage\_thr$

Require: Best Matching List BML
 1: **for** each record rec in BML **do**
 2:    **if** $rec.diff\_percentage < diff\_percentage\_thr$ **then**
 3:       $MatchCounter + +$
 4:       **if** $rec.vehicle_{LV}.ID == rec.vehicle_{RLV}.ID$ **then**
 5:          $TruePostiveCount + +$
 6:       **else**
 7:          $FalsePostiveCount + +$
 8:       **end if**
 9:    **else**
10:       $NotMatchCounter + +$
11:       **if** $rec.vehicle_{LV}.ID == rec.vehicle_{RLV}.ID$ **then**
12:          $FalseNegativeCount + +$
13:       **else**
14:          $TrueNegativeCount + +$
15:       **end if**
16:    **end if**
17: **end for**

In order to find the optimal $diff\_percentage\_thr$, the experiment consists of multiple runs, specifically 20 runs, and for each run a different $diff\_percentage\_thr$ value is used starting from 0% to 100% with a 5% increment. The following measures are recorded with each run, average $TruePositiveCount$, average $FalsePositiveCount$, average $TrueNegativeCount$ and average $FalseNegativeCount$. Each of these average measures is the overall average for this run for all vehicles. For example, at each vehicle level, the $FalsePositiveCount$ is calculated for each RLV frame received as shown in Algorithm 1. Then we average these values to calculate the average $FalsePositiveCount$ per vehicle, and then finally calculate the overall average $FalsePositiveCount$ by averaging the average $FalsePositiveCount$ for all vehicles.

The other metrics are calculated in a similar manner.

Figure 21 shows the experiment results when using the graph representation of the vehicular data. In the figure, the x-axis represents the $diff\_percentage\_thr$, while the y-axis represents the vehicles' percentages as they are classified into one of the four metrics, average $TruePositiveCount$, average $FalsePositiveCount$, average $TrueNegativeCount$ and average $FalseNegativeCount$.

The most conservative matching criteria is when the $diff\_percentage\_thr$ is 0%, because it will reject any vehicular pair, if the $diff\_percentage$ for this pair is greater than 0%, which requires the vehicular data for the two vehicles in this pair to be exactly matching. As we see in Figure 21, the average $TruePositiveCount$ percentage curve starts at 12%, representing the percentage of vehicles that exactly match when using a $diff\_percentage\_thr$ of 0%. Then the average $TruePositiveCount$ percentage keeps increasing as the $diff\_percentage\_thr$ increases, because the $diff\_percentage\_thr$ increase implies that we are relaxing the matching criteria, till it saturates at 68%. We can see the same behavior for the average $FalsePositiveCount$ percentage curve as it starts at 2% when the $diff\_percentage\_thr$ is 0% and keeps increasing till it saturates at 32%. While for the average $TrueNegativeCount$ percentage curve, it starts at 30% and keeps decreasing till it reaches 0%, and the same behavior applies to the average $FalseNegativeCount$ percentage curve that starts at 56% and also keeps decreasing till 0%. The relationship between these four counters was explained in detail in Section IV.4.

As we explained in Section IV.1, each experiment will be run twice using the two suggested data structures to represent the vehicular data to determine which of them is more suitable. The results for the previous experiment using the KD-Tree data structure are shown in Figure 22. All the curves in Figure 22 show behavior similar to their corresponding curves in Figure 21.

Determining the optimal $diff\_percentage\_thr$ using Figures 21 and 22 is not straight forward. So we grouped the two curves for the average $TruePositiveCount$ percentage and the average $TrueNegativeCount$ percentage together as they represent the maximization term and grouped the other two curves for the average $FalsePositiveCount$ percentage and the average $FalseNegativeCount$ percentage together as they represent the minimization term. And in Figures 47 and 48 we show a new curve that represents the difference between the maximization and minimization terms. Adding the difference curve makes finding the optimal
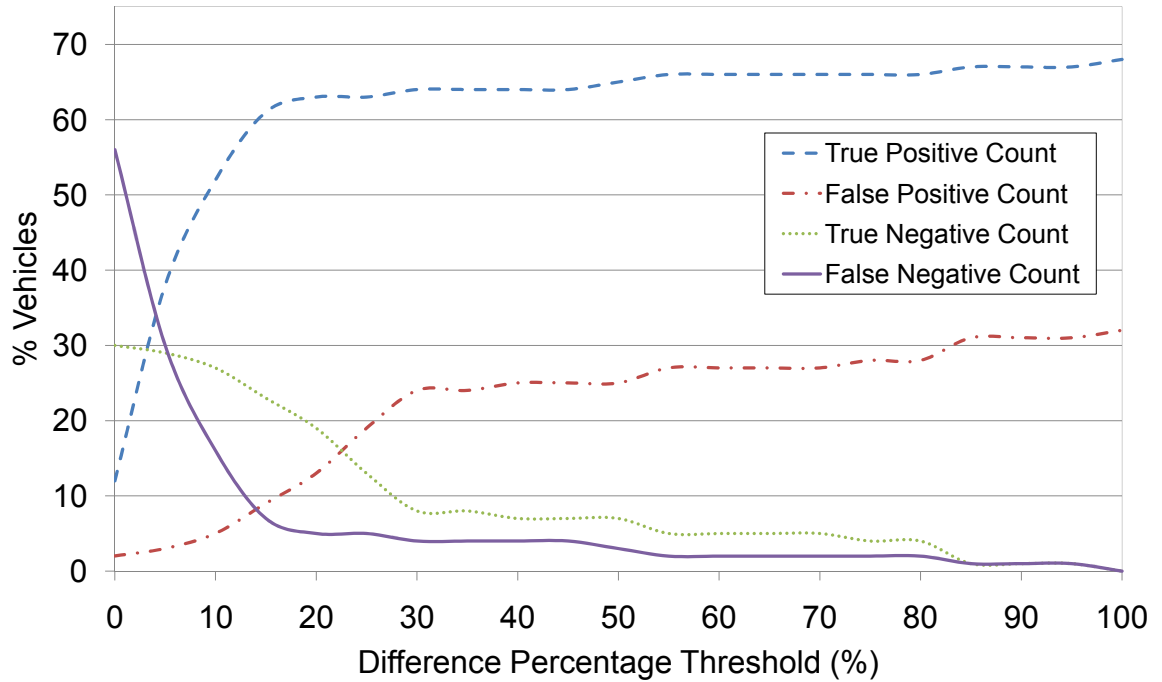
FIG. 21: Classifying the Received Vehicular Data into 4 Categories (*TruePositive*, *FalsePositive*, *TrueNegative* and *FalseNegative*) and the Corresponding Percentage of Each Category as the *diff_percentage_thr* Varies: Using Graph Representation and Medium Density

diff_percentage_thr obvious. Using the graph representation (Figure 23) the peak point in the difference curve occurred when the *diff_percentage_thr* is equal to 16%. And the optimal *diff_percentage_thr* using the KD-Tree representation (Figure 24) occurred when the *diff_percentage_thr* is equal to 16.5%.

The two data structures used to represent the vehicular data, the graph and the KD-Tree, produced very close optimal *diff_percentage_thr* values. This implies that using different representations for the vehicular data has no significant impact on the accuracy or correctness of the results. The only difference between these two representations was noticed on performance measures, such as memory consumption, CPU usage and turn around time. These differences are summarized in the following points:

- Turn around time : KD-Tree representation is 22% faster than graph representation
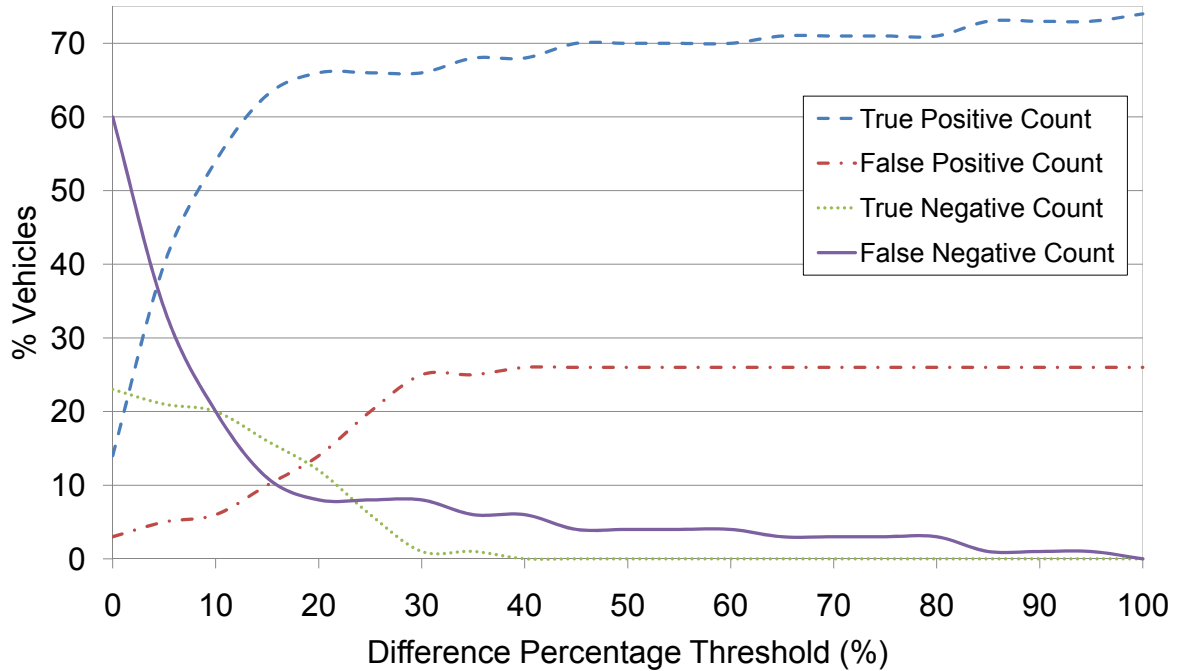
FIG. 22: Classifying the Received Vehicular Data into 4 Categories ($TruePositive$, $FalsePositive$, $TrueNegative$ and $FalseNegative$) and the Corresponding Percentage of Each Category as the $diff\_percentage\_thr$ Varies: Using KD-Tree Representation and Medium Density

- CPU Usage : KD-tree representation uses 36% less CPU than graph representation

- Memory Consumption : KD-tree representation uses 39% more memory than graph representation

Based on the previous results, we can conclude that any application that uses the KD-Tree representation will run faster, yet it will consume more memory, which will make this application dominate the system and leave less room for other applications to run simultaneously on the system. This behavior is acceptable only if the application is a critical one. Using the graph representation will make the application run a bit slower and consume less memory, which will leave more room for other applications to run simultaneously and share the system with it. This is acceptable behavior if that application falls in the non-critical applications category. So, the decision of which representation to use depends on how critical the current application using CASCADE framework is with respect to the other applications that are sharing the
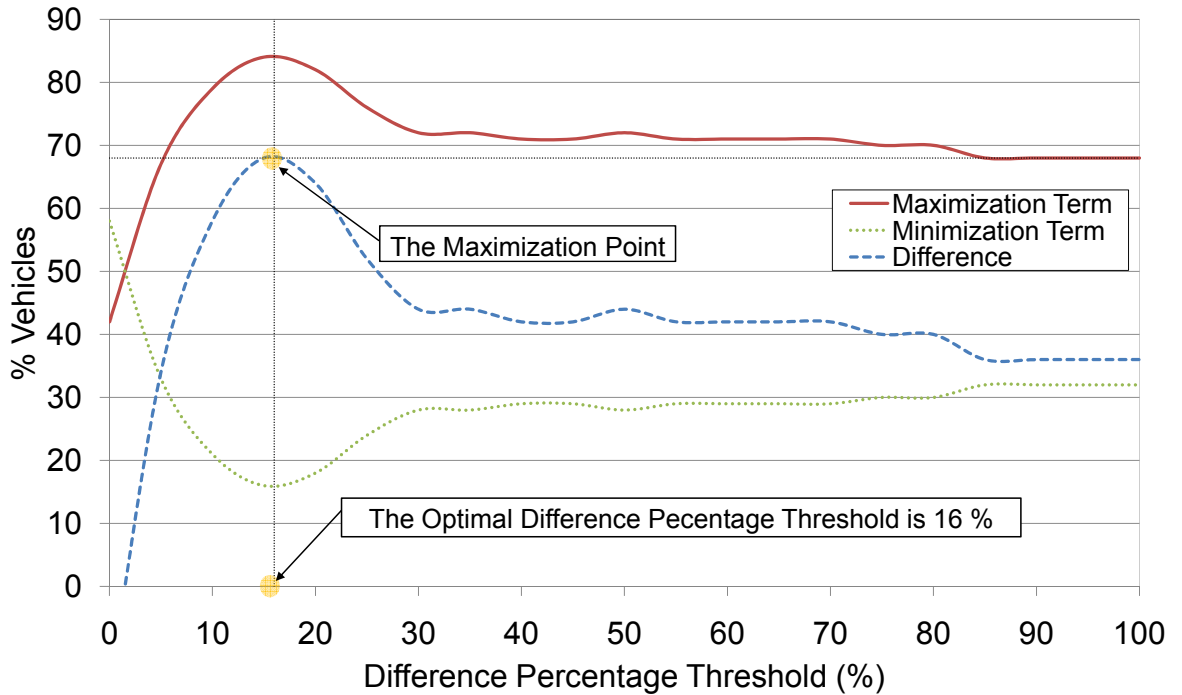
FIG. 23: Grouping the 4 Possible Received Vehicular Data Categories into Two Terms (Maximization Term and Minimization Term) and the Corresponding Percentage of Each Term as the $diff\_percentage\_thr$ Varies: Using Graph Representation and Medium Density

system.

To conclude, in case of medium vehicular density the optimal $diff\_percentage\_thr$ is 16%, considering the graph representation as the formal representation for the vehicular data in CASCADE. This means that if the $diff\_percentage$ between the vehicular data for any two vehicles is less than 16% they will be considered *matching* otherwise, they will be *not matching*.

IV.6 MATCHING PERCENTAGE THRESHOLD AT FRAME LEVEL

Until this point, we have been considering the difference percentage threshold ($diff\_percentage\_thr$) at the vehicle level. When comparing two vehicles' data, this threshold impacts the decision on whether to classify these two vehicles as matching or not. The decision on whether to accept the RLV frame and use its data to extend the local view or not is impacted by the matching percentage threshold, ($matching\_percentage\_thr$), which describes the percentage of matching vehicle pairs
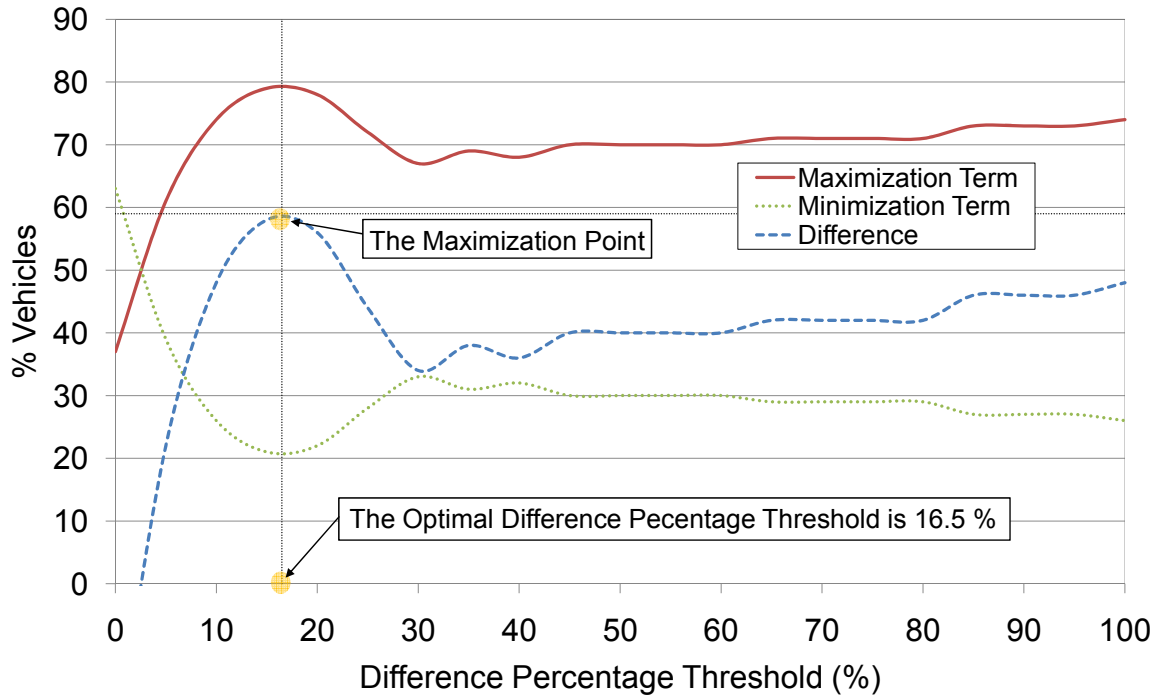
FIG. 24: Grouping the 4 Possible Received Vehicular Data Categories into Two Terms (Maximization Term and Minimization Term) and the Corresponding Percentage of Each Term as the $diff\_percentage\_thr$ Varies: Using KD-Tree Representation and Medium Density

in the intersecting sections of two frames, RLV-Intersect and LV-Intersect. The comparison result controls the decision on whether to accept the RLV frame and use its data to extend the local view or not.

In this section, we focus on finding the optimal $matching\_percentage\_thr$ at the frame comparison level. The matching percentage, $matching\_percentage$, between two intersecting sections may range from 0% to 100%. In order to accept the received frame, RLV frame, and use the data in the non-intersecting section to extend the local view, the $matching\_percentage$ should exceed certain threshold, $matching\_percentage\_thr$. If the $matching\_percentage$ did not exceed it, the RLV frame will be rejected. To be able to find the optimal $matching\_percentage\_thr$, we should first understand the impact of varying its value. Assuming that the optimal $diff\_percentage\_thr$ is being used at vehicle comparison level, this will guarantee the matching results to have the maximum true data, $TruePositive$ and $TrueNegative$ and minimum false data, $FalsePositive$ and $FalseNegative$. As a consequence, varying the $matching\_percentage\_thr$ will not impact the resulting extended view

correctness but it will definitely impact its accuracy[5]. Hence the search for the optimal *matching_percentage_thr* is not applicable because the definition of the optimal *matching_percentage_thr* depends on the required accuracy level by the application using CASCADE framework. For the applications that require highly accurate extended view, they should use high *matching_percentage_thr*. Because those kind of applications are somehow very selective with respect to the RLV frames that they will accept, their extended view length is not expected to be long. While for the other applications that can sacrifice having accurate extended view in order to get longer extended view, they can use lower *matching_percentage_thr* and hence accept more RLV frames and gain the benefit of having longer extended view.

As we have noticed, varying the *matching_percentage_thr* has a direct impact on the resulted extended view length, so in the following experiment we study the trade-off between these variables, *matching_percentage_thr* and extended view length.

IV.6.1 *matching_percentage_thr* and Extended View Length Trade-off

For this experiment we used the simulation settings in Table 5, but we set the vehicular density to medium. The graph data structure has been used to represent the vehicular data in RLV-Intersect and LV-Intersect, and the *diff_percentage_thr* has been set to its optimal value for the medium density case, which is 16%. The only variable will be the *matching_percentage_thr* which will vary from 0% to 100%. For each value, the corresponding average maximum extended view length will be recorded. When *matching_percentage_thr* is set to 0%, this means the RLV frames will be accepted even if there is no matching at all between the data in the RLV-Intersect and LV-Intersect, and the corresponding average maximum extended view length is expected to be the longest one possible. The average maximum extended view length is expected to be the shortest possible when *matching_percentage_thr* is set to 100%, because in this case the data in the RLV-Intersect should exactly match the data in the LV-Intersect.

In Figure 25, the x-axis represents the *matching_percentage_thr* percentage, while the y-axis represents the average maximum extended view length. We repeat the

---

[5]Using the optimal *matching_percentage_thr* will guarantee that the calculated *matching_percentage* is correct and hence any decision based on that value will be correct. But using a low *matching_percentage_thr* will accept frames with a low *matching_percentage*, resulting in a less accurate extended view. As *matching_percentage_thr* increases, CASCADE will only accept frames with high *matching_percentage* and hence build more accurate extended view.
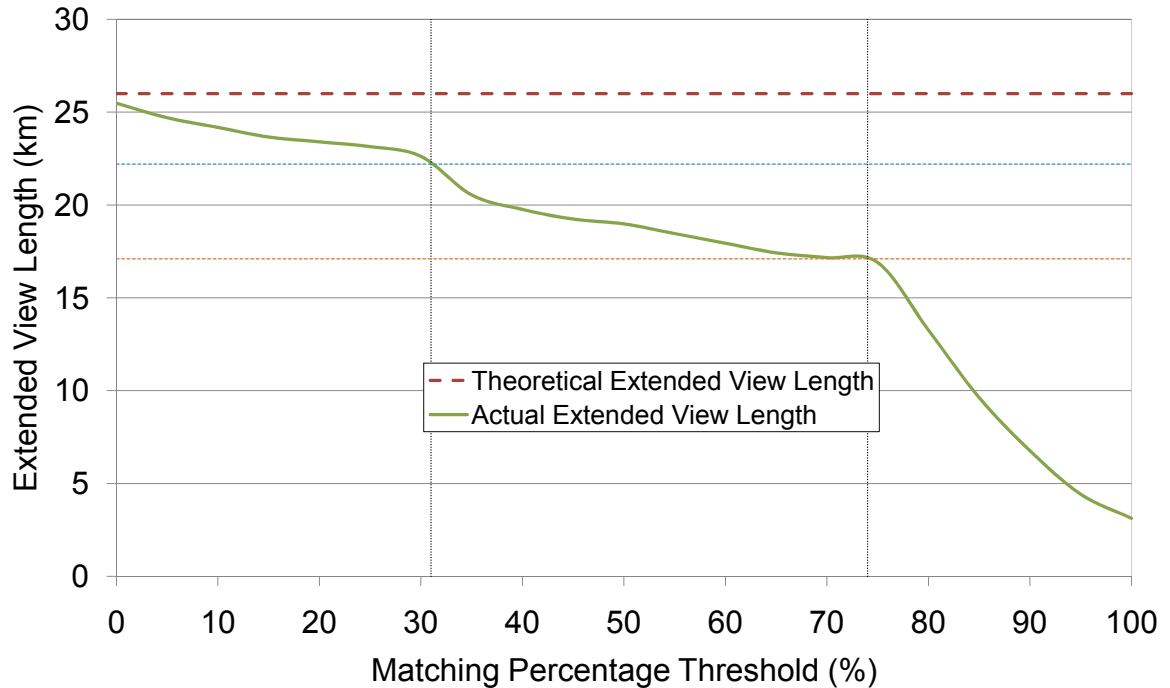
FIG. 25: Extended View Length as *matching_percentage_thr* Varies: Using Medium Density

experiment multiple times for the same *matching_percentage_thr*, and then we calculate the average for the maximum extended view length found for all runs with the same *matching_percentage_thr*. As shown in Figure 25, and as expected, the average maximum extended view length reaches its maximum, 25.48 km, when the *matching_percentage_thr* equals 0%. Although the curve for the average maximum extended view length comes very close to the theoretical limit for the maximum extended view length, 26 km, it never reaches this limit. The general trend for the curve is that as the *matching_percentage_thr* increases, the average maximum extended view length decreases. We can easily distinguish between three different sections of the average maximum extended view length curve.

In the first section of the curve, the average maximum extended view length starts at 25.48 km when the *matching_percentage_thr* is 0%, and it continues decreasing as the *matching_percentage_thr* increases. The curve decrease continues smoothly till the *matching_percentage_thr* reaches 31% where the corresponding average maximum extended view length is 22.2 km. After this point the average maximum extended view length drops a little bit which makes this point a good candidate for the

first section boundary.

The second section starts at 22.2 km right after the drop in the average maximum extended view length that occurred at the end of the first section. In this section, the average maximum extended view length continues decreasing but with faster rate than in the first section, till the *matching_percentage_thr* reaches 74% where the corresponding average maximum extended view length is 22.2 km. At this point another drop in the average maximum extended view length occurs which makes it the boundary for the second section.

The curve in the third section has the highest decrease rate among the other two sections. It starts with the *matching_percentage_thr* equal to 74% with a corresponding average maximum extended view length equal to 17.1 km and decreases sharply till it reaches 3.12 km when the *matching_percentage_thr* is 100%.

To conclude, as the *matching_percentage_thr* changes, the resulting average maximum extended view length changes too. The higher the *matching_percentage_thr*, the shorter the resulting average maximum extended view length. The average maximum extended view length drops gradually as the *matching_percentage_thr* increases, and the drop occurs in three distinguishable stages. The *matching_percentage_thr* at the boundary of each stage is the recommended value to be used by any application that will utilize CASCADE framework. For the applications that require highly accurate extended view, the recommended *matching_percentage_thr* is 74%, and this will result in average maximum extended view length of 17.1 km. For applications that require a less accurate extended view, the recommended *matching_percentage_thr* is 21%, which will result in an average maximum extended view length of 22.2 km.

IV.7   SUMMARY

In this chapter, we presented how each vehicle can extend its local view beyond the 1.5 km limit, which is the maximum length for the local view. All vehicles in the system share their local view data by broadcasting the most recent version frequently. Upon receiving the surrounding vehicles' local views (or the RLV frames), each vehicle reconstructs the original view in each of these RLV frames and determines the intersection between the received view and its own local view. Based on comparing the vehicular data in the intersecting section between these two frames, the vehicle determines to use the data in the received frame to extend its local view

or not. To process and compare the vehicular data in the intersecting views, we need to represent these vehicular data in a suitable data structure. Out of the possible data structures to represent the vehicular data, we selected the graph and KD-Tree data structures. We showed how to compare the vehicular data for two vehicles and find a measure, $diff\_percentage$, that shows how different these two vehicles are. By comparing the $diff\_percentage$ against $diff\_percentage\_thr$ and the vehicles IDs, we can classify the comparison state for these two vehicles into one of the following: $TruePositive$, $FalsePositive$, $TrueNegative$ and $FalseNegative$. Finding the optimal $diff\_percentage\_thr$ that will maximize the sum of $TruePositive$ and $TrueNegative$ and minimize the sum of $FalseNegative$ and $FalsePositive$ is an optimization problem that we solved experimentally for the case of medium vehicular density. The data structure used for representing the vehicular data introduced a small difference in the calculated optimal $diff\_percentage\_thr$, in case of using the graph data structure, the optimal $diff\_percentage\_thr$ is 16% and in case of using the KD-Tree data structure, the optimal $diff\_percentage\_thr$ is 16.5%. At the frame comparison level, the result of comparing the received view to the local view is represented as percentage, $matching\_percentage$, that shows how similar these two views are. Based on comparing the $matching\_percentage$ to $matching\_percentage\_thr$, the received view will be accepted or not. Hence, the $matching\_percentage\_thr$ will have an impact on the extended view length, which we studied.

CHAPTER V

DATA SECURITY COMPONENT

The goal of CASCADE is to allow a vehicle to obtain an accurate view of upcoming traffic conditions. Vehicles pass data about traffic conditions ahead to vehicles behind them so that they will have timely notification of upcoming traffic conditions. In this chapter, we look at how to add security to CASCADE framework. Recent work in VANET security has introduced the idea of using certified public keys (*i.e.*, PKI) stored in a tamper-proof device [25] or running on a tamper-proof service [14] to authenticate vehicles. While these features keep an attacker from obtaining false identities, they do not prevent a malicious user from tampering with other parts of the system, which provide the data that will be digitally signed by components in the tamper-proof device or service. We focus on additions to CASCADE that can detect and defend against false data whether malicious or the result of equipment malfunction. We analyze the possible sources for sending false position information, and we group them into four categories. We propose a technique for verifying vehicles' claimed locations by supplementing the received signal strength (RSSI) technique with a scanning laser rangefinder. We show that our defense technique can quickly detect false inputs given to CASCADE and will quickly quarantine those vehicles providing the false input.

In this chapter, we analyze possible false position information attacks and categorize them according to the data source of the attack. Also, we present a new position verification technique that supplements RSSI with a laser rangefinder to reduce the inaccuracy incurred in RSSI, which usually results in a high false positive rate. Without loss of generality, we applied the proposed technique to CASCADE in order to evaluate the technique's performance and measure its incurred overhead and, at the same time, to tighten CASCADE's security. The proposed technique proved to be fast in detecting any malicious vehicle in less than 30 seconds and detecting the GPS equipment malfunction in less than one minute even under low penetration rates. Also, it has a very low false positive rate making it highly accurate. Moreover, it is considered practical because the implementation cost is relatively low. Additionally, the incurred overhead resulting from applying the proposed technique to CASCADE under different vehicular densities was negligible.

V.1 POSITION VERIFICATION IN CASCADE

With the goal of improving security in CASCADE, we present our technique for verifying the locations provided by vehicles about themselves in primary frames. The position verification technique takes advantage of recent advances in the use of scanning laser rangefinders for vehicular safety applications. In this section, we describe our threat model, how false data is detected, and how misbehaving vehicles are quarantined to prevent false data from spreading.

V.1.1 Assumptions

We assume that each vehicle is equipped with a GPS receiver using DGPS [23] with an accuracy on the order of centimeters, a communications device using DSRC [11], a scanning laser rangefinder, and the CASCADE application. Scanning laser rangefinders, such as those produced by SICK [85] and ibeo [86], have been recently used for vehicular safety applications [87, 88] to detect and measure the distance to objects with high accuracy. The ibeo LUX, for instance, can complete 25 scans a second with a range of 3-200 m and a field of view (FOV) of 100°. To cover 360° of FOV, the laser rangefinder may be installed on a rotating base. If this adds too much delay, multiple (up to 4) laser rangefinders could be installed. This is feasible, especially as the cost of a single laser rangefinder is only about 100 EURO. Each vehicle is also pre-assigned multiple public/private key pairs and the public keys' certificates, which is a common assumption in VANET research [24, 25]. At any certain point of time, each vehicle will use only one key pair for authentication, and this primary key will be used as the vehicle's ID. However, to address privacy concerns, it will be able to change the current active key pair and switch to use another one from the pre-assigned public/private key pairs. The vehicle will be allowed to switch keys only if it is not recently identified as a suspect vehicle.

In addition to using the scanning laser rangefinder to verify position, we also will use RSSI (Radio-based Signal Strength Indicator) provided by IEEE 802.11. With RSSI, a receiver can estimate the transmitter location using known mathematical models for the wireless channel. The estimated locations using RSSI are not highly accurate since there is a large variation in signal attenuation in different environments [89]. Despite some inaccuracies in location estimation, RSSI is widely used due to its low cost as it requires no special hardware [90, 91]. Because of its inaccuracies,

we use RSSI as only one part of our position verification technique.

We assume that since the system equipment (GPS, transceiver, and laser rangefinder) can be located anywhere in the vehicle, they are not tamper-proof. The only components that are tamper-proof are the CASCADE application and the new detection and quarantine modules, residing inside the tamper-proof device. Because of this, we only consider verifying the position of vehicles provided in primary frames. The aggregation component of CASCADE will be located inside the tamper-proof device and thus can be trusted.

V.1.2 Threat Model

Here we describe the types of attacks that could be perpetrated against the system. Since we assume that the CASCADE application itself is secure inside a tamper-proof device, attacks can only come from tampering with outgoing frames, tampering with received frames, or providing false input. Tampering with outgoing frames will be easily detected because the vehicle's private key is not available outside of the tamper-proof device. Any change to the contents of outgoing messages will be detected when the receiver checks the digital signature on the message. This also eliminates the Sybil attack, where one vehicle impersonates multiple non-existent vehicles.

The only way a malicious user could tamper with received frames without detection is by dropping the frame before sending it to the CASCADE application. We will address this issue when we discuss how malicious vehicles are quarantined (Section V.1.3).

The remaining attacks come from providing false position information to the CASCADE application. We classify these attacks into four categories according to the source of the false information:

- *Unsynchronized Trace:* The attacker could record a trace of position information during a previous trip on the same road and then substitute these recorded positions for the vehicle's actual current GPS coordinates. An unsynchronized trace is one where the positions being sent to CASCADE are outside the nominal transmission range from a potential receiver.

- *Synchronized Trace:* This attack is similar to the unsynchronized trace, but the claimed position is within the nominal transmission range of a potential receiver.

- *Primary Frame Replay:* The attacker could record the position information from recently received primary frames from other vehicles and then substitute these positions for its actual GPS coordinates. The difference between this attack and the previous two is that the attacker will be sure of having a vehicle in the claimed position which will make detection more difficult.

- *Malfunctioning GPS:* If the GPS device is malfunctioning, it will not be able to translate the received signal into the correct coordinates and will report incorrect location coordinates to CASCADE. Actually, this is not an intentional attack, but it has the same effect.

V.1.3 Defense Technique

Here we present how CASCADE will defend itself against the attacks described above. The proposed defense system consists of two main modules, the *detection module* and the *quarantine module.* The detection module is responsible for checking the consistency of received primary frames. If the detection module determines that the information in a primary frame is false, it will place the sending vehicle in the *suspect list* and trigger the quarantine module. The quarantine module in a vehicle detecting an attack is responsible for issuing a quarantine proposal and a quarantine request. Upon receiving certain number of quarantine requests, the quarantine module in a suspected vehicle is responsible for temporarily suspending the CASCADE application from sending any frames.

Detection Module

The detection module verifies that the location claimed in a received primary frame is consistent with the estimated location of the sending vehicle. Note that re-broadcasted primary frames (where the sender is not the original source) bypass this consistency check. The consistency check is performed in two levels: first using the signal strength (RSSI), and then using the laser rangefinder, if needed. The process of the detection module is described in Algorithm 2.

In the first-level consistency check, the vehicle's claimed location *rcvd_loc* is extracted from the received primary frame, and the location of the sending vehicle *rssi_loc* is estimated using RSSI. Also, the distance from the receiver to *rcvd_loc*, *rcvd_dist*, is compared with the nominal transmission range *xmit_rng*. The next

steps depend upon how the difference $rssi\_diff$, which is the difference between $rcvd\_loc$ and $rssi\_loc$, compares to the RSSI accuracy threshold $rssi\_thr$[1]:

- $rssi\_diff < rssi\_thr$

  The two locations are considered to be consistent, and the received primary frame is added to the local view and processed normally by CASCADE.

- $rssi\_diff > rssi\_thr$ && $rcvd\_dist < xmit\_rng$

  The vehicle's ID is placed in the suspect list, and the second-level consistency check is triggered.

- $rssi\_diff > rssi\_thr$ && $rcvd\_dist > xmit\_rng$

  The second-level consistency check is bypassed, the vehicle's ID is placed in the suspect list, and the quarantine module is triggered.

In most cases, primary frames failing the first-level consistency check will be detected by multiple vehicles - all those that received the primary frame, as well. Each of these vehicles will check to see if the suspected vehicle claims to be a direct neighbor[2], using $rcvd\_loc$ and their local view. If the suspected vehicle is a direct neighbor, the detecting vehicle will begin the second-level consistency check, otherwise it will wait for notification from a direct neighbor of the suspect.

In the second-level consistency check, a direct neighboring vehicle of the suspect will use the readings from its scanning laser rangefinder to determine if there is a vehicle near $rcvd\_loc$. If there is a vehicle present, the location will be measured by the laser rangefinder as $laser\_loc$. The local view is checked to see if there is a different vehicle that claims to be at $rcvd\_loc$. If so, the vehicle is marked as *conflict*, otherwise it is marked as *unique*. As with the first-level check, the next actions depend upon how the difference, $laser\_diff$, between $rcvd\_loc$ and $laser\_loc$ compares to the laser threshold $laser\_thr$[3]:

- $laser\_diff < laser\_thr$ && *unique*

  The vehicle is removed from the suspect list, added to the local view, and the frame is processed normally by CASCADE.

---

[1]The value of the $rssi\_thr$ depends on the accuracy of the RSSI technique, in the range of meters.

[2]Vehicle x is considered a direct neighbor of vehicle y, if there is no vehicle between them, and x and y are within the laser rangefinder FOV of each other.

[3]The value of the $laser\_thr$ depends on the accuracy of the accuracy of the laser rangefinder, in the range of centimeters.

- $laser\_diff < laser\_thr$ && $conflict$

  The quarantine module is triggered.

- $laser\_diff > laser\_thr$

  The quarantine module is triggered.

We note that a malicious user could falsify the information coming from the laser rangefinder. The false information could be used to either (1) deny the existence of an actual vehicle, or (2) confirm the existence of a false vehicle. To address the first case, we require that a majority of the neighboring vehicles be involved in quarantining a suspected vehicle; a single vehicle cannot achieve this. In the second case, this vehicle will not send a quarantine proposal, but other direct neighboring vehicles likely will, and the vehicle will still be detected.

Algorithm 2 Detection Module

Require: receiving a primary frame(pf)
1: **if** $pf.sender\_loc = pf.originator\_loc$ **then** {first level consistency check}
2:     $rcvd\_loc \leftarrow$ extract_rcvd_loc(pf);
3:     $rssi\_loc \leftarrow$ estimate_rssi_loc(pf.rcvd_signal_power);
4:     $rssi\_diff \leftarrow |rssi\_loc - rcvd\_loc|$;
5:     $rcvd\_dist \leftarrow |rcvd\_loc - current\_loc|$;
6:     **if** $rssi\_diff > rssi\_thr$ **then**
7:         add $pf.veh\_id$ to suspect list;
8:         **if** $rcvd\_dist > xmit\_rng$ **then**
9:             call Quarantine_Module;
10:         **else**
11:             $isdirect\_neighbor \leftarrow$ chk_localview($rcvd\_loc$);
12:             **if** $isdirect\_neighbor$ **then** {second level consistency check}
13:                 $isdetected \leftarrow$ laser_detector($rssi\_loc$);
14:                 **if** $isdetcted$ **then**
15:                     $laser\_loc \leftarrow$ estimate_laser_loc($rssi\_loc$);
16:                     $laser\_diff \leftarrow |laser\_loc - rcvd\_loc|$;
17:                     $isclaimed \leftarrow$ chk_localview($rssi\_loc$);
18:                     **if** $laser\_diff < laser\_thr$ **then**
19:                         **if** $isclaimed$ **then**
20:                             call Quarantine_Module; {$conflict$}
21:                         **else** {innocent}
22:                             Forward pf to CASCADE; {$unique$}
23:                         **end if**
24:                     **else**
25:                         call Quarantine_Module;
26:                     **end if**
27:                 **else**
28:                     call Quarantine_Module;
29:                 **end if**
30:             **else** {not direct neighbor}
31:                 do nothing;
32:             **end if**
33:         **end if**
34:     **else**
35:         Forward pf to CASCADE;
36:     **end if**
37: **else**
38:     Forward pf to CASCADE;
39: **end if**

Timestamp	Suspected Vehicle ID	Signature	Certificate

FIG. 26: Quarantine Proposal Frame

The Quarantine Module

The operation of the quarantine module differs based on the current role of the vehicle. If a vehicle is a direct neighbor of a suspected vehicle, the quarantine module is responsible for sending quarantine proposals and quarantine requests. If a vehicle is a suspect, its quarantine module is responsible for suspending transmissions from the CASCADE application.

For direct neighboring vehicles, the quarantine module is triggered when a consistency check fails. When this occurs, the vehicle will broadcast a *quarantine proposal* (Figure 26). The suspected vehicle may have more than one direct neighbor, which means that there may be more than one quarantine proposal sent at the same time. To avoid this, once a direct neighbor detects a lying vehicle, it will start a random countdown timer. If the timer expires and no other quarantine proposal about this suspect has been sent, the vehicle will send its proposal. This process is described in Algorithm 3.

The quarantine proposal is meant for vehicles other than the suspect. Upon receiving a quarantine proposal (Algorithm 4), a vehicle will check its suspect list for the vehicle ID contained in the proposal. If the vehicle is found in the suspect list, this vehicle will broadcast a *quarantine request* (Figure 27), meant for the suspect vehicle. Once the request has been sent, the suspect vehicle will be removed from the requesting vehicle's suspect list. This prevents multiple quarantine requests from being generated by the same vehicle upon receiving multiple quarantine proposals for the same suspect, which may only happen with low probability[4].

Upon receiving a quarantine request containing its ID (Algorithm 5), a suspected vehicle will initiate its pre-quarantine mode. In this mode, the suspect will initialize its quarantine request counter $Q\_req\_counter$ to 1 and count the number of vehicles N within its transmission range (this information is already contained in the vehicle's local view). Each time a new quarantine request is received, $Q\_req\_counter$ is

---

[4]This case may happen only if two direct neighbors of a suspect have their timers expire simultaneously. Given that the timer value is selected randomly, this will happen with low probability.

Timestamp	Suspected Vehicle ID	Original Quarantine Proposal	Signature	Certificate

FIG. 27: Quarantine Request Frame

incremented. The suspect vehicle moves from pre-quarantine mode to quarantine mode when $Q\_req\_counter > 0.5N$ and $Q\_req\_counter > 2$. This indicates that over 50% of the surrounding vehicles (and more than just two vehicles) have classified the vehicle as a suspect.

When the CASCADE application is started (which is assumed to be each time the vehicle is turned on), the number of times a vehicle has experienced a quarantine, Q_count, is set to 0. Each time quarantine mode is entered, Q_count is incremented. Once in quarantine mode, the suspected vehicle starts a quarantine timer that is set to expire in $60 * 2^{Q\text{-}count-1}$ seconds (*i.e.*, exponential increase). The vehicle will be blocked from sending messages until the timer expires.

In order to avoid receiving quarantine requests, a malicious user might try to disable the receiving capability in its transceiver or discard all quarantine requests without sending them to the CASCADE application. Because CASCADE will never receive a quarantine request, the sending of frames will not be blocked, so bandwidth will be wasted with this vehicle sending potentially false primary frames. To counter this, CASCADE will periodically initiate a self-test to ensure that the transceiver is working properly. In the self-test, a dummy message will be sent out. This message consists of a quarantine request with a random nonce as the vehicle ID (not matching any vehicle in the local view). If a quarantine message with the same nonce is not received and delivered to the CASCADE application, the vehicle will enter quarantine mode immediately.

V.2 SUMMARY

In this chapter, we presented a light-weight position verification technique for VANETs. Our technique used RSSI and laser range finders for position verification. We have used the position verification technique to tighten CASCADE security. The evaluation of this technique and its incurred overhead when applied to a particular VANET application (CASCADE), will be described in Chapter VII.

Algorithm 3 Quarantine Module: Sending a Quarantine Proposal

Require: receiving a primary frame (pf) from Detection Module
 1: $timer \leftarrow random()$;{start a timer}
 2: **while** $timer > 0$ **do**
 3:     wait for one second;
 4:     $timer - -$;
 5: **end while**
 6: $heard \leftarrow$ chk_rcvd_Q_proposal($pf.veh\_id$);{check if someone else send the same quarantine proposal}
 7: **if** $not\ heard$ **then**
 8:     $Q\_proposal \leftarrow$ form_Q_proposal($pf.veh\_id$);{form a quarantine proposal}
 9:     send($Q\_proposal$);[send the quarantine proposal]
10:     remove $pf.veh\_id$ from suspect list;
11: **end if**

Algorithm 4 Quarantine Module: Sending a Quarantine Request

Require: receiving a quarantine proposal (Q_proposal)
 1: $found \leftarrow chk\_suspect\_list(Q\_proposal.veh\_id)$;
 2: **if** $found$ **then**
 3:     $Q\_rqst \leftarrow$ form_Q_rqst($Q\_proposal.veh\_id$);{form a quarantine request}
 4:     send(Q_rqst);[send the quarantine request]
 5:     remove $pf.veh\_id$ from suspect list;
 6: **else**
 7:     drop Q_proposal;
 8: **end if**

Algorithm 5 Quarantine Module: The Quarantine Process Sequence

Require: receiving a quarantine request (Q_rqst)
1: **if** $Q\_rqst.veh\_id = veh\_id$ **then** {start the pre-quarantine mode}
2:    $timer \leftarrow$ random();{start a timer}
3:    $density \leftarrow estimate\_denisty(trans\_range)$;{estimate the density within the transmission range}
4:    $Q\_rqst\_counter \leftarrow 1$;
5:    $ismalicious \leftarrow false$;
6:    **while** $timer > 0$ and not $ismalicious$ **do**
7:       wait for one second and store all Q_rqsts in Q_rqst_queue;
8:       $timer - -$;
9:       **while** Q_rqst_queue is not empty **do**
10:          $new\_rqst\_rcvd \leftarrow$ chk_rcvd_Q_rqst($Q\_rqst.veh\_id$);{get a new quarantine request from the quarantine request queue}
11:          **if** new_rqst_rcvd **then**
12:             $Q\_rqst\_counter + +$;
13:          **end if**
14:       **end while**
15:       **if** $\frac{Q\_rqst\_counter}{density} > 0.5$ **then**
16:          $ismalicious \leftarrow true$;
17:       **end if**
18:    **end while**
19:    **if** $ismalicious$ **then** {start the quarantine mode}
20:       $Q\_count + +$
21:       $Q\_timer \leftarrow 60 * 2^{Q\_count-1}$;
22:       $block\_sending()$;
23:       **while** $Q\_timer > 0$ **do**
24:          wait for one second;
25:          $Q\_timer - -$;
26:       **end while**
27:       $resume\_sending()$;
28:    **end if**
29: **else**
30:    drop Q_rqst;
31: **end if**

CHAPTER VI

DATA DISSEMINATION COMPONENT

In this chapter, we address two important problems in VANETs; the first is how to disseminate vehicular data efficiently, and the second is how to handle traffic discontinuity. The solutions that we provide for these two problems are presented in the context of CASCADE framework, but they are loosely coupled with CASCADE internals so that they can be generalized easily.

In Chapter I, we discussed the broadcast storm problem that can occur whenever the blind message broadcast or message flooding is used in disseminating the vehicular application messages. In safety related applications, the common data dissemination technique is broadcasting, which makes this category of vehicular applications more vulnerable to the broadcast storm problem, especially in dense VANETs. Recently, many protocols [29, 54, 56, 57] have been proposed to alleviate this problem. Most of these protocols are based on selecting the farthest vehicle(s) within transmission range of the original sender to be the re-broadcaster(s). The selection
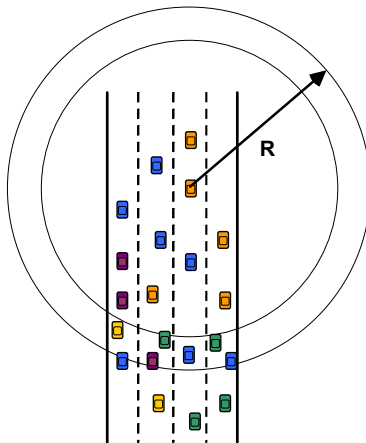
FIG. 28: Multiple Vehicles at the Boundary Can Cause a Spatial Broadcast Storm

is achieved by assigning increasing re-broadcast probabilities or decreasing waiting times as the distance from the sending vehicle increases. We show an example in Figure 28. Based on these criteria, all the vehicles in the highlighted area will have almost the same re-broadcast probability or waiting time, so they will re-broadcast the frame at the same time. This will cause a broadcast storm locally in the highlighted area. Although the generated storm is local, it will affect the overall system

performance. We term this the *spatial broadcast storm problem.*

In CASCADE, each vehicle depends on the primary and aggregated frames received from the vehicles ahead in building and maintaining its local and extended views. As long as the distance between each vehicle and the vehicles ahead is less than or equal to $Effective\_Transmission\_Range$ (250 m), it will be able to receive a continuous stream of frames update its views. If the distance exceeds 250 m, the stream will cease and hence a discontinuity problem will occur. In this chapter, we present a solution to handle and recover from discontinuity problem using the vehicles traveling in the opposite direction.

In Section VI.1, we study the impact of using Inter-Vehicle Geocast (IVG) [29] in case of dense VANETs and find that it will result in a spatial broadcast storm as shown in Figure 28. This work is the first to illustrate the existence of the spatial broadcast storm problem. To alleviate this problem we propose and develop an extension to IVG, *probabilistic IVG (p-IVG)*, to make the re-broadcast decision probabilistic based on awareness of the local topology, *i.e.*, knowledge of surrounding vehicle density, instead of being deterministic. The local topology awareness is acquired through a light-weight topology sensing utility. We will show in Chapter VII that using p-IVG results in improved reception rates, lower channel contention, less redundancy, more coverage, and most importantly, faster dissemination to distant vehicles. Section VI.2 presents the traffic discontinuity problem and how it can be handled in CASCADE using the proposed solution, the On-Demand Vehicular Gap-Bridging (OD-V-GB) technique.

VI.1   PROBABILISTIC INTER-VEHICLE GEOCAST (P-IVG)

In this section, we present probabilistic-IVG (p-IVG), a light-weight extension to Inter-Vehicle Geocast (IVG) [29] for broadcasting messages in dense VANETs. The probability function in p-IVG is based on the density of surrounding vehicles. p-IVG adapts itself according to the current traffic conditions, in order to minimize the number of vehicles that will re-broadcast the received frame and, at the same time, minimize the probability of failing to re-broadcast the received frame. The surrounding vehicle density is detected through a light-weight local topology sensing utility.

In IVG, each vehicle starts a timer for each frame it receives. While the timer is still on, these vehicles will keep listening to the media and if they detect that some

other vehicle broadcasts the same frame, they will cancel their own broadcast for that frame and turn off the timer. If the timer expires and the frame associated with this timer has not been re-broadcast by any other vehicle within transmission range, the vehicle re-broadcasts the frame. The timer value T_x for vehicle x is determined by Equation 19, where R is the transmission range and D_{sx} is the distance between vehicle x and vehicle s, the sender of the frame. The authors of IVG suggest using $\epsilon = 2$ to generate a timer value between $[0, T_{max}]$, where $T_{max} = 200\,\text{ms}$ [29].

$$T_x = T_{max} \cdot \frac{(R^\epsilon - D_{sx}^\epsilon)}{R^\epsilon} \tag{19}$$

By using IVG with its original settings, nodes that are close to each other at the boundary will have very similar, if not equal, timer values. This means that the nodes' timers will expire at almost the same time, and the nodes will re-broadcast the frame essentially simultaneously, resulting in collisions and thus, a spatial broadcast storm (Figure 28). The impact of using IVG on the system performance is presented in Section VII.4.

To alleviate this problem, we modified IVG by making the re-broadcasting of frames probabilistic based on the density of surrounding vehicles. In our *probabilistic-IVG* (*p*-IVG), when a vehicle receives a frame, it first selects a random number, x, in $[0, 1]$. If x is less than $\frac{1}{density}$, the timer is started. If not, then the frame will not be re-broadcasted by this vehicle. As the density increases, the percentage of nodes that will start their timers decreases.

VI.1.1 Analysis of *p*-IVG

Here we present a formal analysis of the p-IVG algorithm.

Notation and Assumptions

First, we introduce our assumptions and notation. We assume that vehicles are uniformly distributed on the road.

Let S be a "Source" vehicle. The uniform distribution assumption implies that the vehicles within S's transmission range are uniformly distributed in a disk of radius R centered at S.

Let D be the random variable that denotes the distance from a generic vehicle to S. Clearly $D \in U(0, R)$ and so

$$P[\{D \le d\}] = \frac{d}{R} \tag{20}$$

Let T be the value of the timer associated by IVG with vehicle with distance D.

$$T = T_{max}[1 - (\frac{D}{R})^{\epsilon}], where \quad \epsilon > 0 \tag{21}$$

Calculating the Probability Distribution of T

In this section, we derive the probability distribution for T. Let $t \in [0, T_{max}]$

$$
\begin{aligned}
P[\{T \le t\}] &= P[\{T_{max}(1 - \frac{D}{R})^{\epsilon} \le t\}] \\
&= P[\{(1 - \frac{D}{R})^{\epsilon} \le \frac{t}{T_{max}}\}] \\
&= P[\{1 - \frac{D}{R} \le (\frac{t}{T_{max}})^{\frac{1}{\epsilon}}\}] \\
&= P[\{\frac{D}{R} \ge 1 - (\frac{t}{T_{max}})^{\frac{1}{\epsilon}}\}] \\
&= P[\{D \ge R[1 - (\frac{t}{T_{max}})^{\frac{1}{\epsilon}}]\}] \\
&= 1 - P[\{D < R[1 - (\frac{t}{T_{max}})^{\frac{1}{\epsilon}}]\}] \\
&= 1 - \frac{R}{R}[1 - (\frac{t}{T_{max}})^{\frac{1}{\epsilon}}] \\
&= 1 - [1 - (\frac{t}{T_{max}})^{\frac{1}{\epsilon}}] \\
&= (\frac{t}{T_{max}})^{\frac{1}{\epsilon}} \tag{22}
\end{aligned}
$$

Equation 22 implies that for $\epsilon = 1$, T is uniformly distributed in $[0, T_{max}]$.

$$P[\{T \le t\}] = 1 - (1 - \frac{t}{T_{max}}) = \frac{t}{T_{max}} \tag{23}$$

For $\epsilon \ne 1$ the distribution of T is not uniform

Accurate Density Estimation

Each vehicle can determine the density, ρ, of vehicles around it. Obviously, the most natural set of candidates resides in the shaded area in Figure 29 determined by the intersection of a wedge subtended by an angle θ centered at S in the two concentric circles of radii $R - \delta$ and R, where δ will be estimated precisely later. The area of shaded area is

$$\frac{\theta}{2}[R^2 - (R - \delta)^2] = \frac{\theta}{2}\delta(2R - \delta) \tag{24}$$
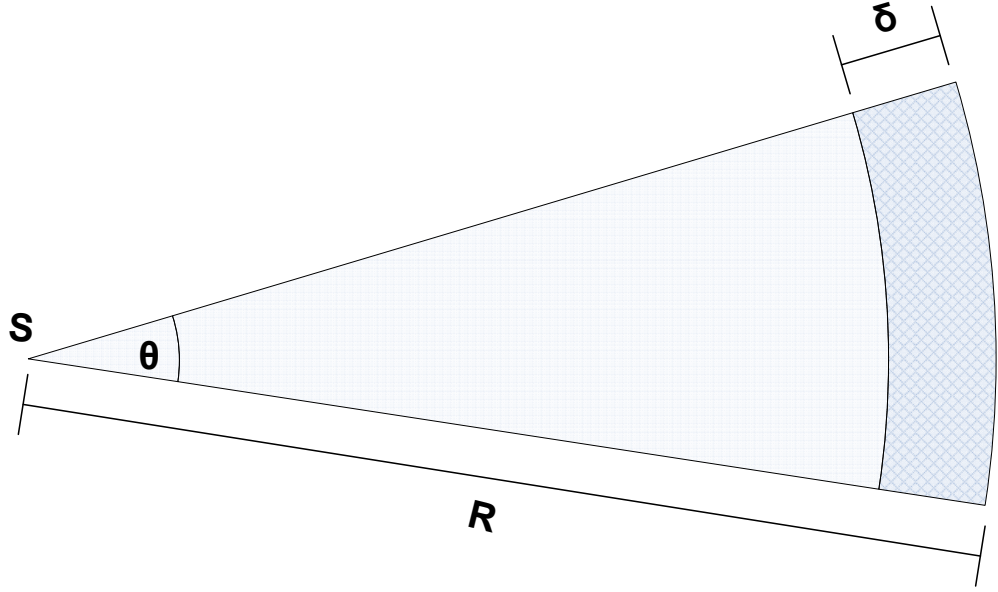
FIG. 29: Estimating the Vehicular Density

Using the density, ρ, the expected number of rebroadcast candidates in the hashed area in Figure 29 is

$$N = \rho\frac{\theta}{2}\delta(2R - \delta) \tag{25}$$

Observe that each vehicle in the shaded area can determine θ, ρ and R, assuming that it also can determine δ, it can compute N. Now, each vehicle in the shaded area can use $\frac{1}{N}$ instead of $\frac{1}{density}$ in deciding whether to rebroadcast or not.

Electing a Rebroadcast Leader

The probability of electing a rebroadcast leader in one election (round) is

$$p = \binom{N}{1}(\frac{1}{N})(1 - \frac{1}{N})^{N-1} = (1 - \frac{1}{N})^{N-1} \tag{26}$$

Let L be the random variable that keeps track of the number of rounds required to elect a leader.

$$\begin{aligned} P[\{L = K\}] &= (1-p)^{k-1}p \\ &= (1-\frac{1}{N})^{N-1}[1-(1-\frac{1}{N})^{N-1}]^{k-1} \end{aligned} \tag{27}$$

Notes: Recall that $\forall N$

$$(1-\frac{1}{N})^N < \frac{1}{e} < (1-\frac{1}{N})^{N-1} \tag{28}$$

This implies that $p \approx \frac{1}{e} \approx 36\%$. We can tabulate the probability of the election L=2, L=3, etc.

VI.1.2 Calculating δ
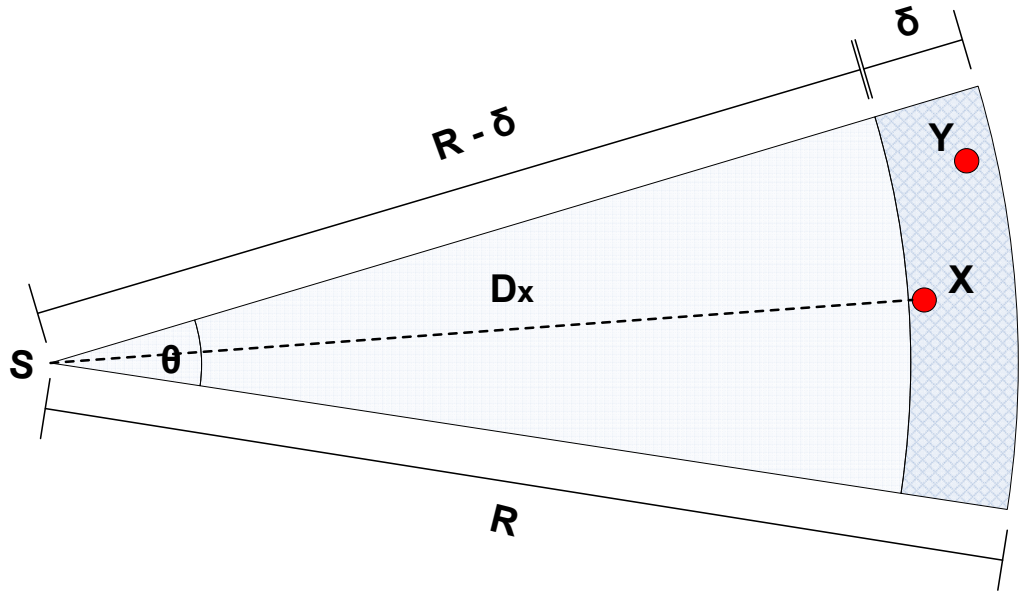
FIG. 30: Calculating δ

Let us partition T_{max} into $n, (n > 0)$ time quantum, each of size $\frac{T_{max}}{n}$. Assume that we have two vehicles x,y in the shaded area if

$$0 < T_y \leq T_x < \frac{T_{max}}{n} \tag{29}$$

The insight here is that both T_x and T_y "hash" for the same number and so they will, if allowed, transmit at the same time. Now, Equation 29 can be rewritten as

$$0 < T_{max}[1 - (\frac{D_y}{R})^\epsilon] \leq T_{max}[1 - (\frac{D_x}{R})^\epsilon] < \frac{T_{max}}{n}$$
$$0 < 1 - (\frac{D_y}{R})^\epsilon \leq 1 - (\frac{D_x}{R})^\epsilon \leq \frac{1}{n}$$
$$1 - \frac{1}{n} < (\frac{D_y}{R})^\epsilon \leq (\frac{D_x}{R})^\epsilon \leq 1$$
$$(1 - \frac{1}{n})^{\frac{1}{\epsilon}} < \frac{D_y}{R}) \leq \frac{D_x}{R} \leq 1$$
$$R(1 - \frac{1}{n})^{\frac{1}{\epsilon}} < D_y \leq D_x \leq R \qquad (30)$$

Now, we can compute δ

$$\begin{aligned} \delta &= sup(D_y - D_x), where \quad x < y \\ &= R - R(1 - \frac{1}{n})^{\frac{1}{\epsilon}} \\ &= R[1 - (1 - \frac{1}{n})^{\frac{1}{\epsilon}}] \end{aligned} \qquad (31)$$

We can conclude that first, n is technology dependant and known to all vehicles. So is, too, R and ϵ. In other words, every vehicle can calculate δ and, refereing back to Equation 25, the value of N, the number of potential candidates.

VI.1.3 Effect of ϵ

ϵ is a critical parameter in the timer value equation of IVG. The authors of IVG provided no evaluation of how it affects the distribution of timer values. Here, we study this parameter carefully and investigate its effect on the whole system. Then for p-IVG, we tune this parameter in order to achieve maximum performance.

One of the main goals of the timer value equation is to reduce the waiting time, T_x, before frames are re-broadcast, and thus, the overall delay to send the frame to distant areas. In order to achieve this, smaller timer values should be generated as the distance from the original frame sender (D_{sx}) increases.

Changing ϵ has a great impact on the generated timer values. $\epsilon > 1$ generates a family of convex curves for the timer values, $\epsilon < 1$ generates a family of concave curves, and $\epsilon = 1$ generates a linear curve. Generally, the convex curve produces larger and sparser values than the concave curve, resulting in an increased time for frames to travel longer distances. Figure 31 shows how changing ϵ affects the frame
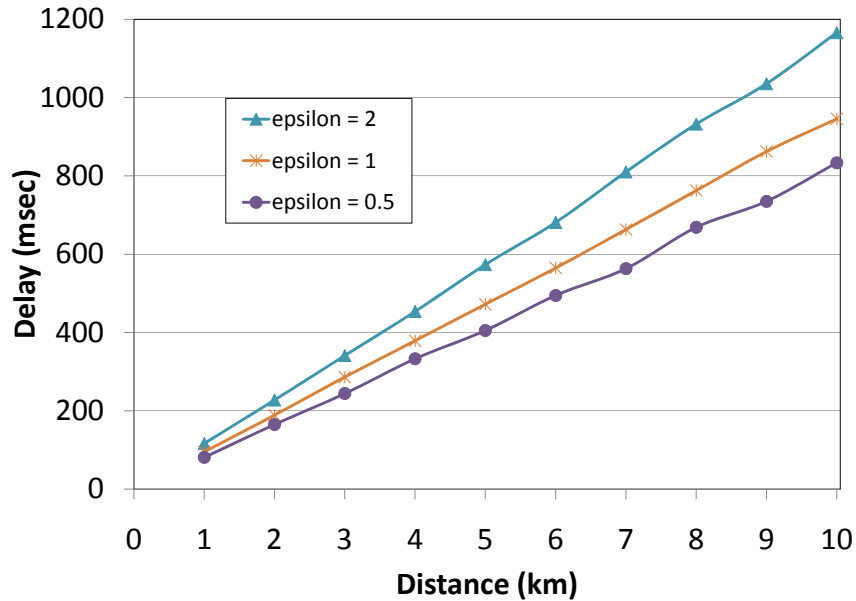
FIG. 31: Effect of ϵ on Frame Propagation Delay Using p-IVG

propagation delay for p-IVG. As ϵ decreases, the propagation delay also decreases. But, through experimentation we found that if $\epsilon < 0.5$, although the frame propagation delay continued to decrease, contention increased. This was revealed through decreasing reception rates, increasing probabilities of backoffs and high collision percentages when attempting to send frames. In Section VI.1.6 we study the impact of changing ϵ on the frame propagation delay and collision percentage in more detail. So, for the remainder of this work, we use $\epsilon = 0.5$ and $T_{max} = 200\,\text{ms}$. These values have been selected based on the experiments in Section VI.1.6.

VI.1.4 Local Topology Sensing

The local network topology, or the density of vehicles within the transmission range, is sensed in p-IVG using a beacon broadcasting utility running in each vehicle. This utility broadcasts a beacon containing the vehicle's information (location, speed, etc.) at least twice a second[1]. Each vehicle maintains a database to store the information of vehicles within its transmission range. Upon receiving a beacon, each vehicle updates its database with the new vehicle information. The database is checked every second. If there has been no update beacon received regarding a specific

---

[1]The safety related applications should perform a broadcast every $100\,\text{ms}$ (10 times per second) [81]

vehicle, this vehicle's record will be marked and its location information will be estimated using the most recent location and speed information. If the estimated location lies outside the transmission range, the vehicle's record will be purged from the database. If there has been no update beacon received regarding a specific vehicle for two consecutive checks (*i.e.*, no update beacon received from the vehicle for two seconds), its record will be purged from the database. As these beacons are used for local topology sensing only, they are not re-broadcast. We do not consider these beacons as overhead because most vehicular applications already require each vehicle to broadcast its information frequently. These application messages, then, could be used to determine vehicle density.

VI.1.5 Vehicular Density

Vehicular density is a key parameter that has a great influence on broadcasting techniques. Some of these techniques may fail in certain vehicular densities, so it is important to study the impact of density on our proposed technique, *p*-IVG. In the case of IVG, as the density increases, the number of candidates for re-broadcasting a frame increases. This increases collisions, resulting in reduced reception rates. For *p*-IVG, the number of candidates for re-broadcasting a frame remain almost the same regardless of the density. This is because the number of candidates depends on both the density and the probability function that is inversely proportional to the density. So, as the density increases, the probability function decreases, and vice versa, keeping the number of candidates almost constant, typically between 1 and 3 vehicles. This makes *p*-IVG provide stable performance regardless of the surrounding vehicular density.

VI.1.6 Extended Analysis of ϵ

Due to the importance of the ϵ parameter, we dedicate this section to experimentally analyzing it. The main goal of the experiments that we performed is to study the impact of changing ϵ on *p*-IVG performance, represented as collision percentage and message travel time.

To emphasize the impact of varying the value of ϵ on the performance metrics, we fixed all other parameters. In these experiments, the vehicular density used is 66 vehicles/km, which is classified as medium density according to Table 6[2] and we

---

[2]This table is a copy of Table 8.

TABLE 6: Simulation Settings for Extended Analysis of ϵ Experiment

transmission range	300 m
highway length	100 km
max distance traveled	10 km
vehicles generated	500
max speed	30 m/s
simulation runtime	360 seconds
high density	90 vehicles/km
medium density	66 vehicles/km
low density	53 vehicles/km

set the message sending rate to three messages/second. Hence, in each experiment we only need to change ϵ and measure the performance evaluation metrics, collision percentage and message travel time.

If a vehicle veh_x starts to receive a message msg_A and, while receiving this message, another vehicle within the transmission range of veh_x starts to send a message msg_B, both msg_A and msg_B will collide at vehicle veh_x. Although vehicle veh_x will discard both of these two messages, msg_A and msg_B, they will be logged by vehicle veh_x and marked as collided.

To calculate the percentage of collided messages at any certain vehicle, we simply need to process the log file for this vehicle and calculate the ratio between the number of messages marked as collided to the total number of messages received (collided and not collided) by this vehicle. The collision percentage at any vehicle is the same as the percentage of collided messages calculated for the same vehicle. Hence, we can define the *collision percentage* for any experiment run as the average of all individual vehicles' collision percentages.

Calculating the message travel time is a bit tricker than the collision percentage, because p-IVG has a constraint on the maximum lifetime for rebroadcasted messages, *i.e.*, not to exceed 1 second. So, we have two alternatives, either relax this constraint for all messages in the experiment, or select a subset of these messages and mark them to be excluded from being checked against this constraint. The problem with the first alternative is that relaxing this constraint for all messages will let them propagate further and hence change the p-IVG behavior, which is not recommended. So in this

experiment, we will randomly select a subset of the messages to be marked at sending time and let them travel up to 8 km, then calculate the actual travel time they took to travel this distance. The maximum distance that a message can travel depends on the location of the sending vehicle. The messages sent by the front vehicles[3] will have more chance to travel longer distance than the messages sent by the vehicles that entered the simulation recently. Because we want the marked messages to travel up to 8 km, we will favor the front vehicles' messages when selecting the messages that will be marked as they are more likely to travel up to the required distance. Although p-IVG minimizes the chance of having more than one vehicle rebroadcast the same message, it is possible. In this case, we will consider only the first instance of each marked message that reaches the 8 km limit and record its travel time and discard the other instances. By averaging the message travel time for all the marked messages, we can calculate the average message travel time for the experiment.

Before we start explaining and analyzing the experiment's results, we will consider the expected behavior for both collision percentage and message travel time as ϵ changes. According to the timer value T_x (Equation 19), as ϵ decreases, T_x will decrease too. The positive side of having a short timer value, T_x, is that it will cause the messages to be rebroadcasted faster and hence take a shorter travel time to reach distant areas. While the negative side for having short timer value, T_x, is that it implies short listening time[4], which will result in premature rebroadcast decisions and hence redundant rebroadcasts and more collisions. The opposite behavior is expected for both collision percentage and message travel time as ϵ increases. As ϵ increases, T_x will increase. Having long timer value will result in longer message travel time, at the same time having longer timer value implies better chance to listen longer to the media and avoid any premature rebroadcast decisions and hence avoid any redundant rebroadcasts and result in fewer collisions. To summarize, having smaller ϵ results in generating smaller T_x which produces smaller message travel time and higher collision percentage while, having larger ϵ results in generating larger T_x which produces longer message travel time and lower collision percentage. Based on the expected behavior for both collision percentage and message travel time as ϵ changes, we conclude that there is a trade-off between these two metrics, which makes finding

---

[3]The front vehicles are the vehicles that entered the simulation early so that they come at the front of other vehicles that entered the simulation at a later time.

[4]The listening time is the time period during which a vehicle listens to the media to check whether someone else has already rebroadcasted the same message.
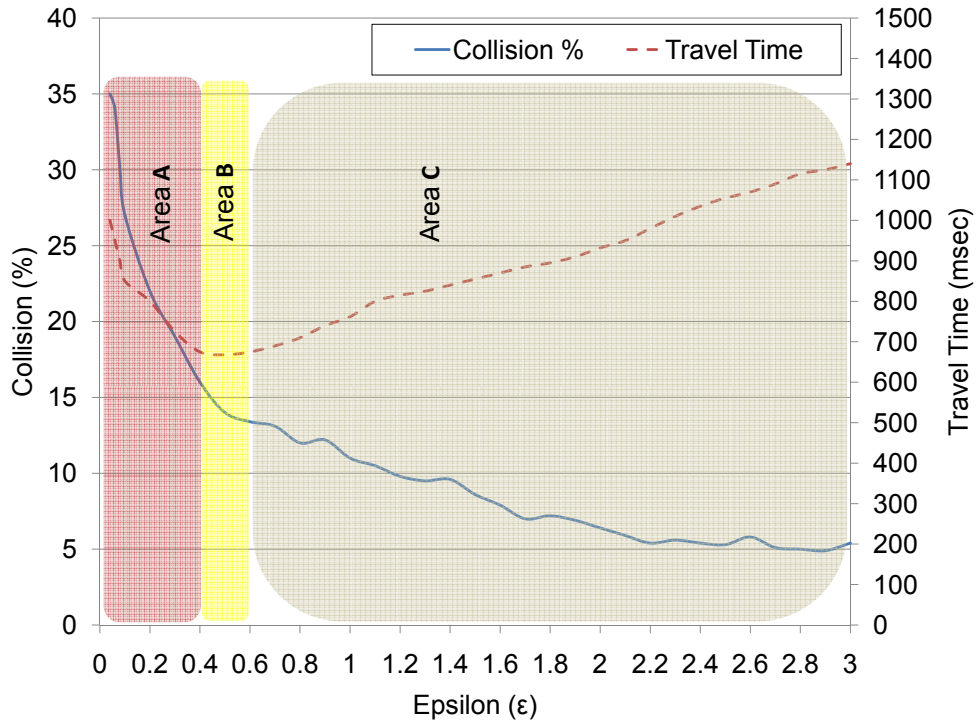
FIG. 32: Collision % and Message Travel Time as ϵ Changes

an optimal value for ϵ that will enhance both metrics not straightforward.

Figure 32 shows the impact of changing ϵ on both the collision percentage and message travel time. This was calculated experimentally. Based on how the performance metrics react to change in ϵ, we can identify three distinct areas in the figure, areas A, B and C.

In area A, ϵ ranges from 0.0 to less than 0.4, resulting in small T_x values, which tend to zero as ϵ approaches zero. We expect to see small message travel times and high collision percentages as ϵ gets smaller, but contrary to our expectation both the message travel time and collision percentage increase exponentially. After analyzing the trace files, we noticed that the marked messages are either marked as collided, and hence never reach the 8 km limit, or marked as received but their corresponding travel times are much higher than expected. The reasons for such unexpected behavior are explained below:

- As T_x decreases, the message collision probability increases. This increase grows exponentially as T_x approaches zero, which causes huge message loss, including the marked messages. As we only record the message travel time for

the marked messages when they reach the 8 km limit, the lost marked messages will not have travel time associated with each one of them, so we penalize the lost marked message to have maximum message lifetime, 1 second, as its travel time

- As ϵ decreases, T_x for vehicles at the transmission range boundary will be very small, close to zero, that results in having those vehicles rebroadcast their messages simultaneously. So, many collisions occur in this area and hence all the messages rebroadcast by the vehicles there are lost. Because the vehicles at the transmission range boundary will not be able to make any successful rebroadcasts, the vehicles closer to the sender will have more chance to do the rebroadcasts. This implies that the messages will visit more hops to reach its 8 km limit and hence they will have a longer travel time

To summarize the performance metrics behavior in area A, having very small ϵ results in generating small, close to zero, T_x which produces *longer message travel time* and higher collision percentage.

Area B spans the following range, $0.4 \leq \epsilon \leq 0.6$, for ϵ. In this area, as ϵ increases, T_x increases too, but the message travel time looks almost constant. The collision percentage continues to decrease, which nominates this range to be the optimal range for selecting the best value for ϵ. In this area, the value of ϵ increases with a very small step, which makes all the ϵ values close to each other and hence results in having T_x values with tiny differences, which produces almost constant message travel time. The performance metrics in area B follow the following behavior, larger ϵ results in longer T_x which produces *almost constant message travel time* and less collision percentage.

Area C, $0.6 < \epsilon \leq 3.0$, is the only area in which the actual results match our expectations. In this area, as ϵ increases, the message travel time increases due to the T_x increase, and also the collision percentage follows the expected pattern and continues to decrease until it saturates at almost 5.5%.

To conclude, according to the current simulation settings of message sending rate and vehicular density, area B is the optimal area from which ϵ should be selected. Changing any of the simulation settings, especially the message sending rate, may change the behavior. So, to pick the optimal ϵ value for different settings we should regenerate the corresponding curves for the new settings. Because p-IVG is tolerant

to vehicular density change, the selection of the optimal ϵ should not be impacted much with a change in the vehicular density.

VI.2   TRAFFIC DISCONTINUITY PROBLEM

In Chapter I, we explained two possible communication models, V2V and V2I, that are widely used in VANETs and their corresponding pros and cons. In the V2V communication model, only the vehicles driving along the road will be involved in disseminating the vehicular data. Most often, data relevant to a particular vehicle will come from vehicles traveling in the same direction as it. So, we can classify how the vehicular data will be disseminated till it reaches the vehicle, according to the participating vehicles' direction, into three models: same direction, opposite direction and both directions data. Figure 33 shows an illustration of the three data dissemination models.

In case of the same direction data dissemination model, vehicles will *only* be involved in disseminating the vehicular data for the vehicles that are traveling in the same direction as them, as shown in Figure 33-a. Figure 33-b shows the opposite direction data dissemination model in which the vehicles driving in certain direction will be involved *only* in disseminating the vehicular data for the vehicles that are traveling in the opposite direction and will do nothing for the vehicular data for vehicles traveling in the same direction as them. In the both directions data dissemination model shown in Figure 33-c, all vehicles will be involved in disseminating the vehicular data for the other vehicles regardless of their direction.

These three data dissemination models have been studied by Nadeem *et al.* [92], and the results showed that using the opposite direction data dissemination model will disseminate the vehicular data faster than the other two dissemination models. But what has not been considered in this study is the impact of having a discontinuity in the traffic flow in a certain direction. Also, the case in which there are signal impairments between the two roads *i.e.*, long or dense trees between the two roads, will make the opposite direction data dissemination model not the best data dissemination model to be used. In this chapter, we adopt a hybrid data dissemination model that dynamically switches from one model to another according to the traffic conditions to alleviate the previous problems.

In Section VI.1, we explained how the vehicular data will be disseminated in CASCADE. The data dissemination model that we used was the same direction
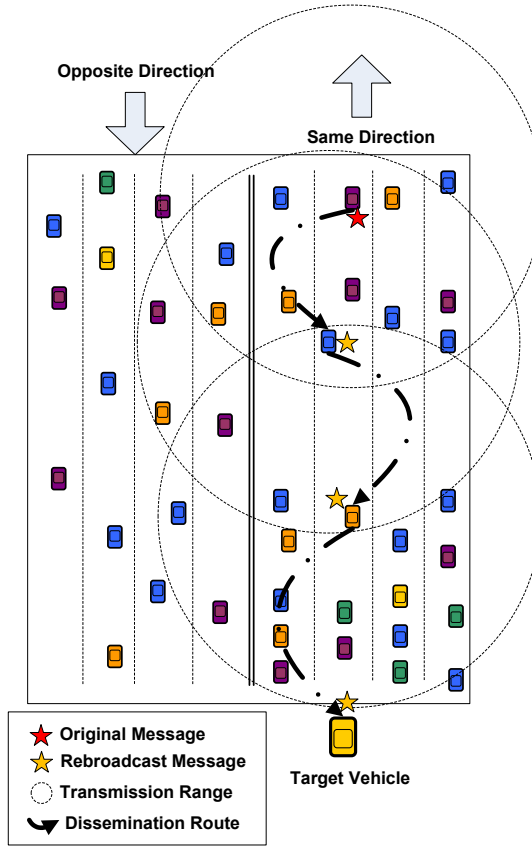
(a) Same Direction Data Dissemination Model
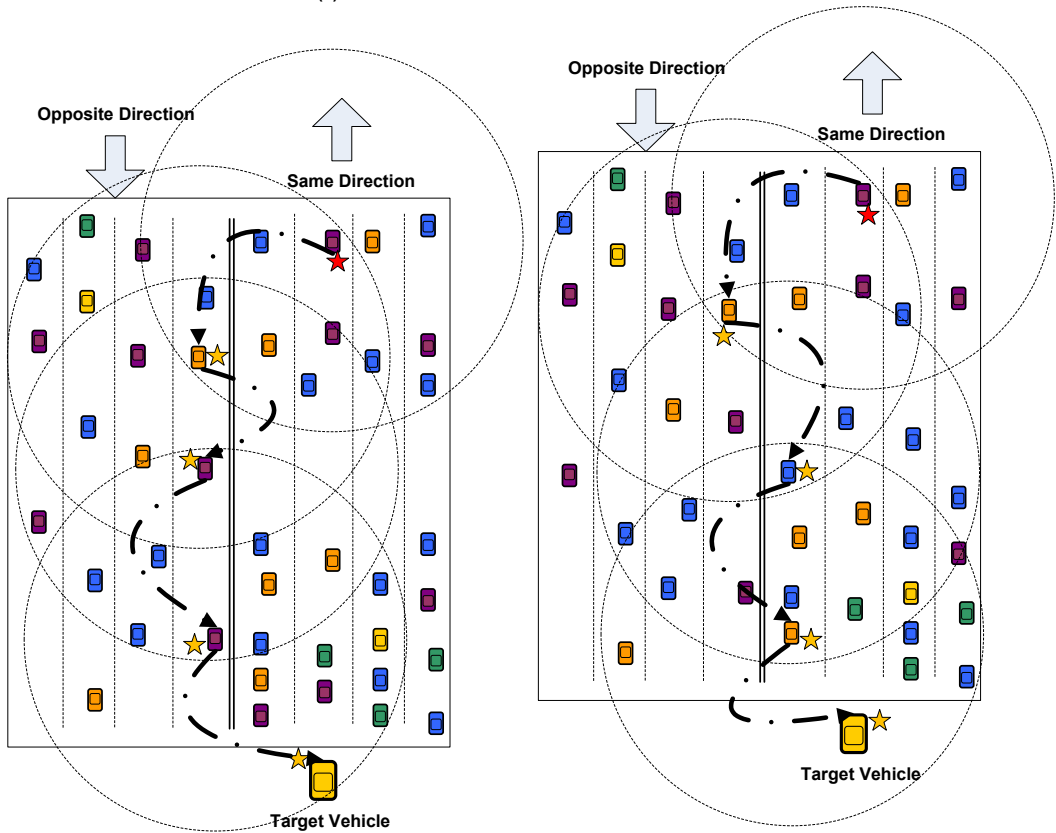
(b) Opposite Direction Data Dissemination Model

(c) Both Direction Data Dissemination Model

FIG. 33: Data Dissemination Models

data dissemination model. So before we start explaining how CASCADE will use a hybrid data dissemination model to overcome the traffic discontinuity problem, we will show the impact of using the opposite direction data dissemination model in CASCADE instead of the same direction data dissemination model.

VI.2.1 Using the Opposite Direction Data Dissemination Model in CAS-CADE

Earlier in this chapter, we have explained p-IVG, the data dissemination technique used in CASCADE. Although p-IVG is based on the same direction data dissemination model, it showed a good results with respect to MAC delay, dissemination delay, hop count used, etc, as we will see in Chapter VII. In this section, we investigate the impact of using the opposite data dissemination model with p-IVG instead of the same direction data dissemination model.

Among the metrics that have been used to evaluate p-IVG with the same direction data dissemination model, only the dissemination delay and hop count metric are expected to be impacted by replacing the same direction data dissemination model with the opposite direction data dissemination model. One of the points that should be mentioned here before addressing it in more detail later in this section is the correlation between the data dissemination model that p-IVG is using and the security enforcement component used in CASCADE (Chapter V). Briefly, the location estimation based on received signal strength module in the security enforcement component is optimized more to work with p-IVG when the data dissemination model being used is the same direction data dissemination model. However, the security enforcement component will continue to work properly, but with extra overhead in the case of using the opposite direction data dissemination model.

In the following sections, we conduct two experiments to study the correlation between the data dissemination model used by p-IVG and the security enforcement component. In the first experiment, we disable the security enforcement component to study the impact of using the opposite direction data dissemination model with p-IVG. In the second experiment, we enable the security enforcement component to study the extra overhead resulting from using the security enforcement component in CASCADE while having p-IVG using the opposite direction data dissemination model.

FIG. 34: Dissemination Delay and Hop Count at Medium Density: Opposite Direction Data Dissemination Model with p-IVG and Security Component Disabled

Using The Opposite Direction Data Dissemination Model with p-IVG: Security Enforcement Component Disabled

In CASCADE, p-IVG by default uses the same direction data dissemination model, in which only the frames coming from vehicles driving in the same direction are processed, while the frames coming from vehicles driving in the opposite direction are filtered out. The filtering operation for those unwanted frames is performed by setting up a frame filter to prevent the frames from being processed by the system. In the case of opposite direction data dissemination, the filtering logic should be reversed so that it allows the frames coming from vehicles driving in the opposite direction to be propagated upward in the frames processing stack, while filtering out the frames coming from vehicles driving in the same direction. In this experiment, we have disabled the security component of CASCADE to get accurate results when comparing the two dissemination models.

Figure 34 shows the message propagation delay and hop count used by p-IVG when using the same direction data dissemination model and the opposite direction data dissemination to propagate a message to certain distance. As we see in the figure, using the opposite direction data dissemination model showed better performance than the same direction data dissemination as it reduces the propagation delay and hop count by 7% - 10%. This performance enhancement is expected and
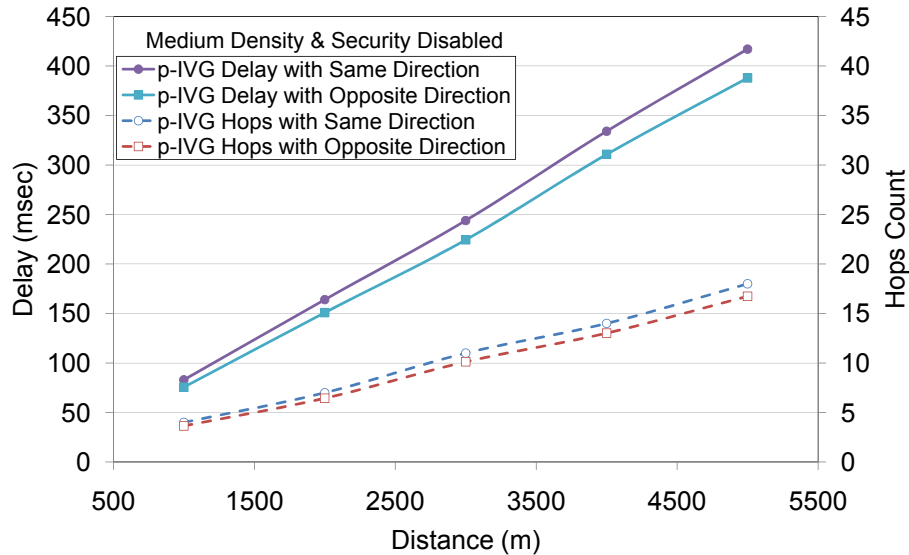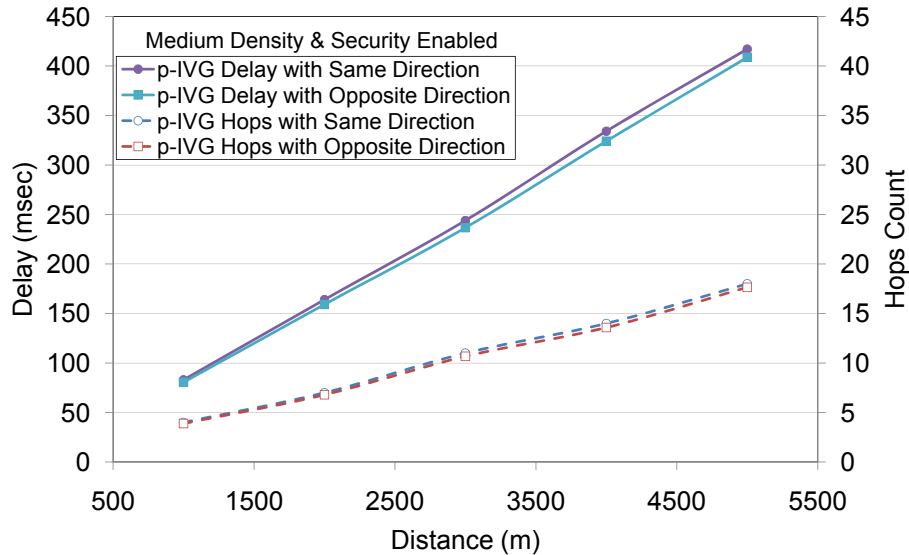
FIG. 35: Dissemination Delay and Hop Count at Medium Density: Opposite Direction Data Dissemination Model with p-IVG and Security Component Enabled

has been shown before by Nadeem *et al.* [92].

Using The Opposite Direction Data Dissemination Model with p-IVG: Security Enforcement Component Enabled

The location estimation based on the received signal strength module in the security component used by CASCADE depends on receiving the frames from surrounding vehicles driving in the same direction to analyze the signal strength that carried those frames and estimate the sending vehicle's location. To let this module operate as usual and at the same time use the opposite direction data dissemination model, we have to disable the frame filter as we explained earlier. As a consequence of disabling this filter, the number of frames being processed by the system will increase greatly and hence cause some delay in CASCADE's various components.

In this experiment, we study the impact of using p-IVG with the opposite direction data dissemination model and disabling the frame filter. Figure 35 shows the experiment results. As we see, using p-IVG with the opposite direction data dissemination model still has better performance metrics, message propagation delay and hop count, than the case when using p-IVG with the same direction data dissemination model, but its advantage has been reduced from 7% - 10% to 2% - 3%. This performance advantage reduction is a result of the extra overhead introduced into

the system after disabling the frame filter and allowing more frames to get into the system, slowing down the system components and hence reducing the whole system performance.

To conclude, using p-IVG with the opposite direction data dissemination model takes slightly less time and fewer hops to disseminate the data than when using the same direction data dissemination model.

There is a strong assumption that has been used by p-IVG, which is there will not be any communication discontinuity among the vehicles driving along the road. This assumption does not reflect reality, so in Section VI.2.2 we will study the discontinuity problem, its impact on CASCADE, and how this problem can be solved.

VI.2.2   Traffic Discontinuity Problem in CASCADE

The ideal situation in the V2V communication model is to have each communicating vehicle along the road be able to continuously receive messages from the surrounding vehicles, assuming the surrounding vehicles are also communicating vehicles. This situation is called continuous communication flow. In CASCADE, a vehicle is only concerned with receiving messages from the vehicles ahead of it on the road. So, we refine the previous definition as the following, the ideal situation in the V2V communication model in CASCADE framework is to have each communicating vehicle along the road be able to continuously receive messages from the vehicles ahead, assuming the vehicles ahead are also communicating vehicles. The previous situation does not reflect reality, because for many reasons any communicating vehicle may not be able receive messages from the vehicles ahead. We call this situation discontinuous communication flow.

Discontinuity in communication flow may happen for one or more of the following reasons:

1. **Low Penetration Rate**: In case of early deployment for any V2V communication system it is highly possible to have a high percentage of vehicles not equipped with communication devices, so that they will not be able to communicate with the surrounding vehicles. With respect to the communicating vehicles, those non-communicating vehicles will be communication-wise non-existent although they physically exist. Having such a situation may end up causing the Inter-Vehicle Gap problem, because the distance between the communicating vehicles will be greater than $Effective\_Transmission\_Range$

2. **Inter-Vehicle Gap**: This gap will cause communication flow discontinuity if it is greater than $Effective\_Transmission\_Range$, which is 250 m. This gap may happen because of either clustered or sparse traffic. More details about these two causes and how to distinguish between them will be provided later in this section, but the final picture is that the distance between a communicating vehicle and the very first communicating vehicle ahead is greater than 250 m

3. **Signal Decay and Distortion**: It is also possible to have the gap between two communicating vehicles less than the $Effective\_Transmission\_Range$ though they cannot communicate with each other due to signal impairments *i.e.*, road curves, weather conditions, etc., that all contribute to decay the signal, in addition to signal distortion due to interference

In this section and the following sections, we are going to focus on the second reason for communication flow discontinuity (Inter-Vehicle Gap) to study its causes in more detail and its impact on CASCADE. In addition, we will present a solution to alleviate this problem.

As we mentioned before, the inter-vehicle gap problem will happen if the gap between the vehicles is greater than 250 m which can happen in both sparse and clustered traffic, as we see in Figure 36-a and Figure 36-b. As 250 m is the minimum distance required to maintain continuous connection among the vehicles, this implies that the minimum number of vehicles required to avoid the inter-vehicle gap problem in 1 km is 4 vehicles, assuming that they are uniformly distributed along the road. To distinguish whether the gap is caused by sparse or clustered traffic, the average vehicular density should be calculated and if its value is less than 4 vehicles then the gap is due to sparse traffic, otherwise it is due to clustered traffic.

The discontinuity in the communication flow greatly impacts CASCADE, specifically the local and extended views in CASCADE, because this discontinuity breaks the primary and aggregated frames flow and causes the length of these views to drop sharply. If the discontinuity lasts for an extended period of time, the length of the views reaches zero. As most roads have two directions, there will be four cases for how the discontinuity will happen in these two directions. Figure 37 shows the four possible cases, and Table 7 summarizes them.

As we see in Figure 37, in case 1, neither of the two directions suffers from the discontinuity problem. In both case 2 and 3, one of two directions suffers from communication flow discontinuity and hence one of them will have discontinuity problem

(a) Sparse Traffic   (b) Clustered Traffic
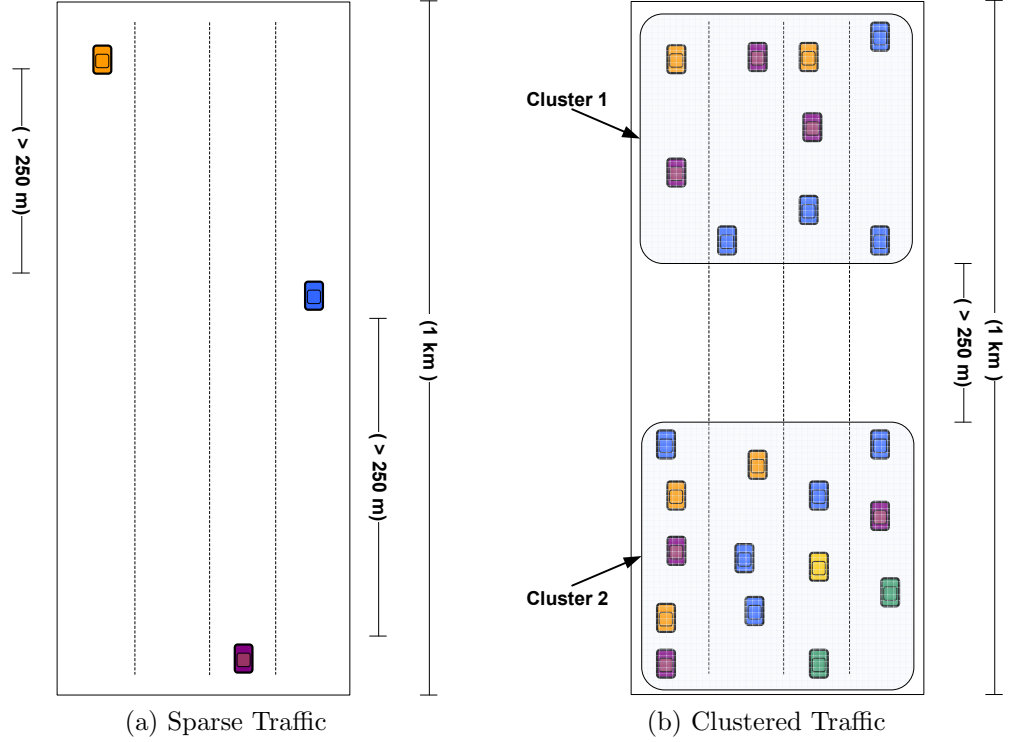
FIG. 36: Discontinuity Problem

based on which data dissemination model is used in p-IVG. In case 3, assuming p-IVG is using same direction data dissemination model, this means the same direction will suffer from the discontinuity problem, while the opposite direction will not. If p-IVG is using opposite direction data dissemination model, this means the same direction will not be suffering from the discontinuity problem, while the opposite direction will be. The same logic applies in case 2. The reason for having one of the two roads suffering from the discontinuity problem is that p-IVG is using only one data dissemination model, either same direction data dissemination model or opposite direction data dissemination model. The optimal data dissemination technique should detect the communication flow discontinuity and switch between the data dissemination models accordingly to avoid having any discontinuity problem. We applied this dissemination technique and it showed good results, see Section VI.2.3. In Case 4, as both directions suffer from communication flow discontinuity, the same solution applies here too.

In the following section, we propose a new hybrid data dissemination model that
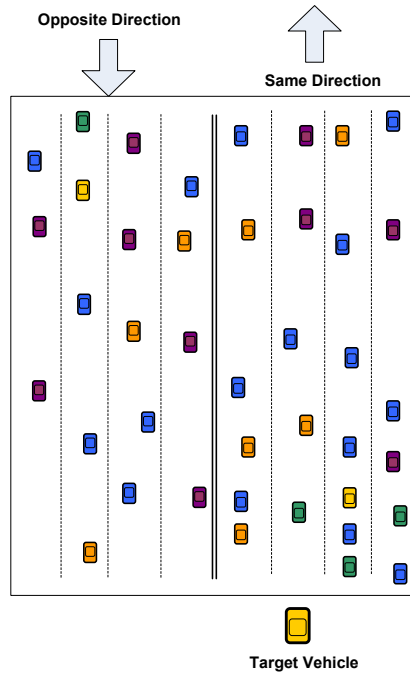
(a) Case 1
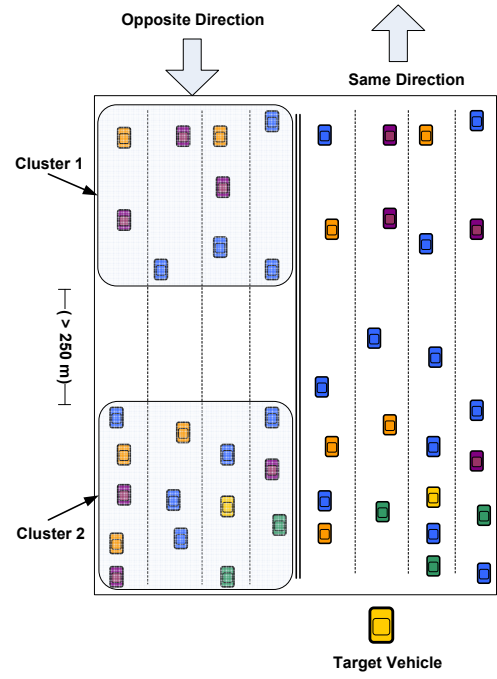
(b) Case 2
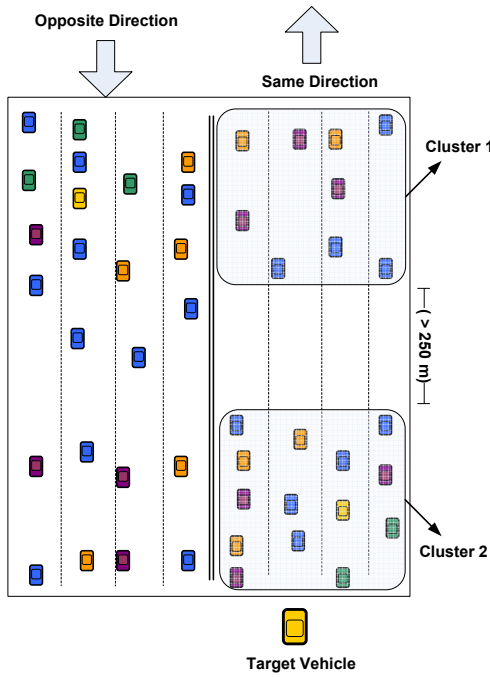
(c) Case 3
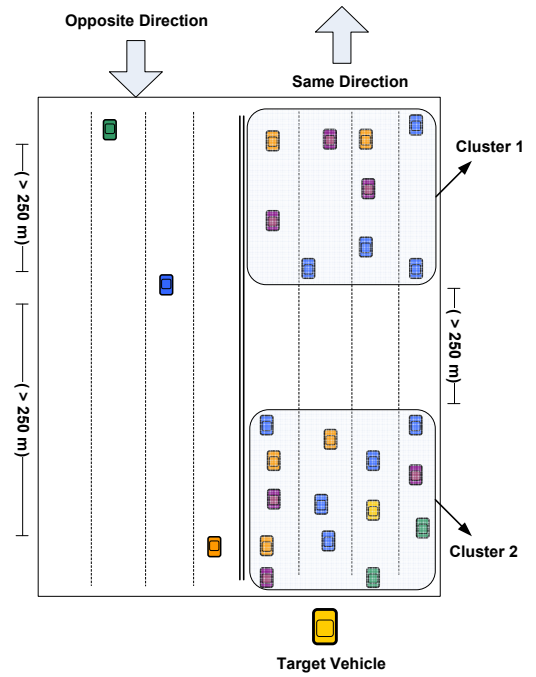
(d) Case 4

FIG. 37: Discontinuity Problem in Two Directions Highway: The Possible 4 Cases

TABLE 7: Discontinuity Problem Cases Summary

		Opposite Direction Connectivity Status	
		Connected	Disconnected
Same Direction Connectivity Status	**Connected**	*Case 1*	*Case 2*
	Disconnected	*Case 3*	*Case 4*

p-IVG will use to avoid the discontinuity problem. This hybrid model detects the communication flow discontinuity and switches between the data dissemination models accordingly to avoid having any discontinuity problem. To evaluate the new hybrid model, we measure the extended view length for certain vehicle considering three scenarios. In the first scenario, there is no discontinuity problem. This scenario is our reference scenario. In the second scenario, our target vehicle will suffer from the discontinuity problem. This scenario will show the impact of discontinuity problem on the extended view length. In the third scenario, p-IVG will be using the new hybrid model. This scenario will show how the new model will detect the communication flow discontinuity and react to it to recover the extended view.

VI.2.3 On-Demand Vehicular Gap-Bridging (OD-V-GB)

The techniques used to handle the discontinuity problem can be categorized as either reactive or proactive techniques. The main characteristic of the reactive techniques is that they start working only when the discontinuity problem is detected so they react to the occurrence of the problem, hence the discontinuity problem can never be

avoided using these techniques. What distinguishes each technique in this category is how fast it can recover from the discontinuity problem and at what cost. But the common thing among the techniques in this category is that their cost, specifically the extra overhead that they introduce into the system, is low. The other category is proactive, in which the techniques anticipate the occurrence of the problem and start early, so that the discontinuity problem never happens. But on the other hand they can introduce much overhead into the system.

The technique that we present in this section to handle the discontinuity problem is called On-Demand Vehicular Gap-Bridging, OD-V-GB. It adopts a hybrid data dissemination model of same direction data dissemination model and opposite direction data dissemination model. OD-V-GB is a reactive technique. The main data dissemination model in OD-V-GB is the same direction data dissemination model and whenever a vehicular gap is detected, which is the cause of the discontinuity problem we are handling here, it switches to use the opposite direction data dissemination model to bridge this gap and recover from the discontinuity problem.

OD-V-GB introduces a new message type called Gap Bridging Request (GBR) that is broadcasted by any vehicle whenever it senses that there are no vehicles behind it, which implies that there is a potential gap. The vehicles that will react to receiving the GBR messages are the vehicles coming in the opposite direction. The reaction is to simply rebroadcast the aggregated frame(s) that summarizes the extended view from the direction requesting the gap bridging.

The gap bridging technique consists of three modules:

Broadcasting GBR Messages Module: The functionality of this module is simple, but crucial at the same time to the whole solution, as it is the triggering point to start the recovery or the gap bridging. This module monitors the status of the vehicles behind by storing the time for the most recently received message from them. Whenever this module does not receive any message for certain amount of time (the *idle_threshold*), it concludes that there must be gap and hence sends out a GBR message

Handling Received Aggregated Frames Module: The main functionality of this module is to maintain a short repository for the aggregated frames received from the vehicles in the opposite direction, the direction requesting the gap bridging. The repository should store the aggregated frames for no more

than 2 seconds (the aggregated message lifetime) then purge them. On receiving these frames and before storing them in the repository, the overlapped sections between these frames will be trimmed. Upon receiving frames from vehicles on the side requesting the gap bridging, no processing will be performed on these frames. In other words, the matching technique will be disabled so that keeping these overlapped sections will be an extra overhead that will cost more broadcasting and receiving time and also more bandwidth consumption, so keeping these overlapped sections will add no value. This module will be running in each vehicle as a continuous background task with low priority, so that it will never compete with any other higher priority applications that may be running simultaneously on the system. Upon receiving a GBR message, this module's priority gets raised to receive more CPU cycles and react faster

On Demand Broadcasting Module: This module will stay idle most of the time and will be activated only on receiving a GBR message. Once activated, it will start consuming frames from the repository in an ascending time order (the oldest frames first) and broadcast them. The logic behind sending the frames in this order is to simulate as if the receiving vehicles are actually receiving those frames from the original senders, so that they can build their extended view correctly. The frames that are consumed from the repository will be purged even if its life is less than 2 seconds to avoid broadcasting the same frame twice

Another functionality for the broadcasting GBR messages module that is not related to the OD-V-GB technique yet critical to CASCADE, is to update the expiration and purging threshold for aggregated frames. In a normal situation where there is no discontinuity detected and it is guaranteed to receive continuous stream of aggregated frames, we use the aggregates frames TTL (2 seconds) as the threshold for expiring and purging the aggregated frames composing the extended view. Upon detecting a traffic discontinuity, it is better to keep the current extended view at its current length even if its data will be a bit obsolete rather than letting its length drop to zero before the OD-V-GB starts working. So, the broadcasting GBR messages module increases the threshold for expiring and purging the aggregated frames to 10 seconds till connectivity is detected. Then it will reset the threshold back to its normal level of 2 seconds. Due to the importance of this functionality, the broadcasting GBR messages module has been modified to operate in isolation of

OD-V-GB and be part of CASCADE to perform the above functionality (updating the threshold) even if OD-V-GB is not active.

VI.3   SUMMARY

In this chapter, we presented the probabilistic Inter-Vehicular Geocast (p-IVG) protocol for data dissemination in dense VANETs. p-IVG was designed to alleviate the spatial broadcast storm problem that we discovered when studying standard IVG. p-IVG uses a light-weight beacon broadcasting utility to sense the local network topology and adapts itself to minimize the number of frame re-broadcasts needed to send a frame to distant area.

Also, we presented in this chapter the various data dissemination models, same direction, opposite direction and both directions data dissemination model, that can be used in the V2V data communication model. It has been proven by Nadeem *et al.* [92] and confirmed by our experiment results that the opposite direction data dissemination model is better than the other two models as it can disseminate the data to further distance using less time and hop count. Due to the correlation between the data dissemination model that p-IVG is using (same direction data dissemination model) and the security enforcement component in CASCADE, changing p-IVG to use the opposite direction data dissemination model still has better performance metrics, but its advantage has been reduced from 7% - 10% to 2% - 3%.

We also, presented the traffic discontinuity problem that may happen in any V2V system whenever the distance between any two communicating vehicles exceeds 250 m due to either sparse or clustered traffic and the data dissemination model used by this system is *only* the same direction or the opposite direction data dissemination model. As the previous two conditions are satisfied in CASCADE, this implies that it is vulnerable to the traffic discontinuity problem. To overcome this problem in CASCADE, we proposed a solution to this problem called On-Demand Vehicular Gap-Bridging or OD-V-GB technique. The evaluation of both p-IVG technique and OD-V-GB technique is presented in Chapter VII.

CHAPTER VII

EVALUATION

Each component in CASCADE framework that we presented in the previous chapters has been evaluated using ASH simulator, which will be discussed in detail in Appendix A. In this chapter, we discuss the evaluation for each component, including the experiment settings and the evaluation metrics. We compare the proposed techniques used in each component to other techniques and analyze the results.

In Section VII.2, we evaluate the local view component of the CASCADE framework and the performance of the p-IVG technique in the context of CASCADE. Section VII.3 presents the evaluation for the defense technique proposed in Chapter V, in addition to assessing the overhead caused. The evaluation of data dissemination, p-IVG, independent of the CASCADE framework is presented in Section VII.4. In Section VII.5, we evaluate the OD-V-GB technique and show how efficiently this technique handles the discontinuity problem in CASCADE.

VII.1 METHODOLOGY

ASH is the simulator used in evaluating the various CASCADE framework components. ASH consists mainly of two components, the network component and the mobility component, between which we enabled the two-way communication. Also, ASH has an implementation for a configurable two-way highway model. Our simulations are based on the well-known Intelligent Driver Model (IDM) [60] that describes vehicular mobility. More detail about ASH is presented in Appendix A.

Table 8 summarizes the simulation settings that we have used to evaluate CASCADE. All vehicles in our simulations have a transmission range of $300\,\text{m}$[1] [84]. The roadway used is a four-lane divided highway of length $100\,\text{km}$, shown in Figure 38. Vehicles enter the highway according to a Poisson distribution and travel at a maximum speed of $30\,\text{m/s}$. In most cases, we completed several runs of each experiment and show the average behavior. Each simulation is run for 360 seconds, resulting in a total of 500 vehicles generated. In the 360-second simulation runtime, the maximum distance traveled by any vehicle is $10\,\text{km}$.

---

[1]The maximum transmission range for the DSCR is $300\,\text{m}$, while the effective transmission range is $250\,\text{m}$.
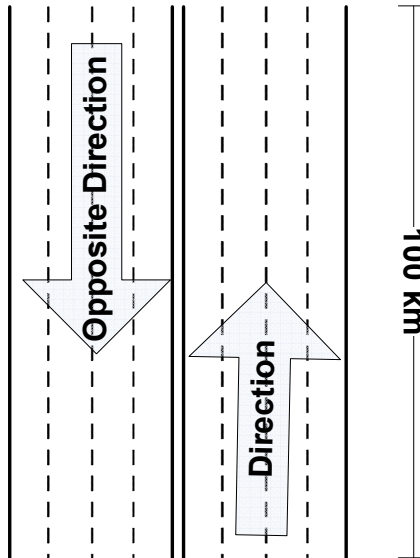
FIG. 38: Highway Configuration Used in the Simulation

We evaluate CASCADE with three different traffic density scenarios. In the high density case, there are an average of 90 vehicles/km. In medium density traffic, there are an average of 66 vehicles/km, and in low density, there are an average of 53 vehicles/km. These densities were gathered from the speed and traffic volume analysis performed by Wisitpongphan *et al.* [93] for the data collected by the Berkeley Highway Lab for traffic on eastbound I-80 on June 27, 2006 [94].

VII.1.1 CASCADE Evaluation Metrics

We have used the following three metrics, the MAC delay, reception rate and visibility, to evaluate CASCADE. In addition to the previous three evaluation metrics, we may use some other evaluation metrics for evaluating a specific system component.

MAC Delay

The MAC delay represents the time between the MAC layer receiving the frame for transmission and delivering it to the physical layer. IEEE 802.11a by default does not use the RTS/CTS mechanism for reserving the wireless channel, so collision avoidance based on detecting the channel idle for a certain amount of time is used. If the wireless medium is very busy, the MAC delay will increase because the sender will not be able to detect that the channel is idle for the entire required period. We

TABLE 8: Simulation Settings

transmission range	300 m
highway length	100 km
max distance traveled	10 km
vehicles generated	500
max speed	30 m/s
simulation runtime	360 seconds
high density	90 vehicles/km
medium density	66 vehicles/km
low density	53 vehicles/km

show this MAC delay as an approximation to the number of wireless collisions, which we cannot directly measure.

Reception Rate

We measured the average reception rate over time. If a frame was transmitted and no other vehicle received it, then the frame was considered to not be received. A reception means that at least one vehicle has received the message. Either the frame experienced a collision or no other vehicle was within 300 m of the sender.

Visibility

The goal of CASCADE is to provide information about upcoming vehicles, so visibility is one of the most important metrics. We consider visibility to be the distance between a vehicle and farthest vehicle in its extended view. Thus, as a vehicle's visibility increases, its knowledge about upcoming traffic conditions increases.

VII.2   LOCAL VIEW EVALUATION

In the evaluation of the local view component, we consider the effects of data compression and the effect of using p-IVG over IVG or flooding for data dissemination.

VII.2.1 CASCADE Compression Evaluation

In CASCADE, we use a variation of differential coding to compress the vehicles' data in the local view in order to form a small aggregated frame. This process is described in detail in Chapter III. To evaluate how well CASCADE compression works, we compare it with lossless Deflate compression [95] and differential coding without using clusters. Deflate combines the LZ77 algorithm, which replaces a repeated pattern with a pointer to the first occurrence of the pattern, with Huffman coding, which assigns a shorter code for more frequently-used symbols. Deflate will work best on data that has many values in common. For differential coding, we took the first vehicle in the local view and calculated the difference between its' properties and the remaining vehicles in the local view. The main difference between applying differential coding on either the local view or each cluster is that in the former case the data fields in the aggregated record will be longer because the differences will be the maximum of the local view size rather than the maximum of the cluster size.

The aggregated frame size resulting from CASCADE depends upon both the number of vehicles in the local view and on the distribution of the vehicles within the local view clusters [96]. In our evaluation, we consider the worst case vehicle distribution over the local view clusters, resulting in the maximum aggregated frame size (CASCADE-Max).

Figure 39 shows the aggregated frame sizes for uncompressed, Deflate, differential coding, and CASCADE-Max as the number of vehicles in the local view increases. The dotted line represents the maximum MAC-layer frame size, which is 2312 bytes. The X-axis value at the intersection point between this dotted line and each of the compression techniques curves represents the maximum number of vehicles that can be represented in a single frame. CASCADE-Max provides the best performance, allowing for 864 vehicles to be represented in a single frame, while differential coding can represent 723 vehicles. Deflate has similar performance as the uncompressed case, only representing 129 vehicles in a single frame.

Each compression technique has overhead. For small data sizes, the sum of the compressed data size and the overhead is often larger than the uncompressed data size. Furthermore, the performance of a compression technique depends on the characteristics of the data that is being compressed. The main characteristic of the data in the local view is that the values are similar. That is why differential coding and CASCADE produce much better compression than Deflate, which depends on
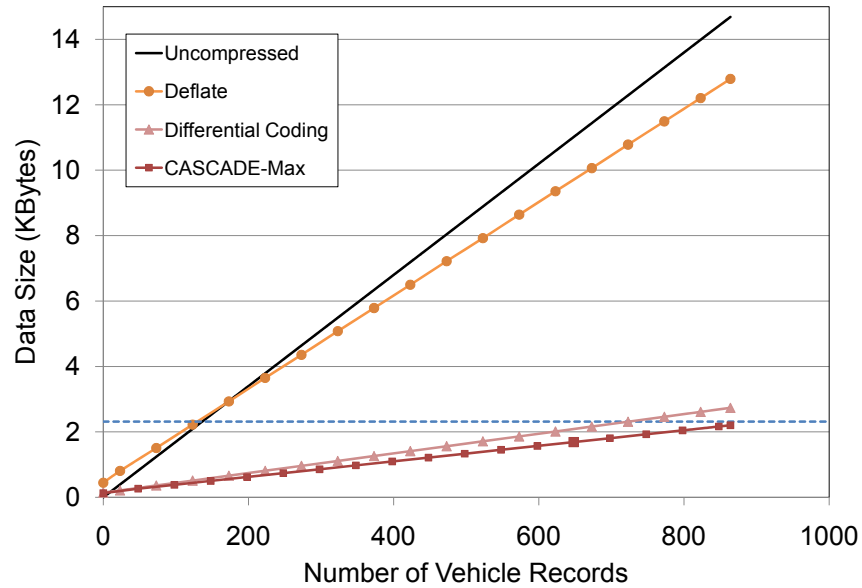
FIG. 39: Aggregated Frame Sizes Generated Using Different Compression Techniques

repeated, rather than similar, values.

To show the improvement provided by CASCADE data compression, we ran experiments where CASCADE either compressed the vehicles' data or sent the vehicles' data uncompressed considering three possible vehicular densities, low, medium and high. The results presented in Figures 40, 41, 42 and 43 are only for medium vehicular density case. However, the experiments results in case of low and high densities were consistent with the results in medium density case.

MAC Delay

In Figures 40 and 41, we show the cumulative distribution functions (CDFs) of the MAC delay experienced by primary frames and aggregated frames, respectively, in case of using compression or sending the vehicles' data with no compression in medium traffic density.

Both primary frames and aggregated frames see about 25% less MAC delay when using compression. This is because CASCADE compression provides smaller aggregated frames that will take less transmission time, thus making the wireless channel more available for both primary and aggregated frames.

FIG. 40: MAC Delay for Primary Frames at Medium Density
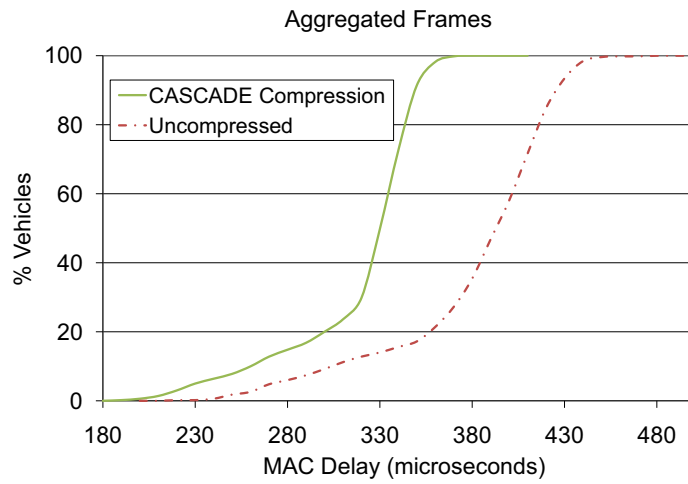
FIG. 41: MAC Delay for Aggregated Frames at Medium Density
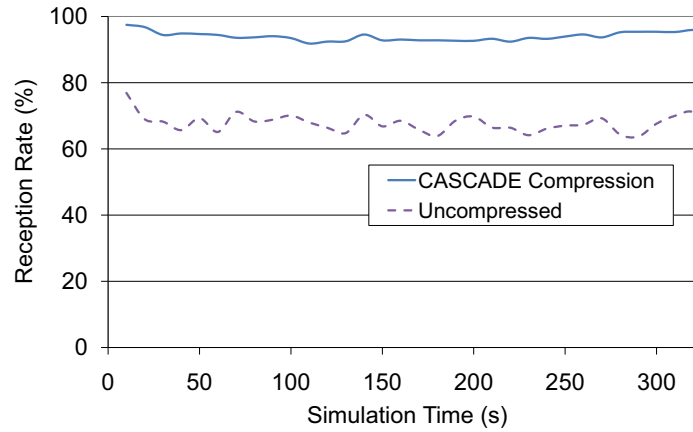
FIG. 42: Reception Rate at Medium Density

Reception Rate

Figure 42 shows the reception rate of CASCADE when using either compression or no compression at medium traffic density, averaged every 10 seconds.

Using CASCADE with compression results in at least a 45% higher reception rate than using CASCADE with no data compression. That is due to the large aggregated frame size generated when no compression is used, which increases the collision probability and reduces the reception rate.

Visibility

Figure 43 shows the percentage of vehicles that have a particular minimum visibility when using either compression or no compression at medium density traffic. Using CASCADE with compression increased the vehicles' visibility with almost 100% over when no compression is used.

For example, with CASCADE compression, 60% of the vehicles have a visibility of 3000 m or more, while with no compression, almost no vehicles have that much visibility. Visibility highlights the importance of the reception rate. If there are many collisions, then information is not able to be disseminated to distant vehicles, so their visibility is reduced.
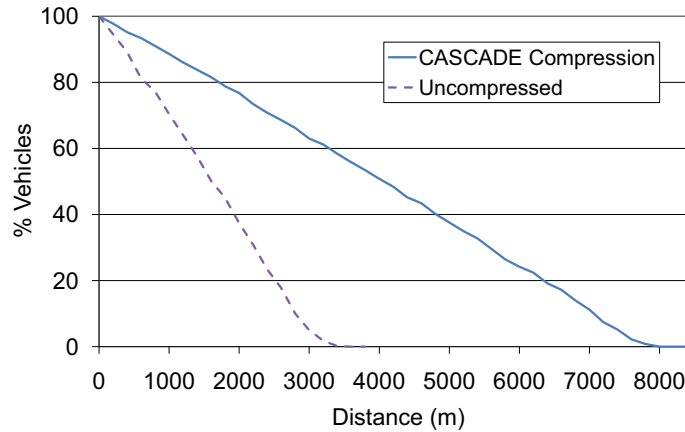
FIG. 43: Visibility at Medium Density

VII.2.2   Evaluating CASCADE with p-IVG

p-IVG is the main data dissemination technique used in CASCADE. p-IVG is considered the probabilistic version of the standard IVG technique, as it uses a probabilistic function based on the surrounding vehicular density in making the message rebroadcast decisions. Both p-IVG and IVG associate a timer with each message to be rebroadcasted, if the timer expires and no other vehicle rebroadcasts the same message, IVG will deterministically decide to rebroadcast the message while p-IVG will depend on its probabilistic function in deciding whether to rebroadcast the message or not. p-IVG is presented in more detail in Chapter VI.

To show the improvement provided by p-IVG, we ran experiments where CASCADE used either basic flooding, standard IVG, or p-IVG.

MAC Delay

In order to investigate the impact of using p-IVG with CASCADE, we measured the amount of MAC-layer delay for each frame (split into primary frames and aggregated frames). In Figures 44 and 45, we show the CDFs of the MAC delay experienced by primary frames and aggregated frames for the three re-broadcast schemes (flooding, IVG, and p-IVG) at medium traffic density.

Primary frames and aggregated frames see the smallest MAC delay when p-IVG is used and the largest delay when basic flooding is used. p-IVG is also shown to be an improvement upon standard IVG. This is because p-IVG limits the number of nodes that may be re-broadcasting frames at the same time, thus making the wireless
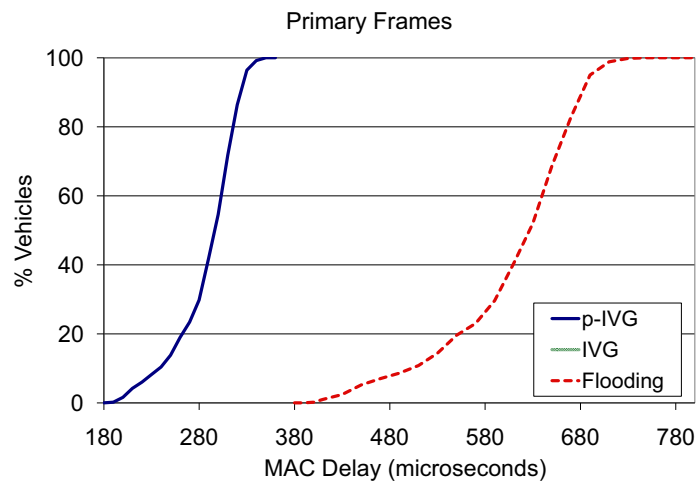
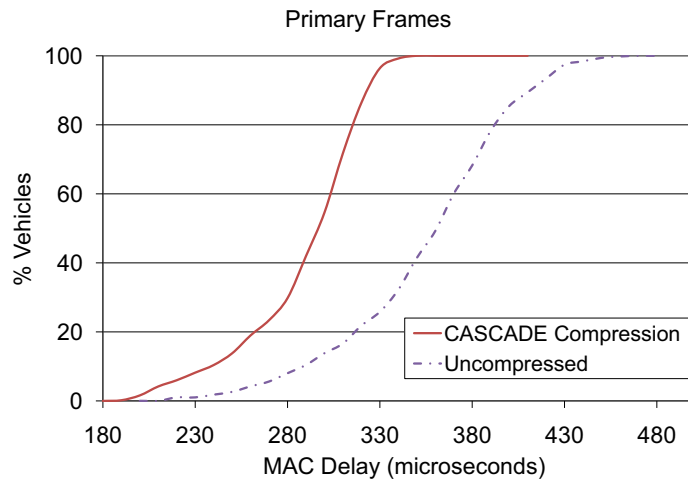FIG. 44: MAC Delay for Primary Frames at Medium Density
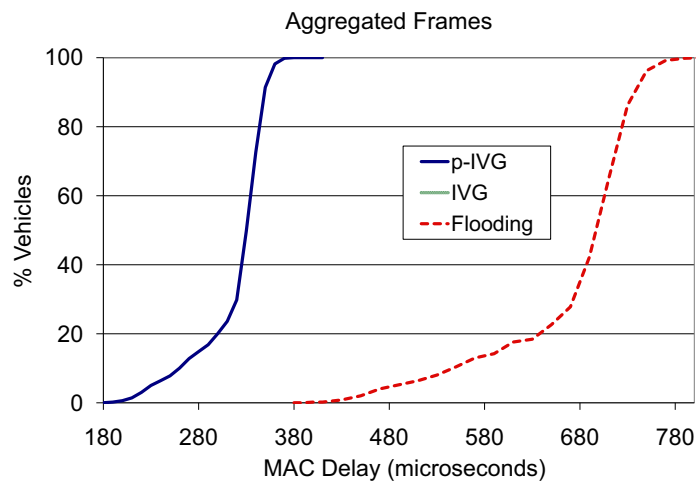

FIG. 45: MAC Delay for Aggregated Frames at Medium Density
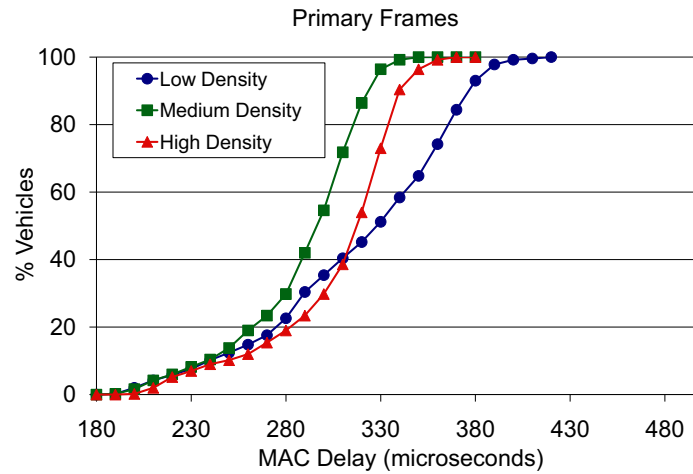
FIG. 46: MAC Delay for Primary Frames with p-IVG
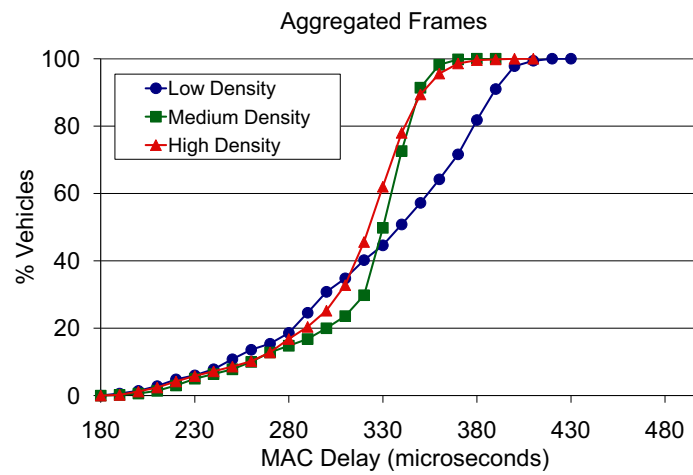
FIG. 47: MAC Delay for Aggregated Frames with p-IVG

channel more available.

Figures 46 and 47 show the CDFs of MAC delay experienced by primary frames and aggregated frames when using p-IVG over various traffic densities. The maximum MAC delay is still relatively small even at high densities. The key point is that with p-IVG, higher density does not necessarily mean more frames are sent, because density affects the amount of re-broadcasted frames. At higher densities, fewer vehicles will re-broadcast frames.
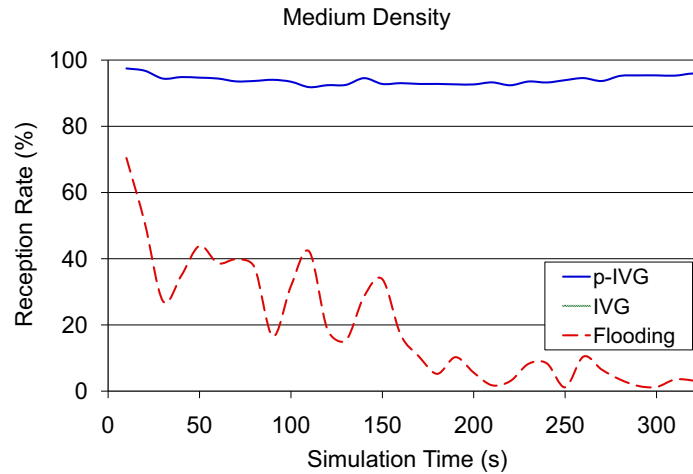
Medium Density

FIG. 48: Reception Rate at Medium Density
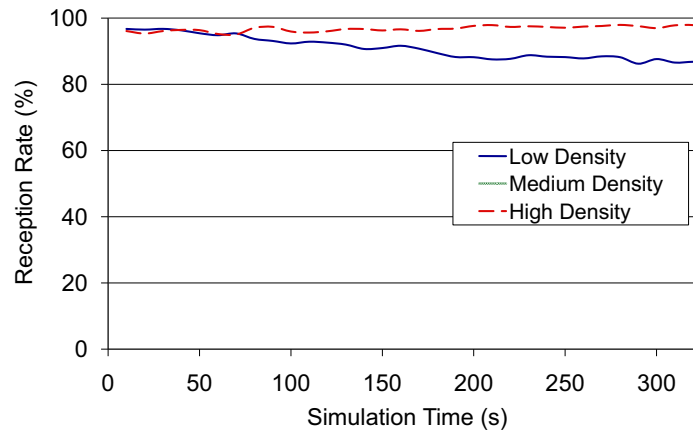
FIG. 49: Reception Rate with p-IVG

Reception Rate

To assess the impact of wireless collisions, we measured the average reception rate over time. Figure 48 shows the reception rate of CASCADE when using p-IVG, IVG, and flooding with medium density traffic, averaged every 10 seconds. The reception rate for IVG and flooding starts high because the number of vehicles in the simulation is still increasing. As more vehicles enter the system, the reception rate decreases. Using p-IVG results in a very high reception rate for the entire simulation.

In Figure 49, we show the reception rate for p-IVG with each of the three traffic densities. In our definition, if just one vehicle receives the frame, it is considered
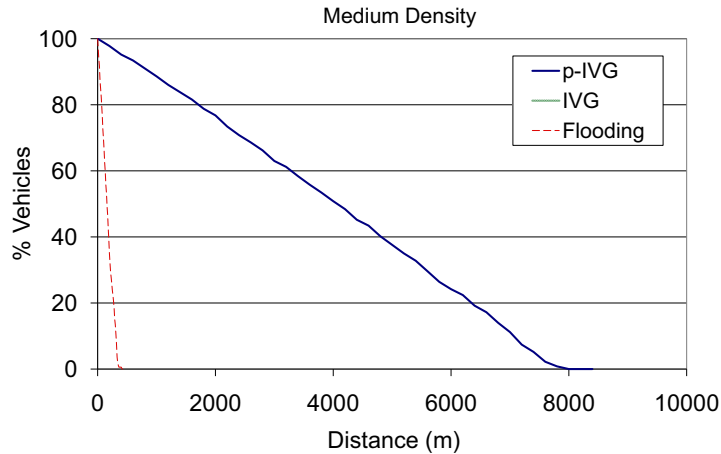
FIG. 50: Visibility at Medium Density

received. Higher density provides the best reception rate because there are more vehicles in range, thus more of a chance to receive the frame. The decreasing reception rate in the low density case is an artifact of how vehicles enter the simulation. Vehicles enter the highway with a low speed and gradually increase towards the maximum of $30\,\mathrm{m/s}$. Until the last vehicle reaches the maximum speed, vehicles in front of it will be traveling faster, thus moving out of its transmission range. For low density, it takes longer for 500 vehicles to enter the system than with high density, so it takes longer for the last vehicle to reach its maximum speed.

Visibility

Figure 50 shows the percentage of vehicles that have a particular minimum visibility with the three re-broadcasting techniques at medium density traffic. For example, with p-IVG, 60% of the vehicles have a visibility of $3000\,\mathrm{m}$ or more, while with standard IVG, almost no vehicles have that much visibility. Visibility highlights the importance of the reception rate. If there are many collisions, then information is not able to be disseminated to distant vehicles, so their visibility is reduced.

Figure 51 shows the visibility for p-IVG at high, medium, and low traffic densities. Consider that the optimal density for high visibility is to have exactly one vehicle positioned every $300\,\mathrm{m}$ (*i.e.*, at the boundaries). Thus, with low density, there is less of a chance that re-broadcasted aggregated frames experience collisions, so they are able to travel farther. As the density decreases, the chance of having a single
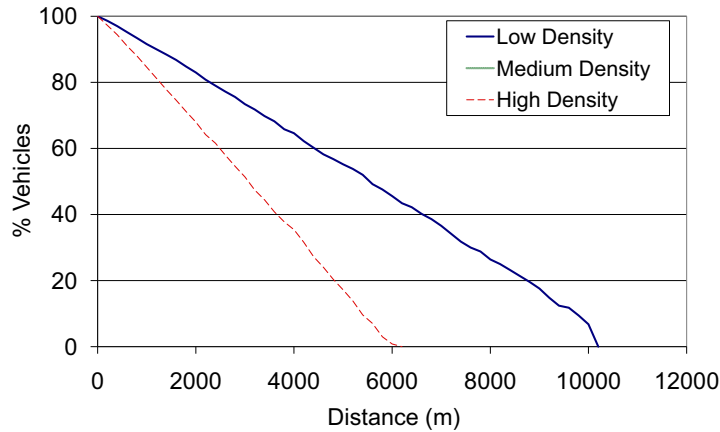
FIG. 51: Visibility with p-IVG

vehicle at the boundary increases. Since p-IVG only deals with nearby vehicles re-broadcasting frames at the same time, there may be other transmissions (such as the initial broadcast of primary or aggregated frames and re-broadcasts from vehicles in range, but not nearby) that may collide with the transmission.

VII.3   DATA SECURITY COMPONENT EVALUATION

Primary frames are considered the main input to the CASCADE framework. So, verifying the correctness of the vehicular data received in these frames, especially the vehicular position (X and Y), is crucial to guarantee the correctness and accuracy of the decisions made in CASCADE. To verify the received vehicular position, CASCADE uses a light-weight position verification technique that supplements the received signal strength (RSSI) technique with a laser rangefinder to reduce the inaccuracy incurred in the RSSI. This technique is not only able to identify and drop false vehicular data, but also can detect and quarantine the malicious vehicle sending this false data. Chapter V presents this technique in detail.

As mentioned in Section V.1.2, malicious vehicles can be grouped into four categories according to the source of the falsified information they send:

- *unsynchronized trace* - uses old positions, but the claimed position is outside the transmission range of the receiver

- *synchronized trace* - uses old positions, and the claimed location is inside the transmission range of the receiver

TABLE 9: Number and Percentage of Malicious Vehicles in Each Category in the Simulation

Unsynchronized trace	9 (2%)
Synchronized trace	15 (3%)
Replay	23 (5%)
GPS malfunction	12 (2%)
Total Malicious	59 (12%)

- *replay* - uses other vehicles' position information as inputs to CASCADE

- *malfunctioning GPS* - uses a malfunctioning GPS that will translate the GPS signals incorrectly

Malicious vehicles are injected onto the highway during the simulation and represent 12% of the total vehicles in the simulation. The number of vehicles in each category is shown in Table 9. There are 59 malicious vehicles out of a total of 500 vehicles.

VII.3.1   Defense with 100% Penetration Rate

The first experiment focuses on how well our proposed defense technique can detect malicious vehicles in each category, assuming all vehicles can communicate and use CASCADE (100% penetration rate). In Figure 52, we show both the average number of original primary frames sent by malicious vehicles and the average lifetime before a malicious vehicle is detected and quarantined the first time. The number of messages sent represents the bandwidth wasted by malicious vehicles before being caught.

The *unsynchronized trace* attack is the easiest to detect because the vehicles are claiming to be in a location that is outside the nominal transmission range.

The *synchronized trace* attack is a little harder because the vehicles claim to be within the transmission range, so they will have to be detected using the laser rangefinder. If there is no vehicle at the claimed location, this attack will be easily detected. But, as the traffic density increases, the probability of having another vehicle in the claimed location also increases.

In the *replay* attack, a vehicle claims to be at another vehicle's location. This attack is similar to the previous case where another vehicle is actually at the claimed
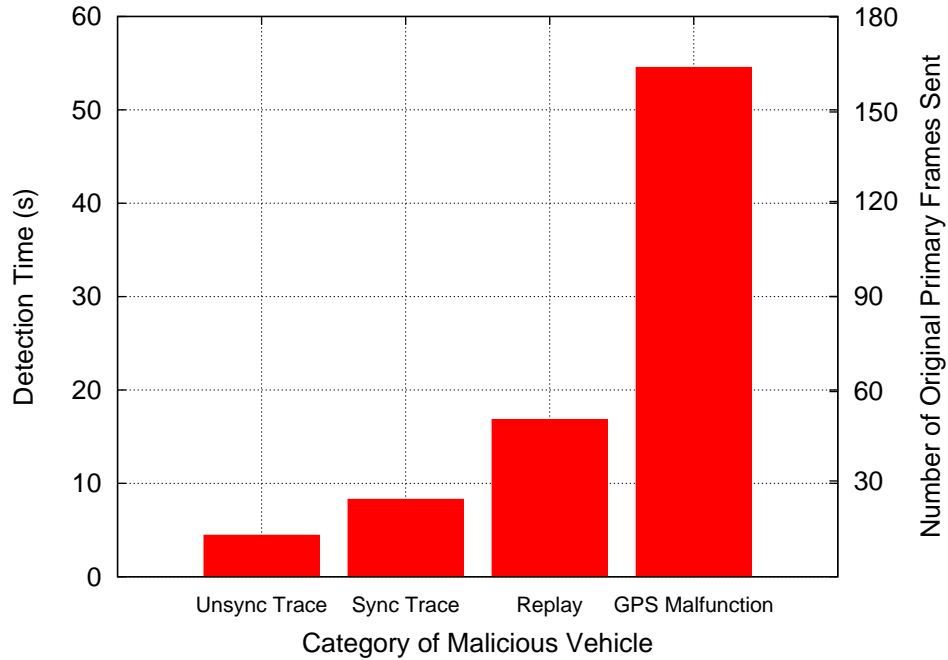
FIG. 52: Average Number of Original Primary Frames Sent and Average Time Elapsed Before First Quarantine for Each Category of Malicious Vehicle Assuming 100% Penetration Rate

location, but now the probability of having another vehicle at the claimed location is 1. Two vehicles will claim the same location, but one vehicle will already be in the local view, while the other is not. In addition, the second vehicle will have a signal strength that does not exactly match its claimed location. Thus, the second vehicle (the malicious one) will be quarantined after costing the defense technique more time.

False information due to *GPS malfunction* takes the longest time to detect especially when the difference is neither large nor persistent.

VII.3.2 Defense with Lower Penetration Rates

During the deployment of any VANET application, the penetration rate will be less than 100%. In this experiment, we explore the effect of having different penetration rates (70% and 30%) on the proposed defense technique.

In Figure 53, we show the detection time (*i.e.*, time until first quarantine) for each of the malicious vehicle categories. Having different penetration rates only greatly affects the detection time for the *synchronized trace* attack. As explained in the
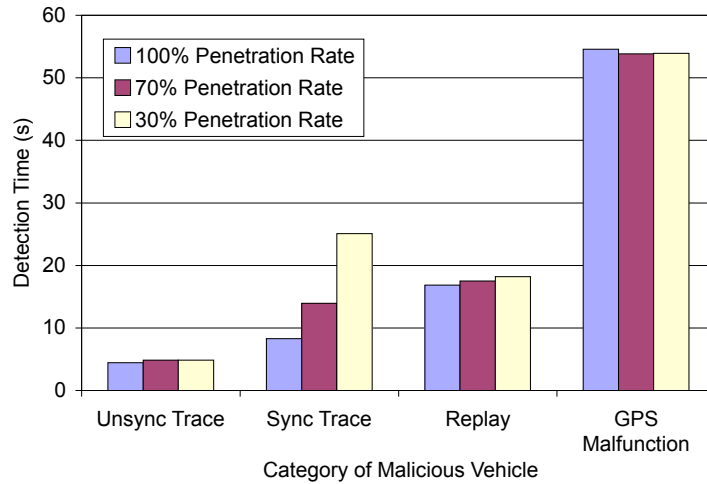
FIG. 53: Effect of the Penetration Rate on the Average Time Before the First Quarantine

previous section, this type of attack is detected either because there is no vehicle at the claimed location or because a communicating, honest vehicle is already at that location and in the detecting vehicle's local view. With a lower penetration rate, the claimed location may be occupied by a non-communicating vehicle. In this case, the detecting vehicle will have no existing entry in its local view and will accept the vehicle as honest. The malicious vehicle will continue to go undetected as long as it can pick locations that are occupied by non-communicating vehicles. But, it is only a matter of time before the malicious vehicle is detected. In addition, there is little advantage to impersonating a non-communicating vehicle; the impersonator is only making it seem that the non-communicating vehicle is communicating and that the attacker is not communicating.

We show in Table 10 how the defense technique treats the honest vehicles in the system. About 5% of the honest vehicles are quarantined once, and only 2% are quarantined twice. Less than 0.3% of the honest vehicles are quarantined for a third time. Honest vehicles may be quarantined due to the inaccuracy of the RSSI, the detecting sensors, or a sudden change in the driver's behavior (*i.e.* fast lane change). Another way that an honest vehicle can be quarantined is through malicious attack. If one of the malicious vehicles is replaying the target vehicle's position and has been added to a detecting vehicle's local view, it will be seen as honest. When the honest vehicle appears in the area for the local view, it will be considered malicious (as there

TABLE 10: Percentage of the False Positive Quarantines

Quarantined 1 time	5%
Quarantined 2 times	2%
Quarantined 3 times	0.27%
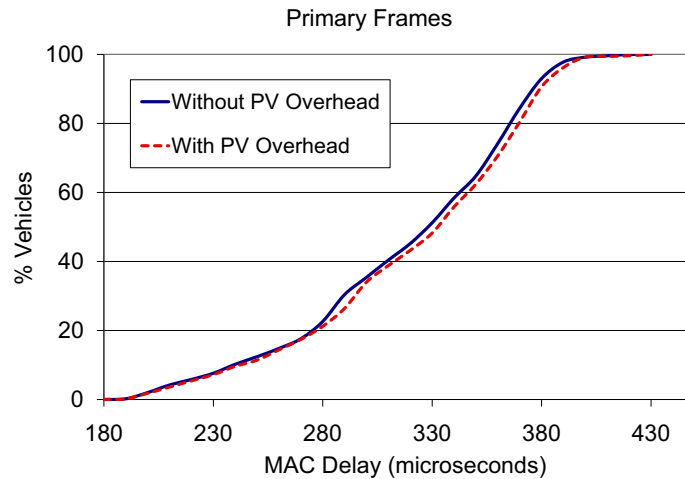
FIG. 54: MAC Delay at Low Density for Primary Frames

is already a supposedly honest vehicle in that location) and will be quarantined. As the quarantine process is only for certain amount of time, the honest vehicle will be able to communicate again after the quarantine timer expires.

VII.3.3 Effect of Position Verification Overhead

To evaluate the effect of overhead added by the position verification technique on CASCADE performance, we measured MAC delay, frame reception rate, and view update delay with and without position verification.

MAC Delay

In order to investigate the impact of using the position verification (PV) with CAS-CADE on the wireless channel, we measured the amount of MAC-layer delay for each frame (split into primary frames and aggregated frames).

In Figures 54-59, we show the CDFs of the MAC delay experienced by primary
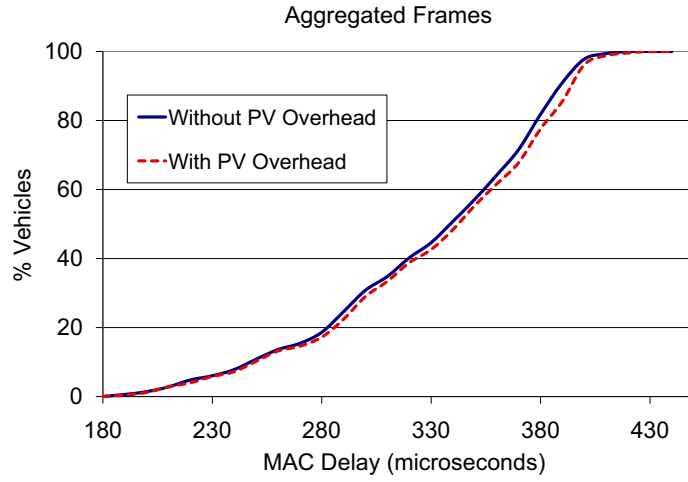
FIG. 55: MAC Delay at Low Density for Aggregated Frames
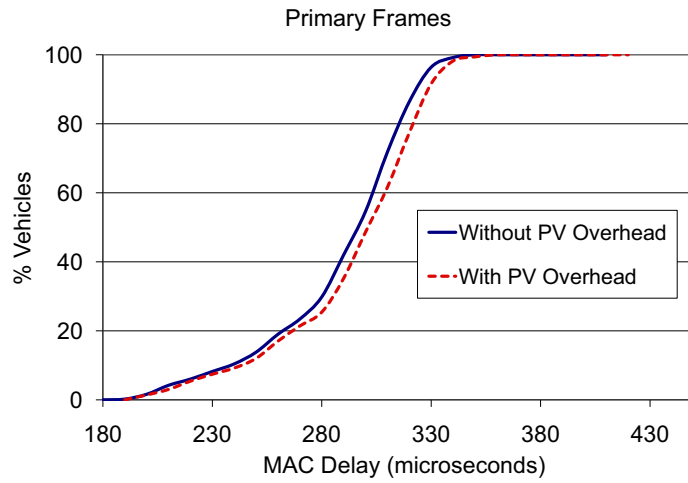
FIG. 56: MAC Delay at Medium Density for Primary Frames
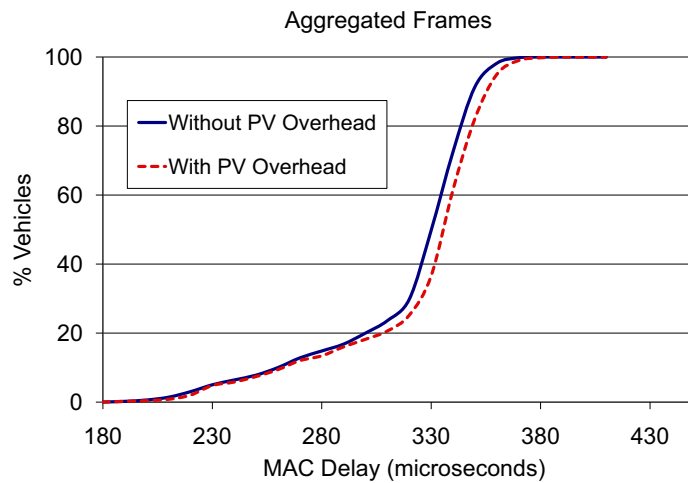
FIG. 57: MAC Delay at Medium Density for Aggregated Frames
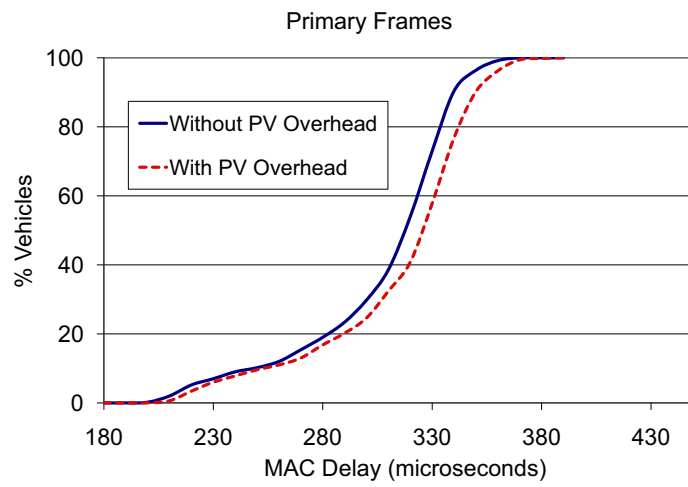
FIG. 58: MAC Delay at High Density for Primary Frames
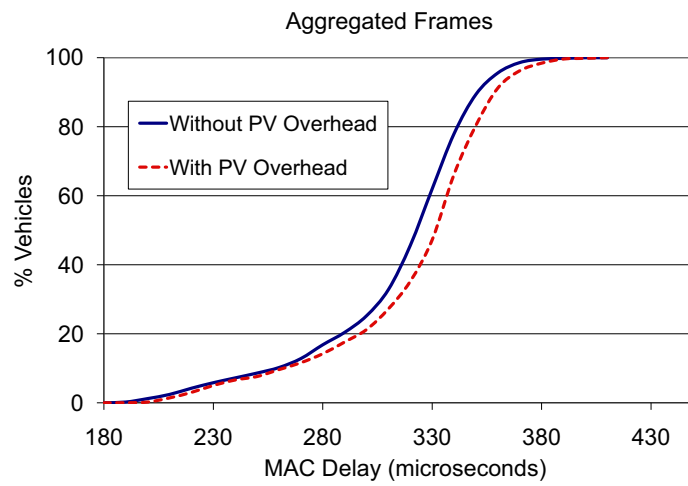
FIG. 59: MAC Delay at High Density for Aggregated Frames

frames and aggregated frames both with and without position verification in CAS-CADE. We considered three different traffic densities (low, medium and high). As shown in the figures, using position verification with CASCADE only adds a small increase in the MAC delay. The percentage of MAC delay increase rises with increasing traffic density. This is because the position verification does not use p-IVG dissemination when sending the quarantine proposals and requests. This implies that as the traffic density increases, more messages will be sent out and the competition on the wireless channel will increase, leading to an increase in the MAC delay. Still, the increase in MAC delay is on the order of a few tens of microseconds, which should not greatly affect the operation of CASCADE.

Reception Rate

To assess the impact of wireless collisions, we measured the average reception rate over time. If a frame was transmitted and no other vehicle received it, then the frame was considered to not be received. Either the frame experienced a collision or no other vehicle was within 300 m of the sender.

Figures 60-62 show the reception rate of CASCADE with and without position verification at different traffic densities (low, medium and high), averaged every 10 seconds. Using CASCADE with position verification results in slightly lower reception rate than without position verification. Again, as with MAC delay, this lower performance varies with the traffic density. The explanation for the lowered reception rate are the same as for the slightly increased MAC delay. The p-IVG dissemination technique is not used, and position verification adds some additional messages that take compete on the wireless channel with the primary and aggregated frames.

View Update Delay

As we have seen, using the position verification technique with CASCADE adds slightly more MAC delay and reduces the reception rate. Although the increase percentage in the MAC delay and the reception rate reduction percentage are very small, they may have an impact on the main CASCADE functionality of providing the driver with an up-to-date view of the traffic ahead. The rate at which the view is updated is a very important factor in the accuracy of the view. We measure the update delay as the time difference between receiving two consecutive aggregated frames, which is the main update source for the driver view.
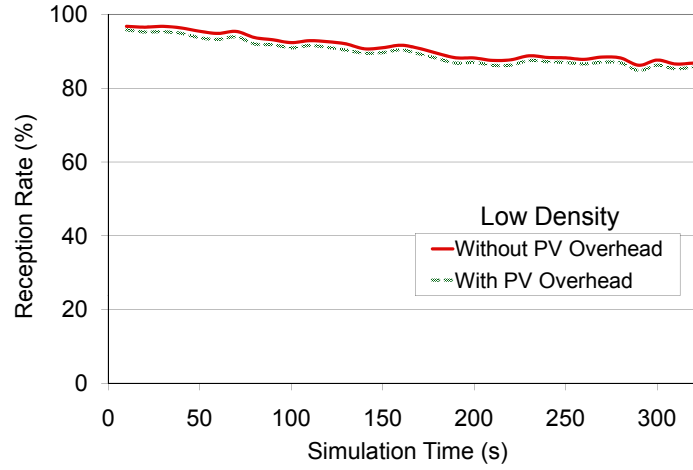
FIG. 60: Reception Rate at Low Density
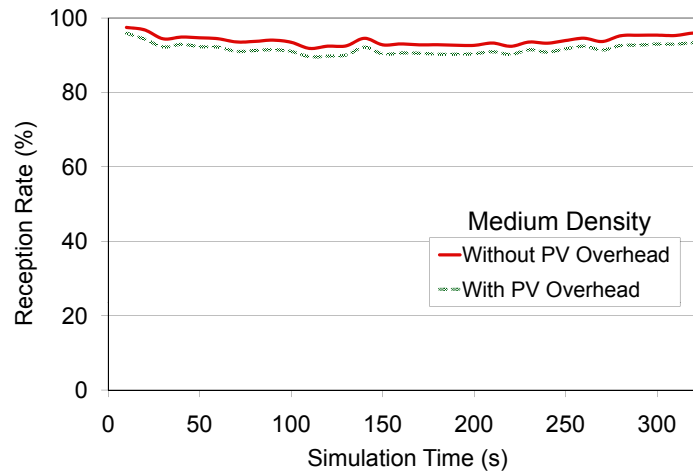
FIG. 61: Reception Rate at Medium Density
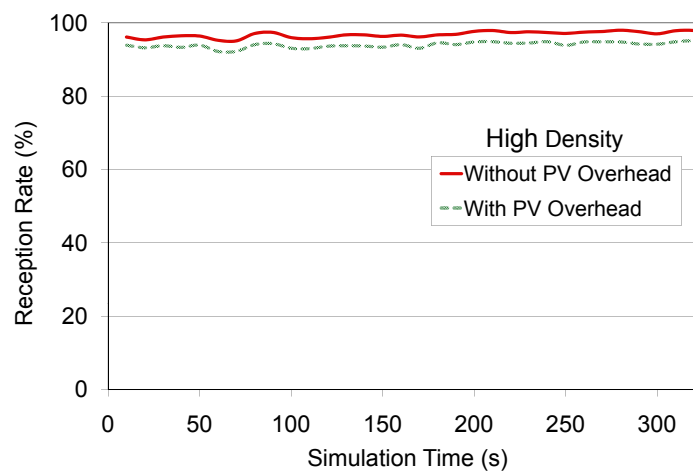
FIG. 62: Reception Rate at High Density
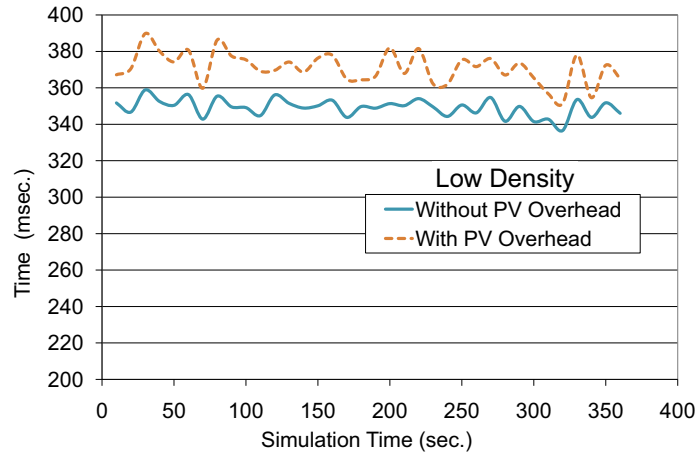
FIG. 63: View Update Rate at Low Density
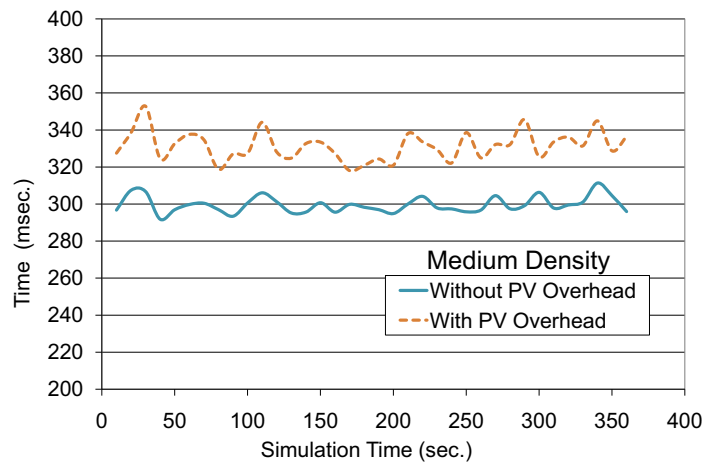
FIG. 64: View Update Rate at Medium Density

Figures 63-65 show the view update delay of CASCADE with and without position verification at different traffic densities (low, medium and high), averaged every 10 seconds. These figures show that although there is a slight increase in the view update delay, the difference even with high density traffic is only about 40 ms. This amount of time is so small that the driver will not be able to notice, much less react to, the difference in update delay.

We have shown that adding position verification to CASCADE can produce great benefit in terms of security from attack with little impact on the operation of CASCADE's main purpose, alerting the driver to upcoming traffic conditions.
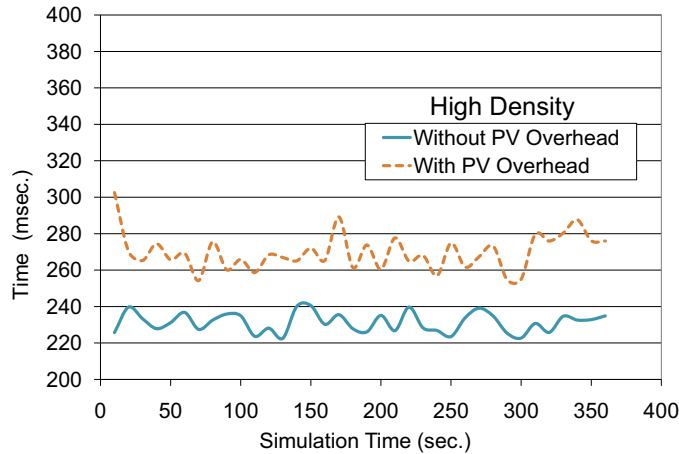
FIG. 65: View Update Rate at High Density

VII.4 PROBABILISTIC IVG (P-IVG) EVALUATION

p-IVG is a data dissemination technique that provides enhancement over standard IVG by making the message rebroadcast decision probabilistic based on the surrounding vehicular density instead of being deterministic. The CASCADE framework uses p-IVG as its main data dissemination technique. In Section VII.2.2, we evaluated p-IVG performance in the context of CASCADE, however p-IVG is loosely coupled with CASCADE framework. It can be used by other frameworks or applications independent of CASCADE. In this section, we evaluate the performance of p-IVG when a hypothetical application uses it in disseminating its messages. We run most of the experiments considering three possible vehicular densities, low, medium and high. In each of the three scenarios, all the vehicles in the simulation run an application that sends a dummy frame with the vehicle's location information every 300-400 ms. These dummy frames can be replaced by any other application's frames that need to be disseminated to a distant area. As mentioned in Section VI.1.4, beacon frames are not rebroadcasted, so only the application dummy frames will be rebroadcasted. To show the improvement provided by p-IVG, we ran experiments where the described application used either basic flooding, standard IVG, or p-IVG.
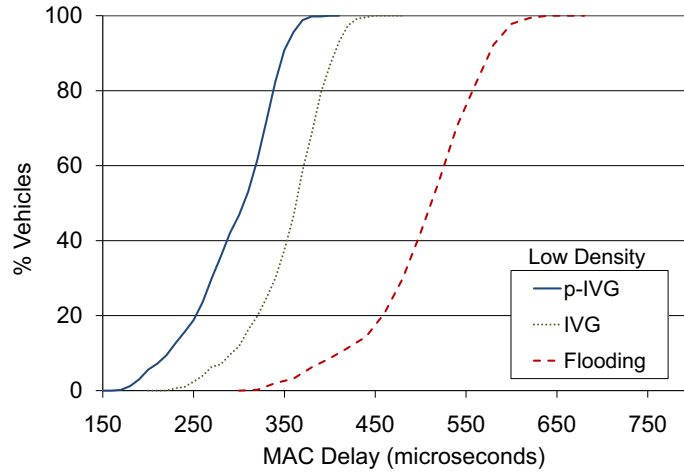
FIG. 66: CDF of MAC Delay for Beacon and Dummy Frames at Low Density
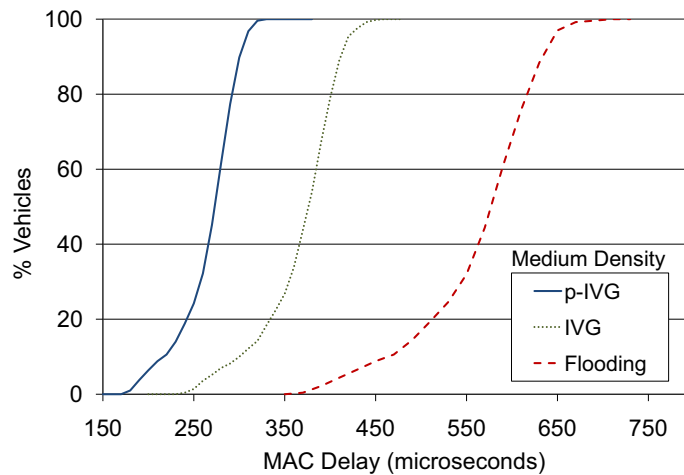
FIG. 67: CDF of MAC Delay for Beacon and Dummy Frames at Medium Density

VII.4.1 MAC Delay

In order to investigate the impact of using p-IVG, we measured the amount of MAC delay for each frame. In Figures 66-68, we show the CDFs of the MAC delay experienced by beacons and dummy frames for the three rebroadcast schemes (flooding, IVG, and p-IVG) at low, medium and high traffic densities.

As expected, beacon and dummy frames see the smallest MAC delay for all densities when p-IVG is used and the largest delay when basic flooding is used. Although beacon frames are not rebroadcast, they still benefit from the reduced contention
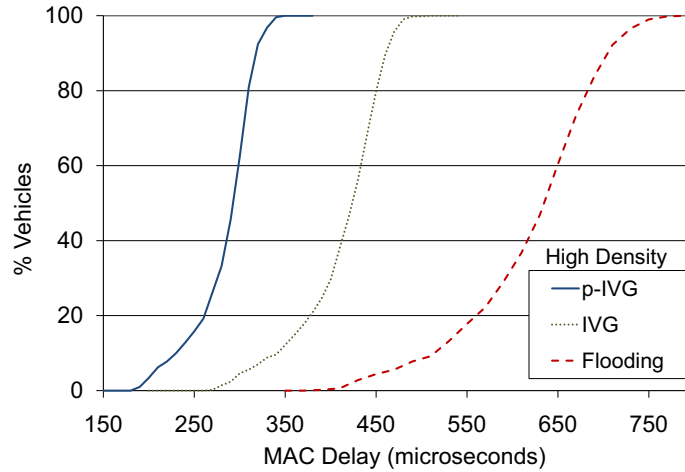
FIG. 68: CDF of MAC Delay for Beacon and Dummy Frames at High Density
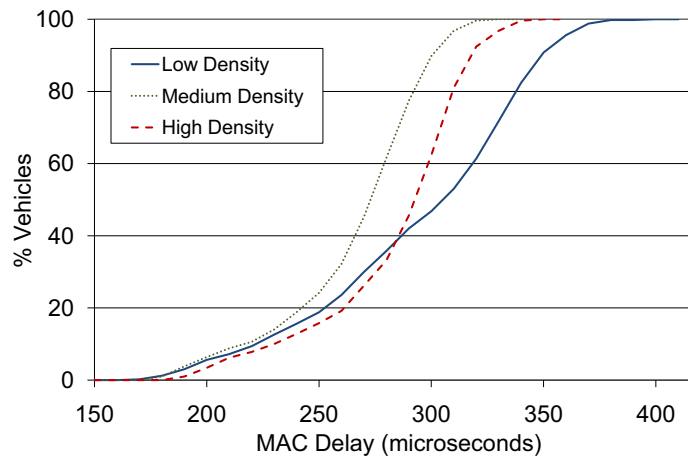
FIG. 69: CDF of MAC Delay for Beacon and Dummy Frames with p-IVG

when p-IVG is used. p-IVG is also shown to be an improvement upon standard IVG. This is because p-IVG limits the number of nodes that may be rebroadcasting frames at the same time, thus making the wireless channel more available.

To investigate how density affects p-IVG, we put the CDFs of MAC delay for p-IVG on one graph in Figure 69. As p-IVG adapts its behavior according to the vehicular density, its rebroadcast behavior is almost steady with a small variation. The MAC delay in Figure 69 reflects this stable behavior, because it shows that as the density varies, the MAC delay does not vary much.
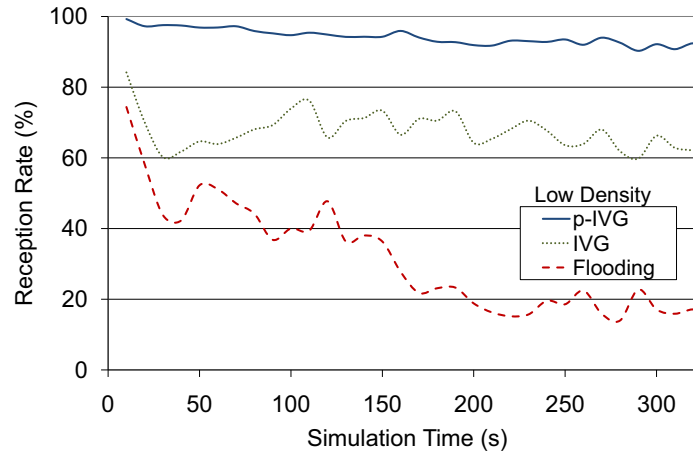
FIG. 70: Reception Rate for Beacon and Dummy Frames at Low Density
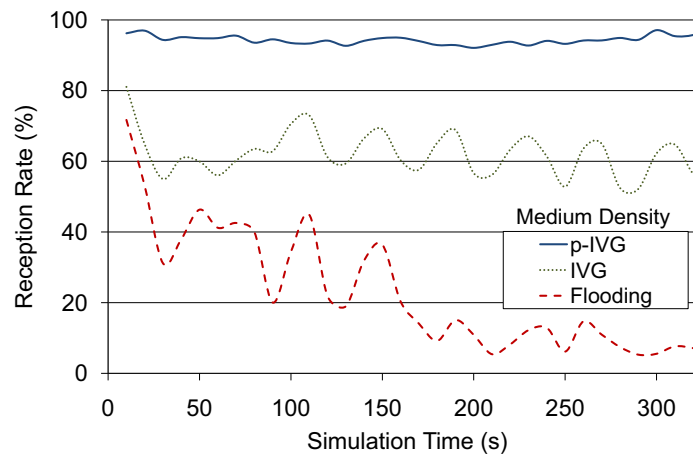
FIG. 71: Reception Rate for Beacon and Dummy Frames at Medium Density

VII.4.2 Reception Rate

To assess the impact of wireless collisions, we measured the average reception rate over time. If a frame was transmitted and no other vehicle received it, then the frame was considered to not be received. Either the frame experienced a collision, or no other vehicle was within 300 m of the sender. Figures 70, 71 and 72 show the reception rates for p-IVG, IVG, and flooding with low, medium and high traffic densities, averaged every 10 seconds.

IVG improves greatly over flooding, but p-IVG results in over 90% reception rate. Figure 73 shows the reception rate for p-IVG with all densities. In our definition of
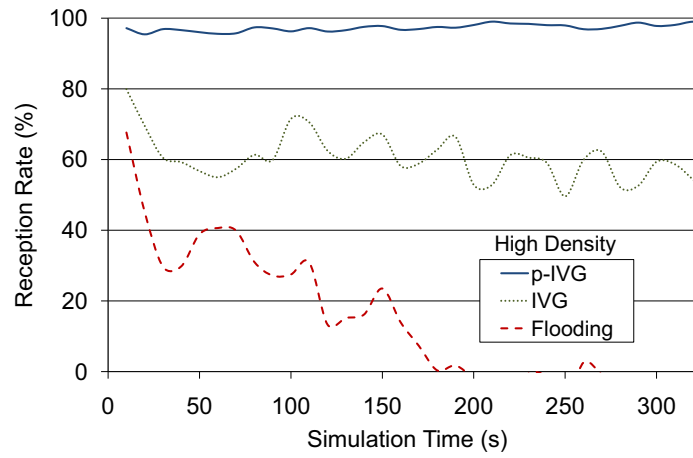
FIG. 72: Reception Rate for Beacon and Dummy Frames at High Density
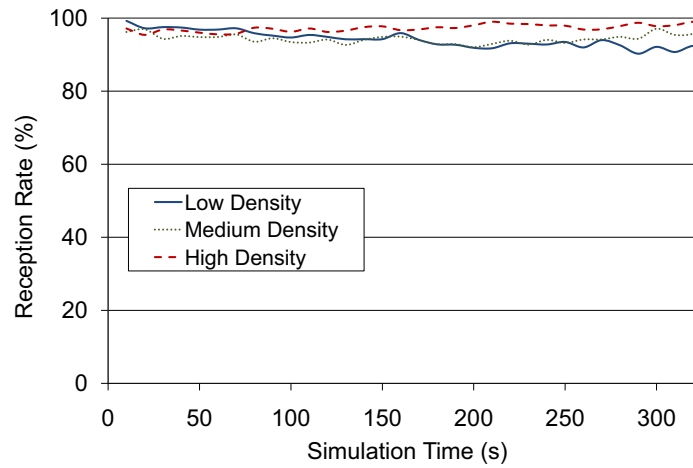
FIG. 73: Reception Rate with p-IVG

reception, if just one vehicle receives the frame, it is considered to be received. In Figure 73 high density traffic provides the best reception rate because there are more vehicles in range, thus there is a higher chance for a vehicle to receive the frame. The decreasing reception rate with low density is an artifact of how vehicles enter the simulation. Vehicles enter the highway with a low speed and gradually increase towards the maximum of $30\,\mathrm{m/s}$. Until the last vehicle reaches the maximum speed, vehicles in front of it will be traveling faster, thus moving out of its transmission range. Figure 74 illustrates the initial disconnection situation that may occur when a vehicle enters the highway, especially at low density. In Figure 74-a, the distance between the vehicle that just entered the highway and the very first vehicle ahead is greater than $300\,\mathrm{m}$ that makes it disconnected. Then after the vehicle gains more speed the distance between them gets reduced until it becomes less than $300\,\mathrm{m}$ so that the vehicle gets connected as we see in Figure 74-b. For low density, it takes longer for 500 vehicles to enter the system than with high density, thus it takes longer (in simulation time) for the last vehicle to reach $30\,\mathrm{m/s}$.

To further demonstrate the problems we found with standard IVG, we show in Figure 75 the reception rate per second when using IVG at medium traffic density. Each drop spike in the reception rate corresponds to a situation where multiple vehicles at the boundary rebroadcast frames at the same time, resulting in the spatial broadcast storm problem.

VII.4.3 Backoff Percentage

To measure the impact of using p-IVG on reducing the channel contention, we calculate the backoff percentage. In 802.11, when a node is ready to send a frame, it first listens on the channel. If the channel is busy, the node backs off and tries to send later. So the backoff percentage is the ratio between the total number of backoffs to the total number of sending attempts, averaged over all of the vehicles. As channel contention increases, the backoff percentage increases. Table 11 summarizes the backoff percentage for flooding, IVG, and p-IVG at low, medium, and high densities. Using p-IVG results in a dramatically lower backoff percentage than either flooding or IVG. This means that channel contention when using p-IVG is much lower than either of the other cases.
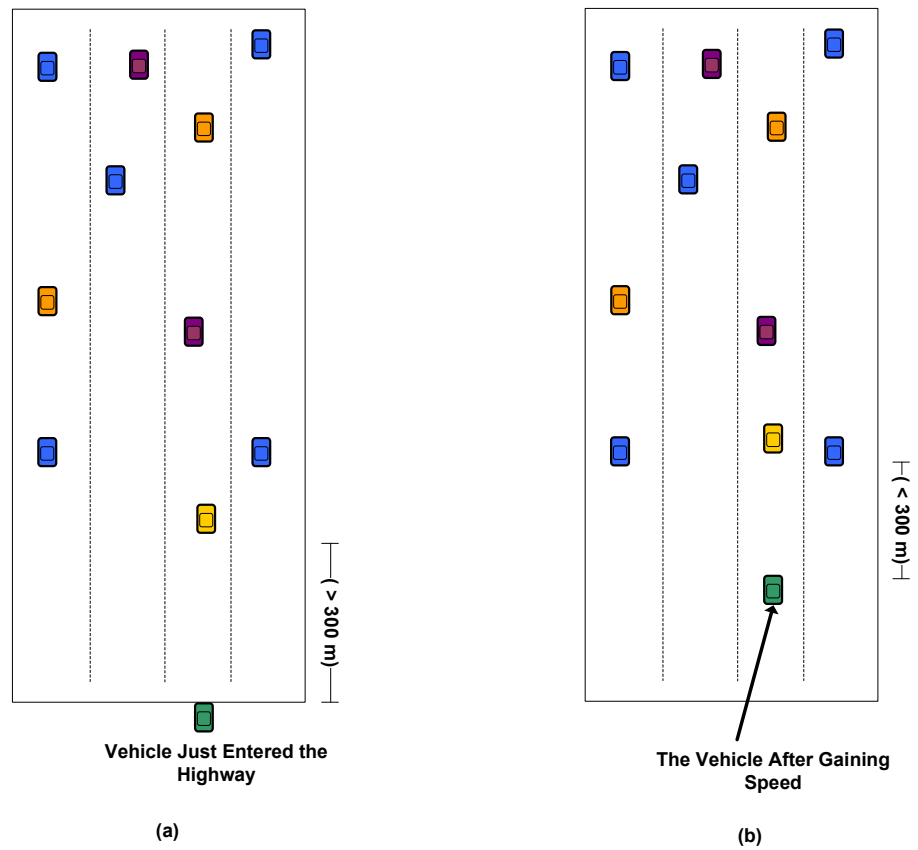
FIG. 74: Initial Disconnection for Vehicles Recently Entered the Highway
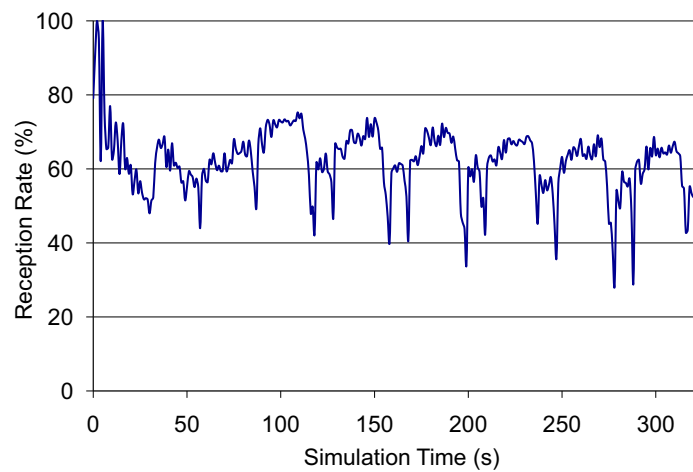
FIG. 75: Reception Rate with IVG at Medium Density

TABLE 11: Backoff Percentage

	Backoff Percentage		
	Low Density	Medium Density	High Density
p-IVG	7.05%	4.32%	5.94%
IVG	31.09%	35.94%	42.67%
Flooding	80.83%	84.38%	92.11%
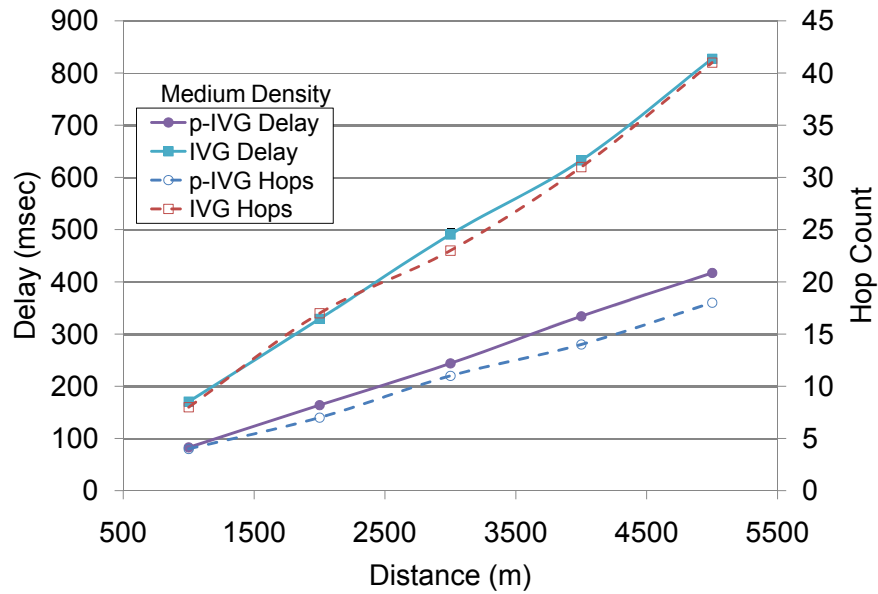
FIG. 76: Dissemination Delay and Hop Count at Medium Density

VII.4.4 Dissemination Delay and Hop Count

The main goal of any dissemination technique is to carry the data as far as possible in the shortest amount of time, using the minimum number of hops. To assess the impact of using p-IVG on achieving this goal, in Figure 76 we show the average time a frame takes to travel a certain distance (dissemination delay) and the number of hops it visits along this trip (hop count) for p-IVG and IVG. p-IVG takes much less time and fewer hops to disseminate the data than IVG, while flooding failed to disseminate the data beyond 1 km and so was not pictured on the graph. With p-IVG, data was able to be disseminated up to 5 km in less than 500 ms.

VII.4.5 Redundancy Factor

To disseminate a certain message using any of the flooding, IVG and p-IVG techniques, this message must be initially broadcast and then rebroadcast again, after applying the rebroadcast criteria in case of IVG and p-IVG or blindly rebroadcast in the case of flooding. As a result of rebroadcasting the message, it may be received multiple times by some vehicles. These duplicate copies of the message are considered redundant. The lower the number of duplicate copies the dissemination technique generates, the better the technique is.

To measure the redundancy amount that each technique introduces, we modified the simulation so that each vehicle will maintain two counters. One to count the number of unique messages received, $Unique\_Message\_Counter$, and another one to count the number of redundant (duplicate) messages received, $Redundant\_Message\_Counter$. Also, the vehicle behavior upon receiving a message has been changed to do the following:

- calculate the message signature - simply the hash of the message contents after excluding the variable components that change at each rebroadcast

- search the list of signatures of the previously received messages for a match - if found, increment the $Redundant\_Message\_Counter$, otherwise add the signature to the list and increment the $Unique\_Message\_Counter$

The redundancy factor, $Redundancy\_Factor$, is calculated from the above two counters using this equation

$$Redundancy\_Factor = \frac{Redundant\_Message\_Counter}{Unique\_Message\_Counter}.$$

For any dissemination technique, the smaller the $Redundancy\_Factor$, the better this technique is, because it introduces less redundant data. In the optimal situation, the dissemination technique should introduce no redundant data and hence have the smallest possible $Redundancy\_Factor$, which is zero. But realistically, it has been proven in [13] that the lower bound for the redundancy factor for any dissemination technique is 0.4.

Table 12 summarizes the redundancy factor for IVG and p-IVG at low, medium, and high densities. p-IVG has smaller redundancy factor values than IVG at all densities, $i.e.$, at medium density, p-IVG has a redundancy factor equal to 0.602,

TABLE 12: Redundancy Factor

	Redundancy Factor		
	Low Density	Medium Density	High Density
p-IVG	0.547	0.602	0.487
IVG	1.39	1.934	2.333

which means for each 10 unique messages it disseminates, it also disseminates 6 redundant messages. At the same density, the redundancy factor for IVG is 1.934, which means it disseminates 19 redundant messages for each 10 unique messages it disseminates. The reason that p-IVG introduces less redundant data than IVG is the probabilistic nature for its rebroadcast criteria, while the rebroadcast criteria in case of IVG is always deterministic.

Also, we notice that in case of IVG as the density increases, the redundancy factor increases because in IVG the number of message rebroadcasts is proportional to the vehicular density. While, in p-IVG, the density increase does not impact the redundancy factor because p-IVG adapts its rebroadcast behavior according to the density.

VII.4.6 Coverage Percentage

The coverage percentage is defined as the percentage of vehicles that actually received a certain message or one of its rebroadcasts with respect to the intended recipients of the original message. The intended recipients are the vehicles within the transmission range (300 m) of the original message sender at the transmission time. The coverage percentage is a key metric to compare between data dissemination techniques because the main goal of these techniques is to deliver data to the maximum reachable vehicles, which is exactly what this metric is measuring but within the scope of the transmission range. The higher the coverage percentage for certain dissemination technique, the more successful this technique is.

The process of calculating the coverage percentage for p-IVG and IVG is an offline process. The following are the steps required to calculate the coverage percentage metric for each message sent (broadcast, not rebroadcast):

- identify the sender location (X, Y) and the message timestamp

TABLE 13: Coverage Percentage

	Coverage Percentage		
	Low Density	Medium Density	High Density
p-IVG	89%	92%	91%
IVG	94%	96%	97%

- identify the intended recipients, all vehicles within the transmission range of the sender at this moment

- search the log file for each vehicle within the intended recipients list identified in the previous step looking for the original message signature (all the original message's rebroadcasts will have the same signature as the original message)

- calculate the coverage percentage for this message

- update the overall coverage percentage

Table 13 summarizes the coverage percentage for IVG and p-IVG at low, medium, and high densities. As we see in Table 13, although p-IVG and IVG have the same transmission range, they have different coverage percentages. There are two reasons behind this. First, p-IVG and IVG have different rebroadcast criteria. The second reason is how we declare certain message as received by one of the intended recipients. For a message to be marked as received by one of intended recipients vehicles, this vehicle should receive the original message or any of its rebroadcasts. Based on the previous definition, the aggressive dissemination techniques that rebroadcasts more often will have higher coverage percentage than any other conservative dissemination techniques that usually rebroadcasts less often. Recalling the rebroadcast criteria for both p-IVG and IVG, we can categorize p-IVG as a conservative dissemination technique. While, on the other side IVG can be categorized as an aggressive dissemination technique. The second reason not only explains why p-IVG and IVG have different coverage percentages, but also explains why IVG has higher coverage percentage than p-IVG.

One more observation on the results in Table 13, although the scope that we are measuring the coverage percentage within is only 300 m, neither p-IVG nor IVG have full, 100% coverage. We can summarize the reasons in the following points:

- The number of intended recipients is calculated based on the maximum transmission range 300 m, while the effective transmission range is only 250 m due to signal fading, which leaves the vehicles at the outer boundary mostly not reachable

- In addition to signal fading, signal interference has also an impact on getting the signal more weak and distorted and hence leaving the vehicles at the outer boundary mostly not reachable

- Collisions are a major reason for less than 100% coverage especially in case of IVG

The previous two metrics, coverage percentage and redundancy factor, are trade-offs metrics, so that it is hard to develop a dissemination technique that performs well at both. As we have seen in the previous two experiments, IVG is better than p-IVG with respect to coverage percentage, while p-IVG showed to be better than IVG with respect to the redundancy factor, which makes it hard to prefer one over the other. The following experiment combines the previous two metrics together to be able to compare p-IVG to IVG. In this experiment, we extend the coverage percentage experiment so that we will not only mark each message as being received or not, but also will count how many times this message has been received.

From the experiment results, shown in Figures 77 and 78, we can conclude that although IVG provides better coverage percentage than the percentage of vehicles that receive, it injects more redundant data than what p-IVG does. In addition to the previous conclusion that Table 12 and 13 confirm its correctness, Figures 77 and 78 provide us with more insight on the redundancy trend with density change.

Figures 77 and 78 show that a larger percentage of vehicles will miss receiving a message with p-IVG than with IVG. This matches with the coverage percentage shown in Table 13. As mentioned, there is a tradeoff between coverage and redundancy.

In Figure 78 with p-IVG, we notice that as the traffic becomes more dense, the percentage of vehicles that receive only one copy increases. The percentage of vehicles that receive more than one copy decreases. So, p-IVG behavior adapts itself towards the optimal behavior as the vehicular density increases.

Figure 77 shows the opposite conclusion in case of IVG, as we notice that as the traffic becomes more dense, the percentage of vehicles that receive more copies
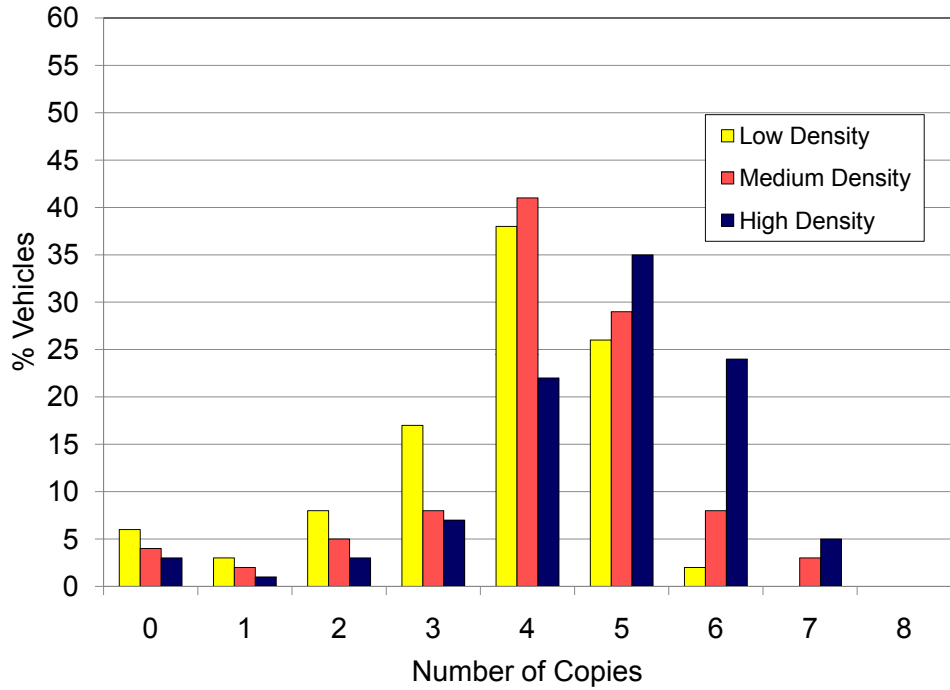
FIG. 77: Average Number of Copies Received per Message in Case of IVG
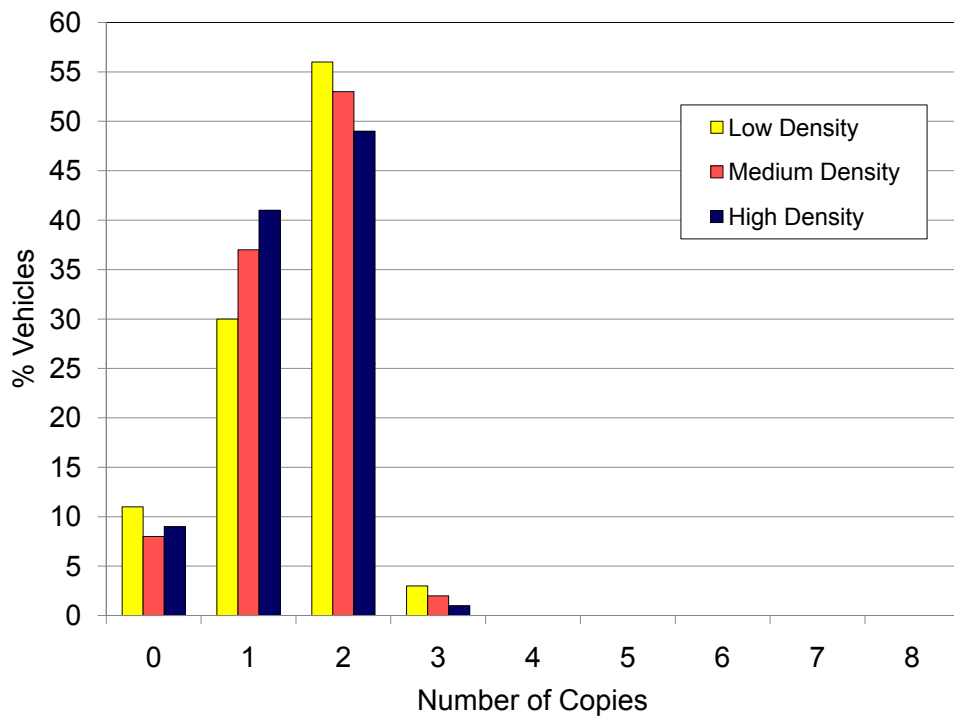
FIG. 78: Average Number of Copies Received per Message in Case of p-IVG

increases. The percentage of vehicles that receive fewer copies decreases. IVG behavior is moving towards worse redundancy as the vehicular density increases. The problem in using a dissemination technique that injects more redundant data is that it consumes more bandwidth, which limits the number of applications that can share the media with it.

VII.5   ON-DEMAND   VEHICULAR   GAP-BRIDGING   (OD-V-GB) EVALUATION

In this section, we concentrate on evaluating OD-V-GB to find how fast it will react to the discontinuity problem when it occurs and what will be the length of the recovered extended view. To obtain these evaluation metrics, we will conduct three experiments, each one of them adopting a different scenario. The first experiment adopts the scenario where there is no discontinuity problem. The scenario adopted by the second experiment is that a discontinuity problem occurs, but OD-V-GB is not being used. The third experiment adopts the scenario where a discontinuity problem occurs and OD-V-GB is used.

The same simulation settings in Table 8 will be used. In addition, the vehicular density for both the same and opposite directions will be medium (66 vehicles/km). The $diff\_percentage\_thr$ will be 16%, and the $matching\_percentage\_thr$ will be 74%. Both of these values were calculated in Chapter IV.

As shown in Figure 79, we have a bi-directional road, and direction 1 is our regular direction while direction 2 is the opposite direction. Our target vehicle, the vehicle that we will focus on its extended view to see the impact of using the proposed OD-V-GB technique, will be the last vehicle entering the road (from direction 1) so that we can control its behavior and movement precisely. For each of the three experiments, we will monitor the impact on the extended view length of the target vehicle.

In all the figures in the following sections, the X-axis does not represent the simulation time, but represents the elapsed time for the target vehicle in the simulation. Also, for all the experiment result figures, except Figures 83, 88, 86 and 90, each point in the curve represents the average extended view length for the previous 10 sec.
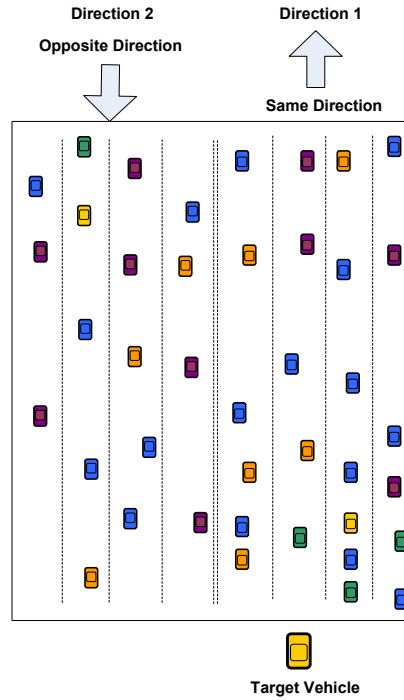
FIG. 79: Highway Scenario for On-Demand Vehicular Gap-Bridging Evaluation

VII.5.1 Scenario 1: No Discontinuity Problem

During this experiment we will monitor the extended view length for the target vehicle assuming that there is no discontinuity problem. To guarantee that this assumption will be maintained during the experiment lifetime, we will keep the distance between the target vehicle and the first vehicle ahead of it less than 250 m. The target vehicle will be connected all the time and hence will be able to receive primary and aggregated frames continuously.

Figure 80 shows the target vehicle's average extended view length. We can clearly distinguish between two sections in this curve, the first section represents the transient period, in which the extended view is growing. During the transient period, the target vehicle will receive a continuous stream of aggregated frames. As the data in these frames is validated, the average extended view length will continue to increase till it reaches a saturation point at time 176 sec. The average extended view length at this point is 17 km. After this point, the second section of the curve starts in which the average extended view length is mostly stable and hence we can see the curve almost saturates around length 17 km for the extended view. During the second section of the curve, the target vehicle will receive the same stream of aggregated
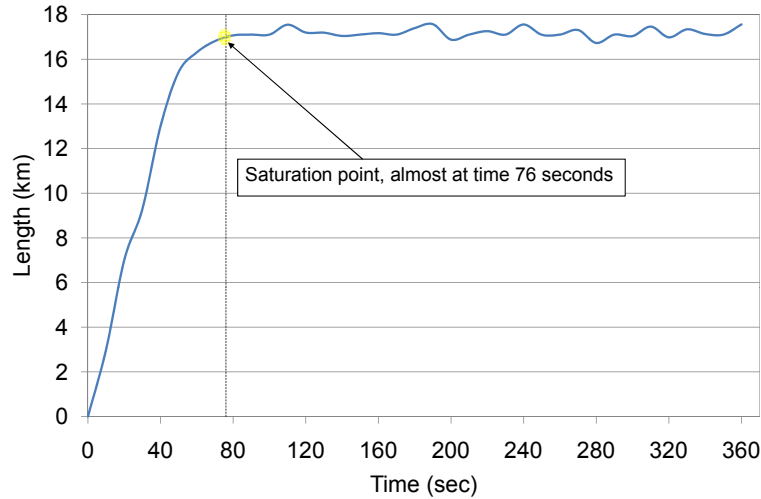
FIG. 80: Average Extended View Length: No Discontinuity Problem

frames that it did during the time period of the curve first section. But the increase in the average extended view length due to the received aggregated frames will be equal to the decrease in the average extended view length due to purging the expired frames. This results in the average extended view length curve not growing beyond this limit.

VII.5.2 Scenario 2: Discontinuity Problem without OD-V-GB

The experiment conducted in this section studies the impact of the discontinuity problem on the length of the extended view for the target vehicle. The experiment consists of two steps, the first step is exactly the same as the experiment in Section VII.5.1, considered as the warm up for our experiment in this section. During this step the target vehicle will be controlled to stay connected till its extended view gets saturated. In the second step of the experiment, the target vehicle will be forced to decelerate so that the distance to the first vehicle ahead becomes greater than 250 m and hence the target vehicle gets disconnected. As the target vehicle gets disconnected, it will not be able to receive any primary or aggregated frames from the vehicles ahead. At the same time, the frames composing its extended view will expire and get purged from the system and hence its extended view will diminish till its length becomes zero.

Figure 81 shows the average extended view length. As we mentioned before, the first step in this experiment is similar to the experiment in Section VII.5.1. As the
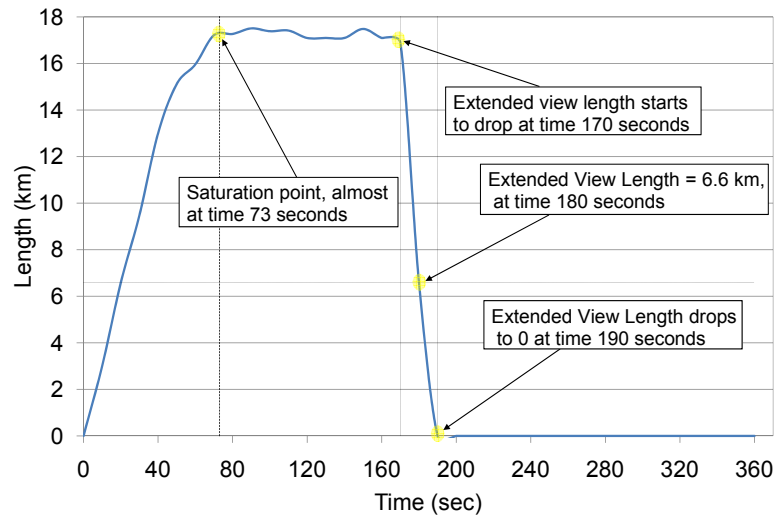
FIG. 81: Average Extended View Length: Discontinuity Problem
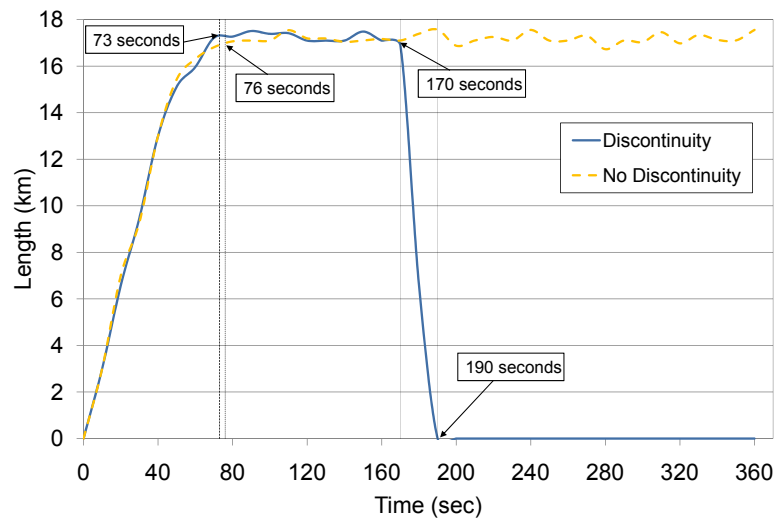
FIG. 82: Comparing the Average Extended View Length in Case of No Discontinuity vs. Discontinuity at Medium Density

first step ends at time 170 sec, the similarity ends at this point then the two scenarios start to differ, especially after the target vehicle starts to decelerate. In Figure 82, we combine the results of both the discontinuity problem experiment (Figure 81) and no discontinuity problem experiment (Figure 80) to make it easier to compare them. The curve in Figure 81 represents the average extended view length for the target vehicle during its life-time in the simulation, we can distinguish between four different sections in that curve. The first section starts at time 0 sec and ends at time 73 sec. The curve trend in this section is similar to the curve trend in the first section of Figure 80, as in both the extended view length for the target vehicle is growing. The behavior similarity applies to the second section of both curves as we see in Figure 82 till time 170 sec where the target vehicle starts to decelerate.

The third section of the curve starts at time 170 sec and ends at time 190 sec. At the beginning of this section, the target vehicle starts to decelerate and gets disconnected from the vehicles ahead as the distance grows to be greater than 250 m and hence it stops receiving any frames (primary and aggregated frames) from them. As a consequence for not receiving any of these frames, the extended view stops growing. The length of the extended view starts to drop as the aggregated frames that compose the extended frames expire and are purged from the system. Figure 83 shows how the extended view length drops to zero. Also, we notice the following from the graph:

1. At each second there are two values for the extended view length

2. The extended view length drops to almost zero at time 180 sec, 10 seconds from the time at which the target vehicle started to decelerate

3. In the detailed graph (Figure 83) we see that the extended view drops to zero at time 180 sec, while in the overall graph (Figure 81) it reaches zero at time 190 sec

Regarding the first observation, the check for the aggregated frames expiration and purging the expired frames is performed each second and hence any drop or shrinking in the extended view length should take effect only at the second boundary. So, at each second we can see two values for the extended view length, one before doing the expiration check and purging the expired frame and one after. We should note that updating the extended view upon receiving aggregated frames is performed
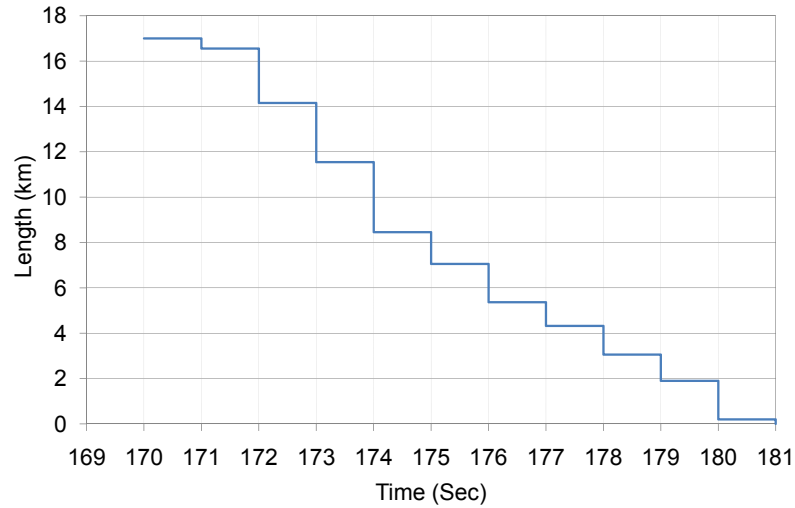
FIG. 83: Average Extended View Length - Section 3 Details: Discontinuity Problem

immediately, and we do not accumulate and process them at the second boundary for the following reasons:

- The volume of the received aggregated frames is an order of magnitude more than the volume of the purged frames

- The percentage of frames that pass the various checks (TTL check, frames overlapping check and matching percentage check) is small

So, based on the above two reasons, storing the received frames and processing them each second will waste storage space and processing time (read/write time). Additionally, updating the extended view with newly received frames is more important than checking for the frames expiration and purging them.

According to the logic implemented in this scenario, the target vehicle should not have received any aggregated frames after the time 170 sec, and the last aggregated frame received by the target vehicle should be on or just before the time 170 sec. Because the new threshold for the aggregated frames is 10 sec, the most recently received aggregated frame, and any other aggregated frames that have been received before it, should expire and get purged from the system on or before time 180 sec. Hence, the extended view length should drop to zero on or before time 180 sec, which explains the second observation.

For the third observation, the detailed graph in Figure 83 shows the actual extended view length at each second while in the overall graph in Figure 81, each point

in the curve represents the average extended view length for the previous 10 sec. So as we see in the detailed graph, the extended view length drops gradually over 10 seconds, starting at time 170 sec till it reaches zero at time 180 sec. The average value for the extended view length over these 10 seconds is 6.6 km, which is the value for the extended view length in the overall graph at time 180 sec. Then for the subsequent 10 seconds starting from time 180 sec till time 190 sec, the extended view length is almost zero in the detailed graph, resulting in having an average value equal to zero that shows up in the overall graph at time 190 sec.

The explanation for the fourth section of the curve is obvious as after the extended view length reaches zero and as the target vehicle is disconnected, it will not receive any frames, so its extended view will continue at the zero level till the end of the simulation.

VII.5.3   Scenario 3: Discontinuity Problem with OD-V-GB

The scenario in this section is similar to the scenario in section VII.5.2, but the vehicles in this scenario will be equipped with OD-V-GB technique that will start once a discontinuity is detected. So, the main focus of the experiments conducted in this section is to measure the impact of using OD-V-GB technique on the extended view length in case of discontinuity.

Similar to the scenario in section VII.5.2, the target vehicle will stay connected till its extended view length gets saturated then it will be forced to decelerate till the distance to the first vehicle ahead becomes greater than 250 m and hence it will be disconnected. In contrast to the scenario in section VII.5.2, the OD-V-GB technique will start to operate actively to bridge the discontinuity gap once it detects the discontinuity using its *Broadcasting GBR Messages Module*. The OD-V-GB technique will continue operating actively till the target vehicle gets connected again with the vehicles ahead. Our experiment does not cover this case as we allow the OD-V-GB technique to take over for the whole simulation time.

Figure 84 shows the results in case where the vehicular density in the opposite direction is medium. Figure 84 shows the extended view length of the target vehicle for the whole simulation lifetime. In this figure we distinguish between five sections in the curve. Due to the similarity in the scenarios in Section VII.5.2 and Section VII.5.3, the corresponding results for the first and second section of the curve will be similar, as we see in Figure 85. The similarity ends at time 170 sec when the
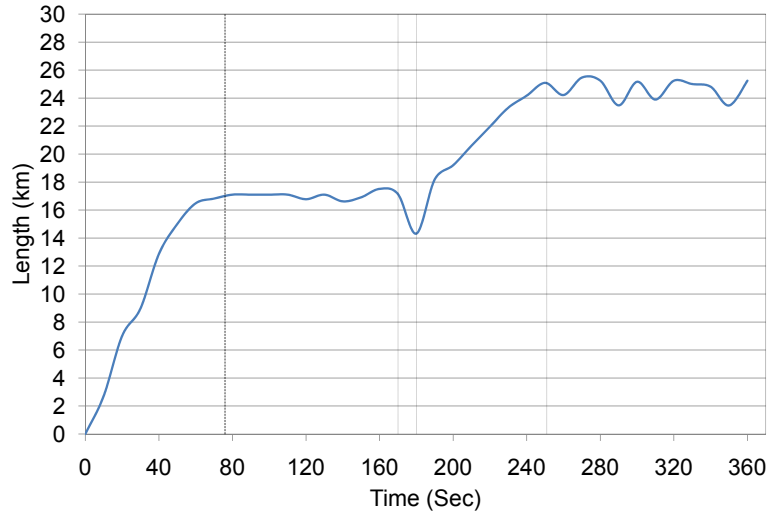
FIG. 84: Average Extended View Length: Discontinuity Problem with OD-V-GB at Medium Density in Opposite Lane

discontinuity occurs and in our current scenario the OD-V-GB technique will start to operate.

As OD-V-GB starts to operate, the vehicles in the opposite direction will forward the aggregated frames in their repository when they get in contact with the first vehicle after the gap, in our case the target vehicle. Hence, the target vehicle will utilize these aggregated frames to build another extended view called the OD-V-GB extended view. In this case, the target vehicle will maintain two extended views, one is the original extended view that will keep decreasing in length as the target vehicle continues to be disconnected and the other is the OD-V-GB extended view that keeps increasing in length as the target vehicle continues to receive aggregated frames from the vehicles in the opposite direction. Although the target vehicle will be maintaining these two views, at any certain point of time, it will be pointing to only one of them as its current view. The view that will be selected to be the current view is the one with the longest extended view length. Intuitively, the current view for the target vehicle will be initially pointing to the original extended view. Then as its the length decreases and the length for the OD-V-GB extended view increases, the target vehicle will switch its current view to point to OD-V-GB extended view. How soon this switch point is reached and how fast the OD-V-GB extended grows afterwards depends on the vehicular density in the opposite direction as we will see in Section VII.5.4.
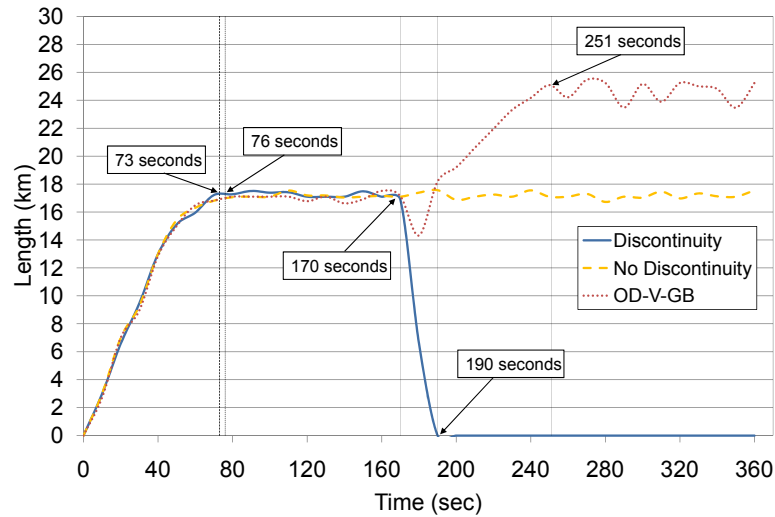
FIG. 85: Comparing the Average Extended View Length in Case of No Discontinuity vs. Discontinuity vs. OD-V-GB at Medium Density

The third section of the curve in Figure 84 starts at time 170 sec and ends at time 180 sec. During this section the switching point occurs, as at its beginning, the current view for the target vehicle will be pointing to the original extended view then it switches to point to the OD-V-GB extended view. Figure 86 shows the details for this section of the curve. As we see in the figure, the curve has two main parts, the step-wise decreasing part and the smooth increasing part. The step-wise decreasing part of the curve starts at time 170 sec and ends at time 173 sec, and it represents the time period at which the original extended view was the current view for the target vehicle. The reason for the step-wise shape for this part of the curve is that the aggregated frames composing the original extended view will be expiring and purged out of the system and this check is being performed at each second boundary which results in this step-wise shape. The time 173 sec represents the switching point for the current view from pointing to the original extended view to the OD-V-GB extended view, because this point is the last point at which the original extended view has longer length than the OD-V-GB extended view. The smooth increasing part of the curve starts at time 173 sec and represents the time period at which the OD-V-GB extended view was the current view for the target vehicle. The reason for the smooth increasing shape is that the target vehicle will be receiving aggregated frames forwarded from the vehicles in the opposite direction. According to extended view updating rules, those frames will be processed immediately and hence the update to

FIG. 86: Average Extended View Length - Section 3 Details: Discontinuity Problem with OD-V-GB at Medium Density in Opposite Lane

the extended view length will be smooth.

The fourth section of the curve in Figure 84 is a natural extension to the third part of the curve in Figure 86, the smooth increasing part. As the target vehicle continues receiving aggregated frames from the vehicles on the opposite direction, its OD-V-GB extended view continues to grow. Because the $matching\_percentage\_thr$ constraint is not being applied on those received aggregated frames, the length of the OD-V-GB extended view exceeds the 17 km limit (see Section IV.6.1 for details). The new limit for OD-V-GB extended view length will be 26 km, which is the theoretical maximum extended view length limit (see Section III.3.4). At the end of this section, at time 251 sec, the curve reaches its maximum growth point and starts to saturate.

The fifth section of the curve starts at time 251 sec and lasts till the simulation ends. In this section, the length of OD-V-GB extended view almost saturates around the value 25.14 km, and the main observation in this section is the curve fluctuation. The fluctuations are not large in number, but in their values, as we can notice deep drops and high rises in the curve. The reason for this behavior is the following, according to OD-V-GB technique each vehicle coming in the opposite direction will forward all the aggregated frames in its repository, which is 2 sec worth of aggregated frames. As the target vehicle will receive a set of aggregated frames that represent much information about the road within a short period of time, this will boost its OD-V-GB extended view length and cause these high rises in the curve. As a

consequence of receiving much information about the road within a short period of time in a small set of aggregated frames, those aggregated frames will also expire in a short period, resulting in the deep drops in the curve, unless another vehicle from the opposite direction supplements it with its aggregated frames repository. As the target vehicle gets in contact with more vehicles coming in the opposite direction, it will get almost a continuous stream of aggregated frames repositories and hence the fluctuations in its extended view length will be much less.

VII.5.4 The Impact of Varying the Vehicular Density in the Opposite Direction

To study the impact of varying the vehicular density in the opposite direction, we conducted two experiments similar to the one in Section VII.5.3. The only change in these experiments is the vehicular density in the opposite direction. We set the vehicular density for the opposite direction to be low in one experiment and high in the other, as the medium density case has been already covered in the experiment in Section VII.5.3. Figures 87 and 89 show the results for these two experiments. Figures 88 and 90 show the details for the third section of the OD-V-GB extended view length curve for each experiment. To be able to compare the experiment results under different vehicular density, low, medium and high density, we combined all three experiments' graphs in Figures 91 and 92.

Using the experiments' graphs, we can notice the following observations:

- Figure 92 shows the details of the third section of the OD-V-GB extended view length curve in case of low, medium and high vehicular density. The original extended view length, which is represented by the first part (the step-wise decreasing part) of the curve, drops to its lowest value, 8.2 km, in case of low density. The drop is much less in the case of high density, as it reaches 11.7 km. In the case of medium density, its value is 12.5 km.

- The switching point in the case of high and medium density occurs at time 173 sec, while it gets delayed in case of low density till time 175 sec.

- The growth rate for the OD-V-GB extended view length is represented by the second part (the smooth increasing part) of the curve that increases as the vehicular density increase. It grows very fast in case of high density, while it has the smallest rate in case of low density.
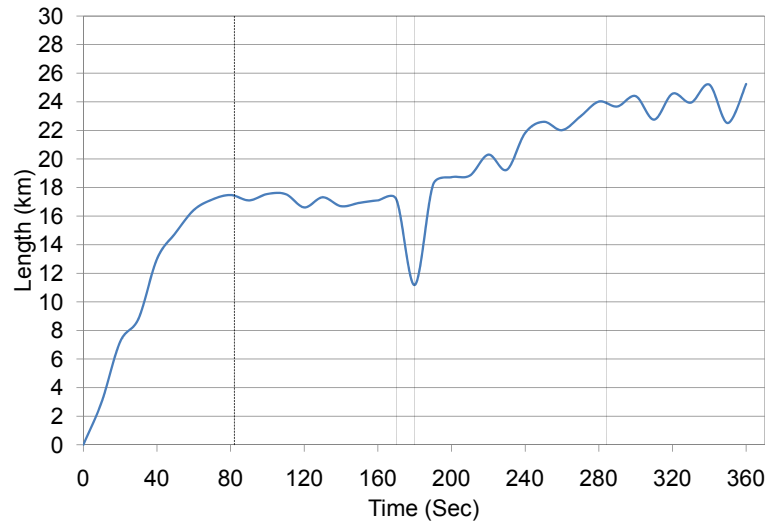
FIG. 87: Average Extended View Length: Discontinuity Problem with OD-V-GB at Low Density in Opposite Lane
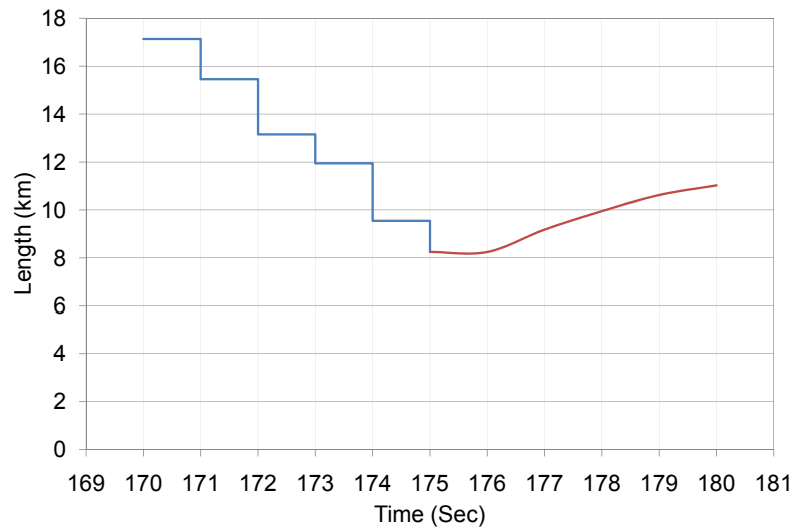
FIG. 88: Average Extended View Length - Section 3 Details: Discontinuity Problem with OD-V-GB at Low Density in Opposite Lane
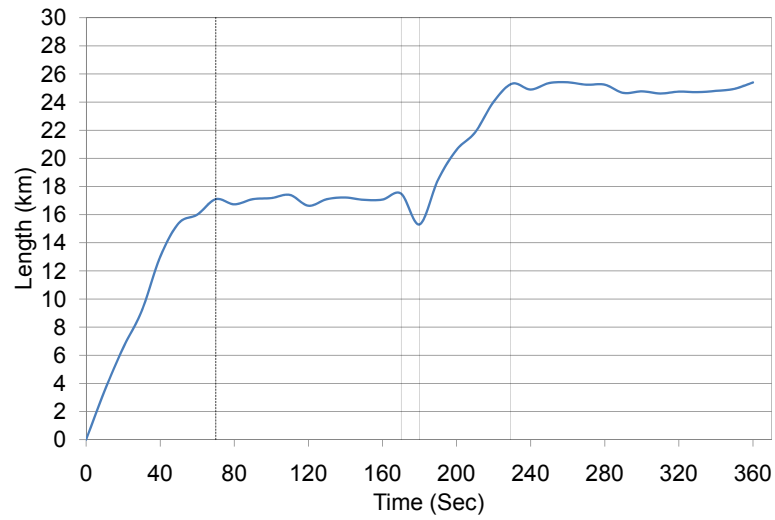
FIG. 89: Average Extended View Length: Discontinuity Problem with OD-V-GB at High Density in Opposite Lane

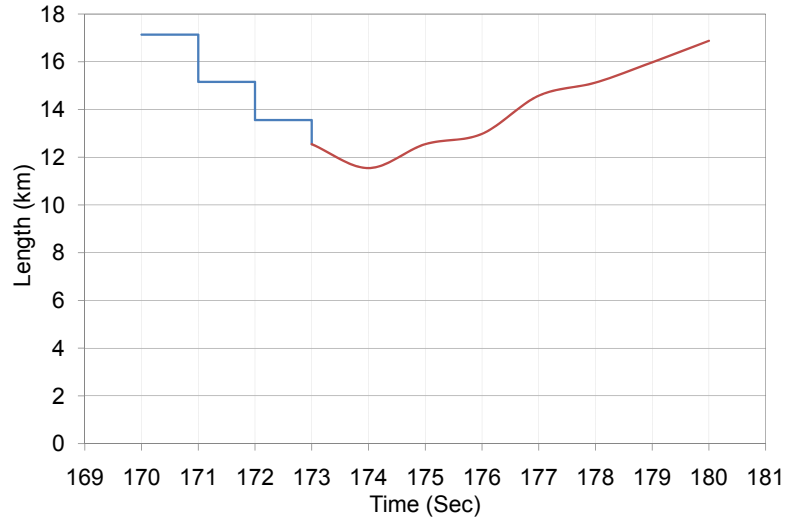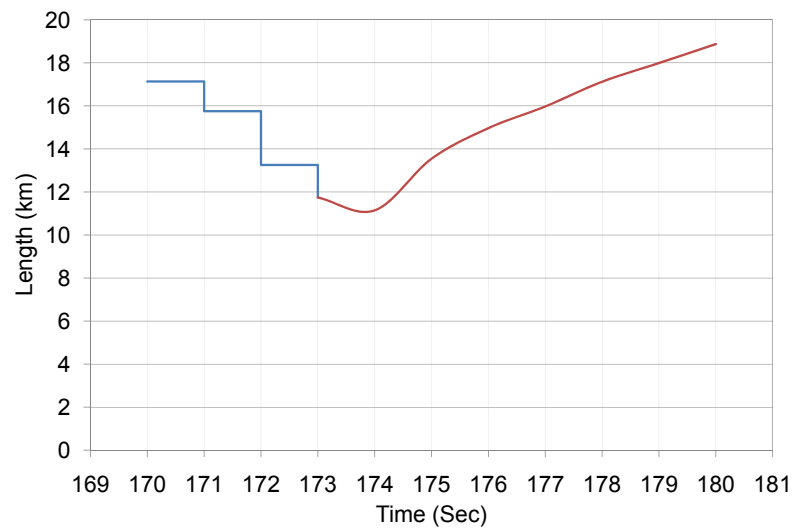
FIG. 90: Average Extended View Length - Section 3 Details: Discontinuity Problem with OD-V-GB at High Density in Opposite Lane
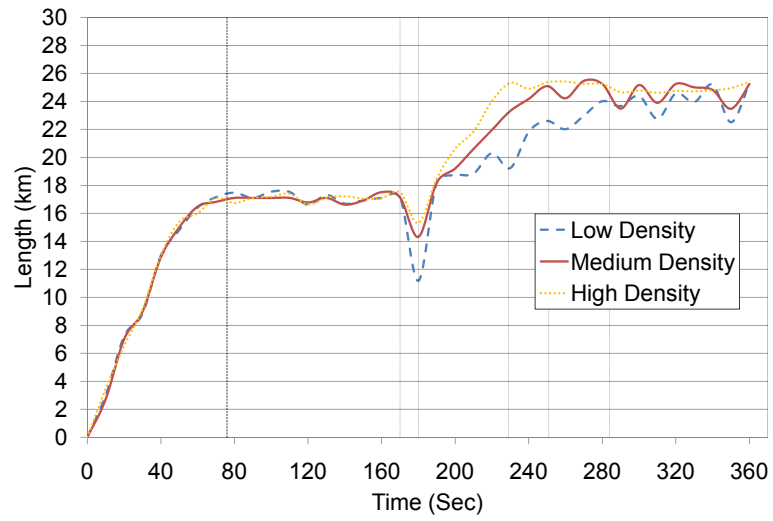
FIG. 91: Average Extended View Length: Discontinuity Problem with OD-V-GB Considering Low, Medium and High Density in Opposite Lane
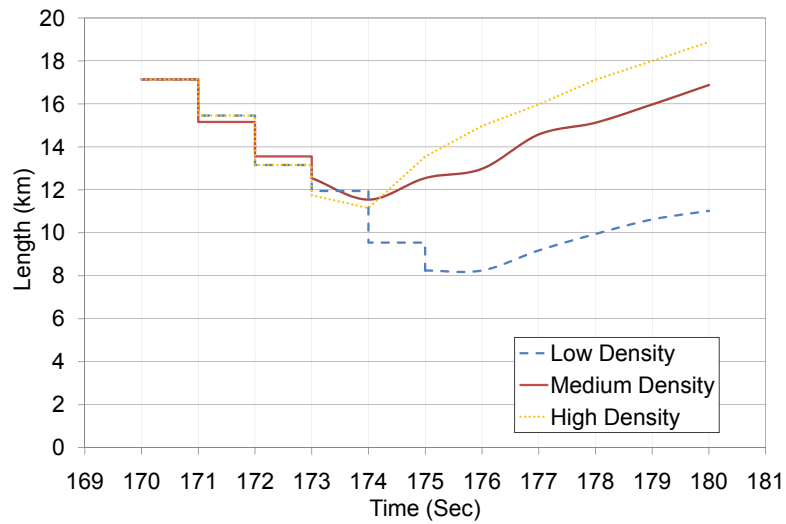
FIG. 92: Average Extended View Length - Section 3 Details: Discontinuity Problem with OD-V-GB Considering Low, Medium and High Density in Opposite Lane

- The growth rate for the OD-V-GB extended view length that we have seen in the previous point continues with the same behavior till it reaches the saturation point, as shown in fourth section of the curves in Figure 91. In case of high density, the OD-V-GB extended view length grows very fast and reaches the saturation point at time 229 sec, earlier than the other two cases. The low density case has the slowest growth rate for the OD-V-GB extended view length and reaches the saturation point the last at time 251 sec. The saturation point is reached at time 284 sec in case of medium density.

- In the fifth section of the curve in Figure 91, the higher the vehicular density in the opposite direction, the fewer the number of fluctuations and the less the magnitude of their drops and rises. In case of high density, the OD-V-GB extended view length curve is almost flat, while in case of low density there are multiple deep drops and high rises.

The explanation for the above observations is the following. In the OD-V-GB technique, the aggregated frames forwarded by the vehicles in the opposite direction are more like a stream of aggregated frames that the target vehicle is using to build its OD-V-GB extended view, and the vehicles on the opposite direction are like the stream suppliers. Any discontinuation in this stream will result in a drop in the OD-V-GB extended view length for the target vehicle, and the depth of the drop depends on the length of the discontinuation. The longer the discontinuation, the deeper the drop. Whenever the stream gets resumed after any discontinuation, this will result in a rise and the amount of aggregated frames coming in the stream controls the rise height. The more aggregated frames the stream carries, the higher the rise. Having low vehicular density in the opposite direction translates into not only fewer stream suppliers but also more sparse. As the stream suppliers are sparse, it means the time elapsed till the target vehicle gets in contact with the first stream supplier will be long, which will result in the deep drop in the step-wise decreasing part of the curve in Figure 88. Also, as the stream suppliers will be few, this means fewer aggregated frames in the stream, resulting in delaying the switching point and taking a longer time to reach the saturation point. The opposite will happen in case of high vehicular density, as it is translated to more and dense stream suppliers. As the stream suppliers will be dense, the time that the target vehicle will wait for coming in contact with the first supplier will be short which results in a small drop in the step-wise decreasing part of the curve in Figure 90. Also, the suppliers, high density

helps in having more aggregated frames in the stream, which will make the switching occur early and the saturation point be reached early.

VII.6 SUMMARY

In this chapter, we evaluated the various components in the CASCADE framework. We have shown through analysis and simulation that CASCADE makes efficient use of the wireless channel while providing each vehicle with data that is highly accurate, represents a large area in front of the vehicle, and can be combined with aggregated data from other vehicles to further extend the covered area. The technique used for vehicular data compression has a better compression ratio than the other compression techniques. Also, disseminating the vehicular data in a compressed format results in less bandwidth consumption and medium contention and a higher reception rate in addition to a longer view for the traffic ahead than disseminating the vehicular data in an uncompressed format. Using p-IVG in disseminating the various frames in CASCADE is showed to produce less medium contention and higher reception rate and visibility than standard IVG and flooding. The evaluation of the position verification technique used to tighten CASCADE security showed that the defense techniques can quickly detect false inputs given to the system and quarantine those vehicles responsible with very low rates of false positives. At the same time, the extra overhead incurred due to applying this technique to CASCADE was negligible. As p-IVG technique is loosely coupled with CASCADE, we have evaluated it outside the context of CASCADE. We have shown that p-IVG has close to 100% reception rate, regardless of the traffic density, which is a large improvement over basic flooding and IVG. In addition, p-IVG reduces the channel contention, so that vehicles with data to send have a low probability of finding the channel busy. Finally, we have shown that p-IVG achieves the goal of disseminating data to distant areas in a short amount of time in addition to having less redundancy and more coverage than IVG. Also, we have evaluated OD-V-GB using CASCADE and considering different vehicular density in the opposite lane. We showed that OD-V-GB is able to solve the discontinuity problem efficiently. As the vehicular density in the opposite lane increases, OD-V-GB takes less time to recover from the discontinuity problem, and it reaches the max extended view length faster.

CHAPTER VIII

EXAMPLE APPLICATIONS USING CASCADE

The CASCADE system has been shown to provide accurate data regarding the traffic ahead for at least 1.5 km (Chapter III). Moreover, it can utilize the aggregated frames that surrounding vehicles broadcast to further extend the driver's knowledge about the traffic ahead up to 26 km, as explained in Chapter IV. Applications can utilize CASCADE to build different views with different characteristics, *e.g.* different lengths and accuracies, using the same data by tuning just one parameter, *matching_percentage_thr*.

CASCADE can host many applications that can utilize the provided data in various ways. The data itself is not customized, but the usage of that data can be customized. Multiple applications can consume the same data and utilize it in different ways. In other words, assuming two applications belong to two different categories, their needs of the extended view characteristics will be different, even though they are running in the same vehicle. Both of these applications can use the same data (aggregated frames) provided by CASCADE and build different views of the traffic ahead with different length and accuracy characteristics with no extra overhead on the network.

In the following sections, we present two different CASCADE-based applications belonging to different vehicular application categories. In Section VIII.1, we present an advertising system that utilizes CASCADE in deciding the optimal time to broadcast advertisements in order minimize the number of advertisement messages broadcasted and hence reduce the network overhead and the advertiser cost. Section VIII.2 presents a merging assistance application based on CASCADE and discusses it briefly. In Section VIII.3, we provide general guidelines for how to set certain CASCADE parameters according to the application category.

VIII.1   VEHICULAR ADVERTISING SYSTEM BASED ON CAS-CADE

Any business (or company) tries to reach out to the maximum number of potential customers. A subset of those contacted will be attracted to the advertised product and buy it, which results in a profit for the company. So, the larger the number of

potential customers this company reaches, the more the profit it can gain. To achieve this goal, most companies use various advertising media.

There are well-known advertising media *i.e.* TV, radio, newspaper, etc. Advertising in these media is controlled by certain authority to manage the broadcasting schedule. For each advertisement broadcasted in certain media, the authority managing this media charges the advertising's company for it. Usually the charge varies based on two parameters, the advertisement duration and the broadcasting time. The intuition behind varying the advertisement cost based on its duration is straight forward, while varying the advertisement cost based on the broadcasting time needs to be explained.

For each advertising media, there are certain times during the day that are identified to have more customers watching/listening that media. These periods are identified by analysts through collecting offline feedback from customers and then building statistics to identify the time periods that attract most of the customers. The analysts usually call these time periods hot spots. Those hot spots vary all over the year and also they vary based on the location. To keep the hot spot estimation accurate, this cycle of collecting feedbacks, building statistics and identifying the hot spots should be repeated multiple times during the year in each geographical location. Because during the hot spots it is expected to have more customers watching/listening to the media, the charge for broadcasting an advertisement during these periods is higher than any other time in the day.

The cost of estimating the hot spots is one of the major drawbacks of the current advertising system. From the advertiser perspective, these hot spots are just an estimation for time periods that may attract the majority of customers and there is no guarantee that during those periods there are actually the maximum number of customers. Also, the only criteria that advertisers can set to get their advertisements broadcasted is the time of day. If there is somehow a method that will help in determining the actual number of media listeners/watchers (potential customers), this will be more accurate estimation for the hot spots. That is what the vehicular advertising system can provide while the current system cannot. Any vehicular advertising system can easily determine if there is a vehicle (a potential customer) passing by or not. Also, it can determine the vehicular density in the surrounding area using the wireless transmission activity level. These capabilities enable the vehicular advertising system to provide a cheap yet more accurate estimation for the

hot spots. As a consequence, the advertiser can set more criteria, such as vehicular density, for broadcasting his advertisements in addition to the time of day. Using this new criteria will allow for dynamic scheduling for broadcasting the advertisements instead of the static predefined schedule.

The benefits of using the vehicular advertising system can be summarized in the following points:

- More accurate estimation for the hot spots

- Maximizes the number of traveling customers who will receive the advertisements and hence increase the advertiser profit

- Minimizes the advertising cost charged to the advertiser as it will be charged only when their advertisements are broadcast according to their predefined criteria

- Gives the advertisers more flexibility for determining when to broadcast their advertisements by providing them with more criteria to set to get their advertisements broadcasted

As the most important advantage of using the vehicular advertising system is the live hot spot estimation, the accuracy of this estimation will be the main differentiating point between the various vehicular advertising systems. The more accurate the hot spot estimation, the more potential customers who will receive the advertisements using fewer broadcasts. Having fewer broadcasts implies less media contention and less charges on the advertiser side. Using vehicular networks for disseminating advertisements was first proposed by Nandan et al. [97]. Our goal here is to show how CASCADE can be used to implement such as system.

In the following sections, we explain a vehicular advertising system that is based on CASCADE. To show the impact of using CASCADE as the foundation for the proposed vehicular advertising system, we compare the proposed system to other vehicular advertising systems that are similar to the proposed system except in how they determine the vehicular density. The proposed system uses CASCADE to obtain an accurate estimation for the vehicular density and the other systems use different methods for estimating the vehicular density.

The proposed vehicular advertising system provides the advertisers with the flexibility to determine the geographical locations where they want to broadcast their
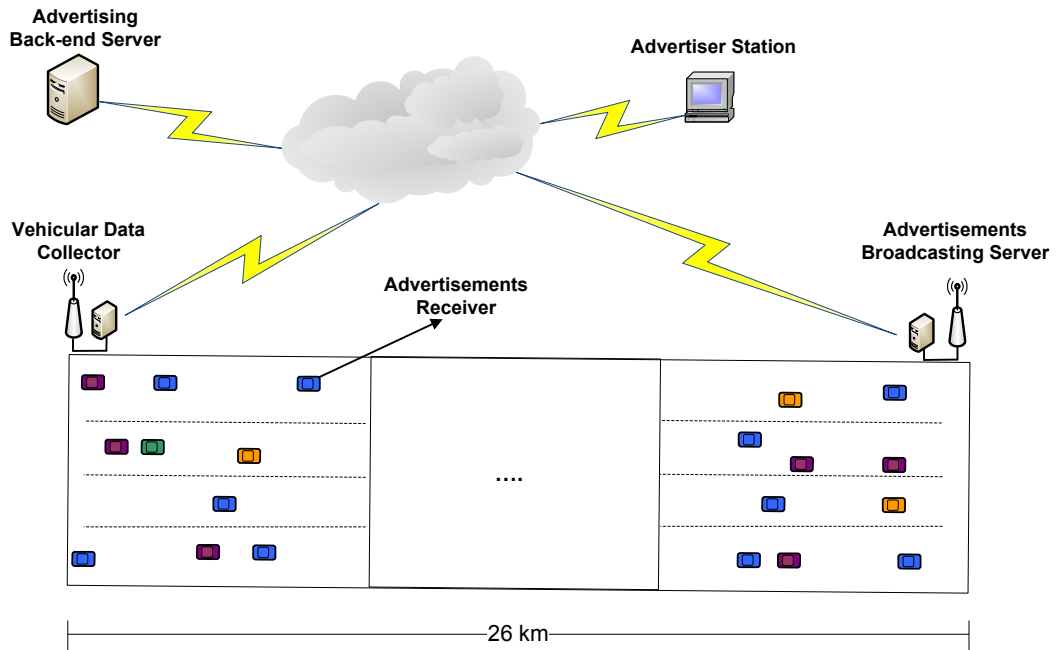
FIG. 93: Vehicular Advertising System Architecture

advertisements. Additionally, they can set time periods during the day and the corresponding vehicular density threshold for each period during which their advertisements should be broadcasted. This gives the advertisers more flexibility to control where and when their advertisements should be broadcasted. In the following sections, we will discuss the architecture for the proposed vehicular advertising system and how it operates, in addition to comparing its performance compared to other vehicular advertising systems that are not based on CASCADE.

VIII.1.1 Architecture of the Vehicular Advertising System

Figure 93 shows the architecture of the vehicular advertising system, consisting of four components:

- Advertising Back-end Server: This is the server where all of the advertisements are stored with the predefined broadcasting criteria that the advertisers set. These advertisements are pushed on demand to the advertising broadcasting servers that satisfy the geographical location constraint.

- Advertisement Broadcasting Server: This server is a key component in the vehicular advertising system as it is responsible for the following:

- Storing the advertisements targeting this geographical area.

- Broadcasting the advertisements after checking their corresponding broadcasting criteria.

- Processing the CASCADE aggregated frames received from the associated data collector and maintaining a recent average vehicular density for each kilometer along the road between its location and the associated data collector location. Also, it should maintain a confidence value for each average vehicular density calculated

- Vehicular Data Collector: Each advertising broadcasting server should have a vehicular data collector associated with it that is placed before it with a distance that differs according to the system settings. The main functionality for the data collector is to relay the aggregated frames received from passing vehicles to the associated broadcasting server.

- Advertisements Receiver: This component is responsible for receiving the advertisements broadcasted by the broadcasting server, verifying them, and then displaying them to the driver in a certain format (visual or audible)

VIII.1.2   Operation Sequence of the Vehicular Advertising System

The system starts with the advertisers submitting their advertisements to the advertising back-end server along with all the broadcasting criteria for each advertisement. The following summarizes the broadcasting criteria that should be specified with each submitted advertisement:

- The targeted geographical location(s) at which each advertisement should be broadcasted

- The targeted broadcasting times of day for each geographical location

- The vehicular density threshold for each time of day specified in the previous step to broadcast the corresponding advertisement

The advertising back-end server categorizes the submitted advertisements according to the geographical location constraint set by the advertisers and pushes each category to the corresponding advertisements broadcasting server in each location. Table 14 shows an example of the advertisements stored at a certain advertisement

TABLE 14: Advertisement Broadcasting Criteria at an Advertisement Broadcasting Server

Advertisements	Broadcast Time	Vehicular Density Threshold
Advertisement 1	6AM to 10AM	40 Veh/km
	1PM to 4PM	70 Veh/km
	5PM to 9PM	90 Veh/km
Advertisement 2	6AM to 10AM	66 Veh/km
	2PM to 8PM	80 Veh/km
⋮	⋮	⋮

broadcasting server and the broadcasting criteria for each of them. As each advertisement broadcasting server has all the advertisements targeting its location, it should continuously evaluate the broadcasting criteria for each advertisement to determine whether to broadcast or not. Whenever the broadcasting criteria for an advertisement is satisfied, the server broadcasts it and charges the advertiser for this broadcast. The vehicular data collector continuously forwards the CASCADE aggregated frames received from the passing vehicles to the associated advertisement broadcasting server. Using these aggregated frames, the advertisements broadcasting server will be able to maintain an accurate estimate of the vehicular density for each kilometer along the highway between the vehicular data collector and the advertisement broadcasting server location. The maximum distance between the vehicular data collector and the advertisement broadcasting server should not exceed 26 km. This limitation is inherited from the maximum extended view length in CASCADE. The actual distance between the vehicular data collector and the advertisement broadcasting server should be set based on the value used for the $matching\_percentage\_thr$ parameter. Figure 94[1] explains the relationship between the values used for the $matching\_percentage\_thr$ parameter and the upper bound for the maximum distance between the vehicular data collector and the advertisement broadcasting server.

By maintaining the vehicular density thresholds, the advertisement broadcasting server can pre-evaluate the broadcasting criteria and prepare a list of candidate advertisements to broadcast for each kilometer. For this subset of advertisements (the

---

[1]This figure is a copy of Figure 25.

FIG. 94:   Selecting the Optimal *matching_percentage_thr* for Each Applications Category

candidate advertisements) in the nearest kilometer, the advertisement broadcasting server can only evaluate the broadcasting criteria continuously till they reach the transmission range and the final broadcast decision is taken.

In the following section, we evaluate the CASCADE-based vehicular advertising system as compared to vehicular advertising systems that use methods other than CASCADE to evaluate the vehicular density.

VIII.1.3   Evaluation

To evaluate the impact of using CASCADE as the framework that provides an estimate for the vehicular density to the proposed vehicular advertising system, we will compare it to similar systems that do not use CASCADE. The following are the vehicular advertising systems that we are going to use in comparison:

- Advertising System 1: This system represents the simplest vehicular advertising system as it only requires the broadcasting server. In this system, the advertisement broadcasting server will be loaded with the advertisements and the waiting time between each advertisement broadcast. For example, if the

waiting time is set to 10 sec, this means all the stored advertisements in this server will be broadcasted every 10 sec. The optimal value for this parameter is the travel time that a certain vehicle will take to cross the transmission range of the advertisement broadcasting server. As the effective transmission range is $500 \, \text{m}^2$ and the maximum speed is $30 \, \text{m/s}$, so the waiting time should be set to 16.667 sec.

- Advertising System 2: The disadvantage of the first advertising system is that it will continue broadcasting advertisements even if there are no vehicles passing by, which is a waste of power and bandwidth, in addition to more charges to the advertisers. So, advertising system 2 avoids such problem by continuously listening to the wireless channel until it senses a transmission activity, which implies that there is at least one vehicle passing by that has transmitted a message. Then the server will broadcast the advertisements and, to avoid performing redundant broadcasts, the server will wait (sleep) till this vehicle leaves its effective transmission range (500 m). During the waiting time, the server will stop listening to the media and hence will not broadcast any advertisements. This system has also one parameter, the waiting time after broadcasting the advertisements (the server sleep time). Using the same settings as in the first advertising system, the optimal value for the waiting time will be 16.667 sec. The main advantage in this system is that if there are no vehicles passing by, there will be no advertisements broadcast and hence no charges to the advertiser and no waste of power or bandwidth.

- Advertising System 3: In the second advertising system, the advertisement broadcast is triggered upon detecting any transmission activity, but does not consider the transmission activity level[3] in deciding whether to broadcast the advertisements or not. This drawback limits the system from considering the vehicular density into its broadcasting criteria. The third advertising system avoids this drawback by applying the following checks before broadcasting the advertisements. It continuously listens to the media and, once it detects a

---

[2]The 500 m is the diameter of the effective transmission range.

[3]The transmission activity level is a measure for the number of transmissions occurring during a certain amount of time within the effective transmission range of the advertisement broadcasting server. The more transmissions, the higher the transmission activity level.

transmission activity, it starts a transmission monitoring timer. If the transmission activity level reaches a certain threshold before the timer expires, the server broadcasts the advertisements and goes into sleep mode for a certain time, similar to the second advertising system. Assuming the transmission activity level is a good estimate for the vehicular density, this system can differentiate between having one vehicle or k vehicles passing by. Hence, it will be able to apply all the criteria set by the advertisers in Table 14.

The evaluation metric that we are going to use in comparing the vehicular advertising systems is the number of messages that each system will broadcast under the same circumstances. Each message will contain all the advertisements that satisfy the broadcast criteria. The behavior of the vehicular advertising systems is impacted by the following traffic characteristics, average vehicle speed, vehicular distribution and density of communicating vehicles. So, studying the performance of these systems considering the variation that may occur in the traffic characteristics is an important way to differentiate between them.

In the optimal vehicular advertising system, the advertisement broadcasting server is expected to broadcast the *minimum* number of advertisements messages so that *all* the communicating vehicles passing by this server will receive *only* one copy of the broadcasted advertisements. In order to achieve these goals, the advertising system should consider the advertisement broadcasting timing, which is the waiting time after detecting the first new vehicle entering the transmission range of the broadcasting advertisements server. Figure 95 shows the different timings that a server could use and the problems that could occur as a consequence. In the optimal situation, the server should broadcast the advertisements when the maximum number of vehicles are within its transmission range (assuming the threshold for communicating vehicular density is reached) as in Figure 95-a. The choice for the perfect broadcast timing depends on the following points:

- Detecting the time at which the first communicating vehicle entered the transmission range for the broadcasting server

- Counting accurately the number of communicating vehicles that are within the transmission range

- Detecting the average speed of the vehicles that are within the transmission range

(a) Optimal Broadcast Time   **(b) Early Broadcast Time**   **(c) Late Broadcast Time**
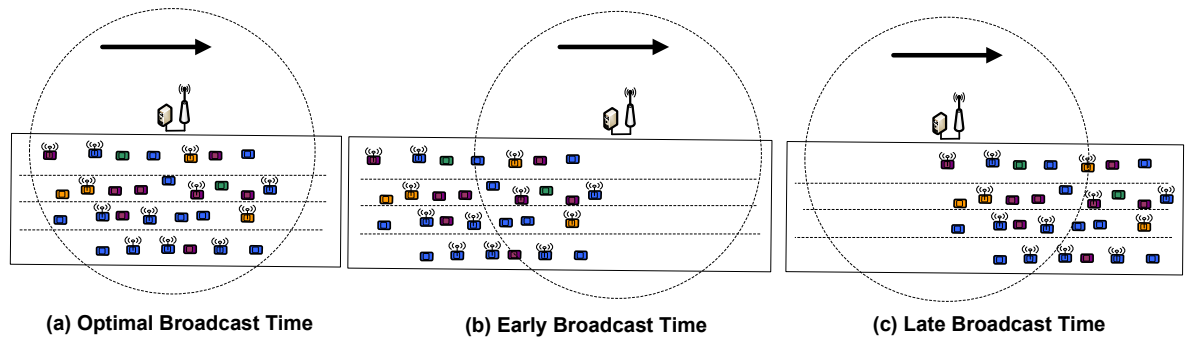
FIG. 95:   Different Timings for Advertisements Broadcasting

Any advertising system that will not perform the above three points with an acceptable accuracy will suffer from either the early broadcast or late broadcast problem that are shown in Figure 95-b and 95-c, or even both of them. The early broadcast problem usually occurs in the systems that cannot differentiate between the identities of message senders. So, it receives two or more messages from the same vehicle and counts them as if they are coming from different vehicles. This will results in an over-estimate of the number of communicating vehicles that are within the transmission range. Also, this problem may occur if the vehicles are traveling with a speed lower than the expected speed. The late broadcast problem usually occurs when the system fails to accurately detect when the first communicating vehicle entered the transmission range. Also, this problem may occur if the vehicles are traveling with a speed higher than the expected speed. Another consequence of the failure to react to an unexpected average speed change is either to have a low coverage percentage if the average speed increased or to have more redundancy if the average speed decreased. If the average speed increases, some vehicles will leave the transmission range earlier than what the server expects, so they will not be able to receive the advertisements, which reduces the coverage percentage. If the average speed decreases, some vehicles will stay within the transmission range longer than what the server expects, so they will receive the advertisements more than once, which increases the redundancy percentage.

The reaction to a change to the vehicular distribution is another important factor that helps to differentiate between the advertising systems. Here we focus on two vehicular distributions, uniform and clustered distribution. We assume a medium vehicular density (66 veh/km) and that the percentage of communicating vehicles is
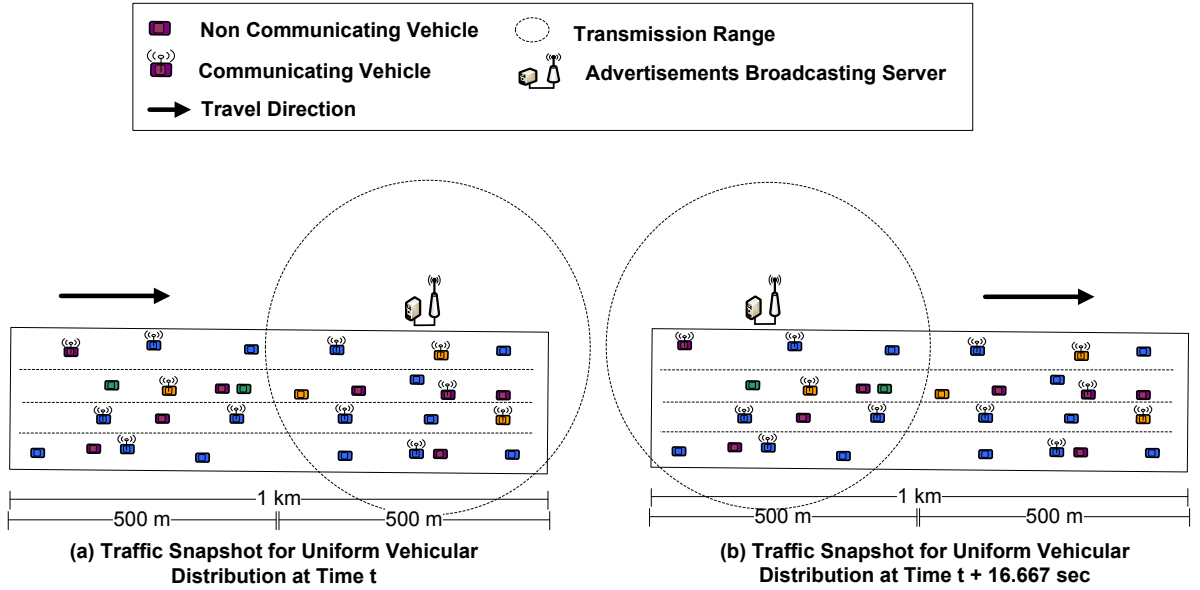
FIG. 96: Traffic Snapshot at Two Contiguous Broadcasting Times: Uniform Vehicular Distribution (Assuming Average Speed 30 m/s)
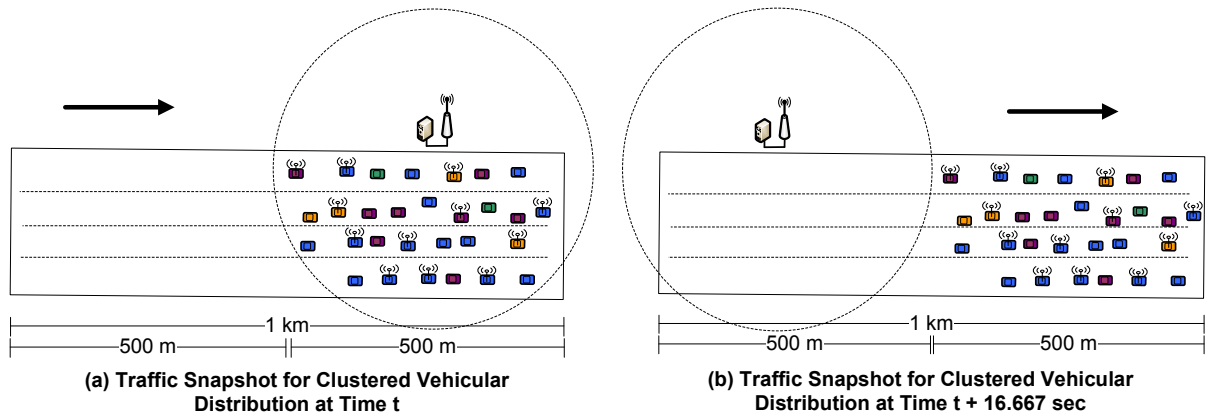
FIG. 97: Traffic Snapshot at Two Contiguous Broadcasting Times: Clustered Vehicular Distribution (Assuming Average Speed 30 m/s)

40%, so the density of communicating vehicles is 26 communicating vehicles/km. In case of a uniform distribution, the effective transmission range (500 m) of the advertisements broadcasting server should have 13 communicating vehicles at any time. In case of a clustered distribution, all the vehicles including the communicating ones will be spread over the shortest possible distance, which will be probably less than 500 m. This will result in having all of the vehicles exist in one effective transmission range, while the other effective transmission range in that kilometer will be empty. Figures 96 and 97 show how the vehicles will be distributed over 1 km in case of uniform and clustered distribution, respectively. The uniform distribution will cause less trouble to the vehicular advertising systems because within a single kilometer there will be no vehicular distribution change that requires special handling. In this case, most of the problems will be caused by a change in the average speed. Any advertising system that will not react appropriately to speed change will suffer from either the less coverage or more redundancy problem. The cluster distribution is more tricky to the advertising system than the uniform distribution, as it requires a reaction to the distribution change that can occur occurs within a single kilometer, in addition to the speed change that may occur.

To simulate the behavior of the optimal vehicular advertising system we will need to record actual traffic data generated during the simulation and replay it offline. To study the impact of varying the following traffic characteristics (*i.e.*, vehicular average speed, vehicular distribution and vehicular density) on the vehicular advertising system, we are going to conduct three experiments, each one will focus on studying the impact of varying one characteristic while fixing the other two.

Average Vehicle Speed

In this experiment we focus on studying the impact of varying the average vehicle speed on the number of advertisement messages that each system broadcasts. We have fixed the other system variables. The vehicular density has been set to medium (66 veh/km). Also, the percentage of communicating vehicles will be set to a 60%. Within the experiment, we simulated 24 hours that we divided into three regions to study various average speeds. In the first region, we set the average speed to 30 m/s (medium), in the second region we set it to 20 m/s (low), and we set the average speed to 40 m/s (high) in the third region. The only parameter that the three advertising systems need to set is the waiting time, which we set to 16.667 sec,
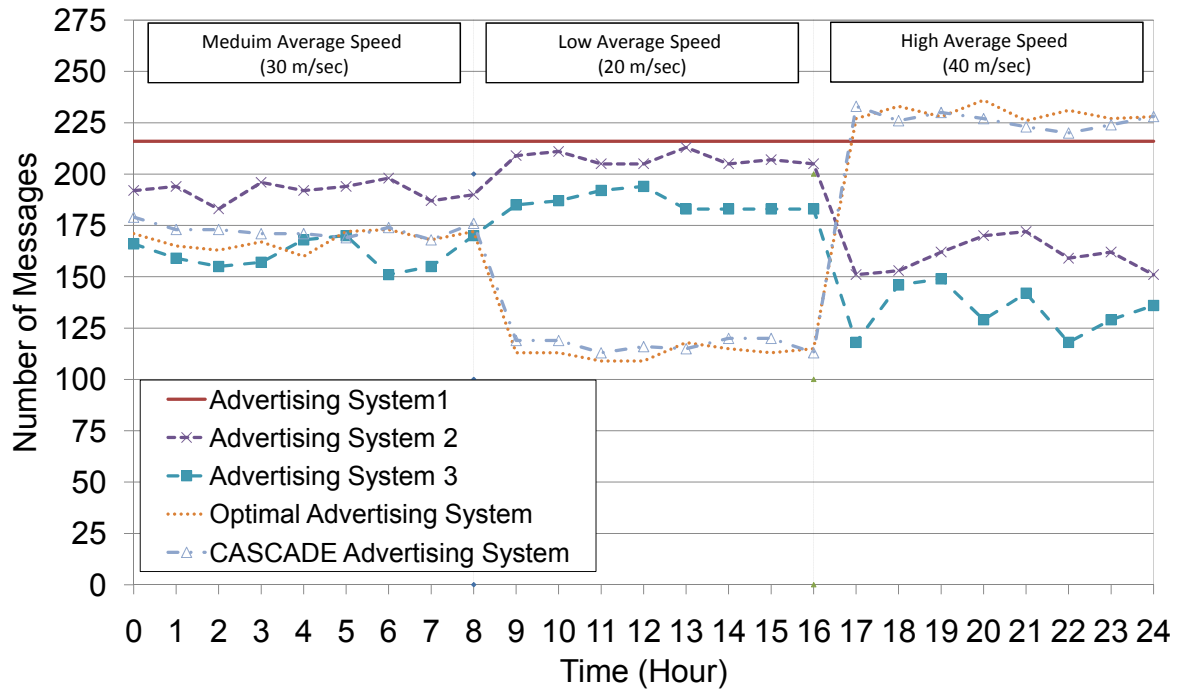
FIG. 98: Impact of Changing the Average Vehicle Speed on the Number of Messages Broadcasted

which is the optimal value for a speed of 30 m/s. We include Table 15 to show the optimal waiting times for each of the average speeds used in this experiment.

Figure 98 shows the experiment results represented as the number of messages broadcasted by each advertising system during low, medium and high average speed. The behavior of the first advertising system is deterministic and independent of any other system variables. The only parameter that impacts this system is the waiting time between the advertisement broadcasts (16.667 sec). The number of advertisements messages that the system broadcasts is constant and depends on the waiting time parameter value. Each 16.667 sec, the system broadcasts one advertisement message, resulting in 216 messages per hour.

The number of messages that the second advertising system broadcasts varies according to the vehicular speed. In general, the number of messages that this system broadcasts is higher than the number of messages that the optimal system broadcasts. This system suffers from the early broadcast problem as it starts broadcasting the advertisement once it detects a single communicating vehicle within the server

TABLE 15: Optimal Values for the Waiting Time Parameter Considering Various Vehicular Speeds

Vehicular Average Speed	The Optimal Waiting Time
Low Speed (20 m/s)	25 sec
Medium Speed (30 m/s)	16.667 sec
High Speed (40 m/s)	12.5 sec

transmission range. Compared to the optimal vehicular advertising system, this advertising system has the closest performance to the optimal system only in case of medium speed because the waiting time parameter is preset to its optimal value (Table 15). In low and high speeds, the performance is far from optimal, because this system cannot adjust its waiting time parameter at runtime according to the change in speed. In case of low speed, we notice that the number of messages broadcasted is higher because the vehicles will stay within the server transmission range longer that what the server expects, while the opposite occurs in case of high speed as the vehicle leave the server transmission range faster than what the server expects.

The behavior of the third advertising system is similar to the second advertising system with the only difference being that the former waits till the transmission activity level (the number of communicating vehicles) within the broadcasting server transmission range reaches certain threshold before it broadcasts the advertisement. As a consequence, it broadcasts fewer messages than second advertising system.

As expected, the performance of the CASCADE based advertising system is the closest to the optimal. As we explained in Sections VIII.1.1 and VIII.1.2, the vehicular data collector using CASCADE provides the advertisement broadcasting server with a continuous flow of aggregated frames that summarizes the traffic between the collector and the server. The server can benefit from this flow in estimating the vehicular density per kilometer and hence adjust the optimal waiting time used according to the recent vehicular density update. Also, the server will be able to know the vehicular distribution within each kilometer by reconstructing the aggregated frames. This allows it to choose the optimal timing for broadcasting the advertisements. With all the capabilities that CASCADE provides to the advertising system, we expected to see the number of messages it broadcasts to be the
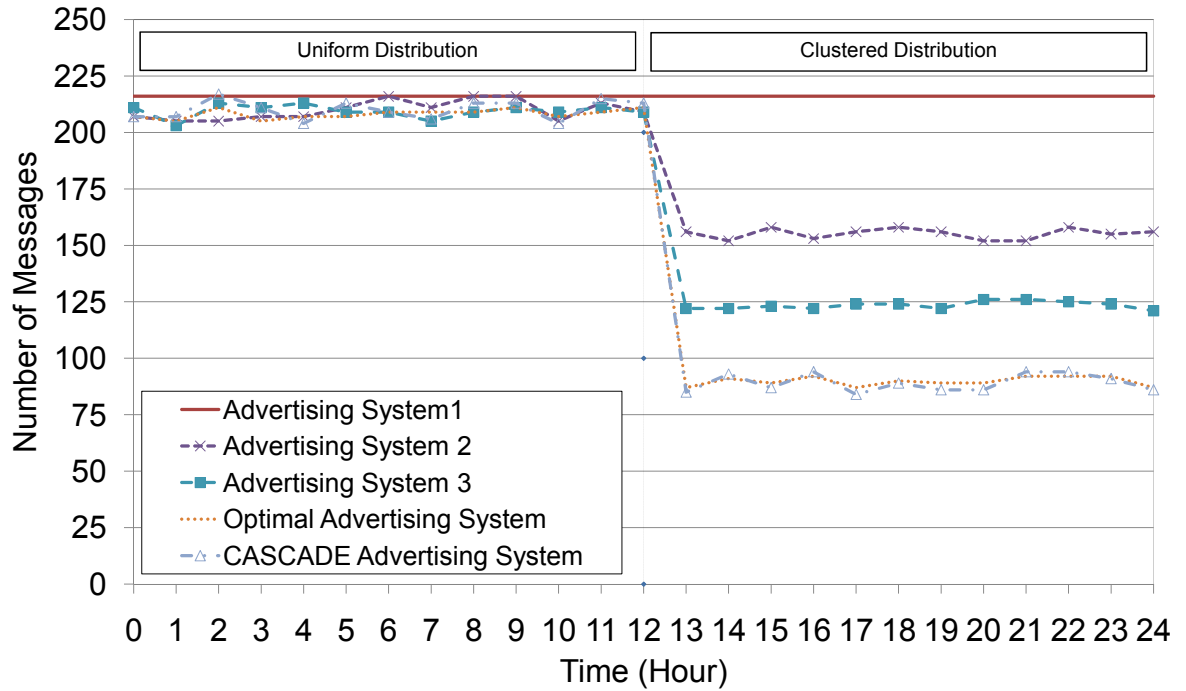
FIG. 99: Impact of Changing the Vehicular Distribution on the Number of Messages Broadcasted

same as the number of messages that the optimal advertising system broadcasts. But we still see small differences between them, ranging from 1.36% to 3.31%. The reason for this difference is the accuracy of the aggregated frames, which directly depends of the *matching_percentage_thr* parameter used in CASCADE. In this experiment, we set the *matching_percentage_thr* to 74%. The impact of changing the *matching_percentage_thr* on the CASCADE based advertising system is studied in later in this section.

Vehicular Distribution

The focus of this experiment is to study the impact of the vehicular distribution on the advertising systems performance. So, for the first part of the experiment we fix the vehicular distribution to be uniform then we switch to use the clustered distribution for the second part. The other experiment variables will be fixed. The vehicular density is set to medium 66 veh/km, the percentage of communicating vehicles will be set to 60%, the vehicular speed is set to 30 m/s, and the waiting time parameter is set to 16.667 sec.

Figure 99 shows the experiment results. As we explained before the number of messages sent by the first advertising system is constant. Where the vehicular distribution is uniform, the performance of all the advertising systems including the CASCADE based one is similar and very close to the optimal. The reason for this is that all of the experiment settings are set to their optimal values. Also, as the vehicular distribution is uniform, this eliminates any traffic flow fluctuations that could make the non-CASCADE based advertising systems behave non-optimally. In the second part of the experiment, the vehicular distribution has been set to clustered. The first observation on the results in Figure 99 is that the number of messages broadcasted by all of the advertising systems except the first one is much less than when the vehicular distribution was uniform. The reason for this is that the advertising systems will detect the empty half in each kilometer and hence do not broadcast any advertisements in this section (Figure 97). Despite the drop in the number of broadcasted messages, the performance of the third advertising system is still far from the optimal, and the performance of the second one is even worse. The second advertising system suffers from the early broadcast problem, as the server will broadcast once at the beginning of the cluster upon detecting the first transmission activity and 40% of the time it performs another broadcast for the same cluster. The third advertising system will only broadcast a second time 17% of the time for the same cluster. For the same reasons that we explained in the previous section, the CASCADE based advertising system has the closest performance to the optimal.

Vehicular Density Threshold

In this experiment we focus of studying how each advertising system will react to the change in the density of communicating vehicles. In the first part of the experiment, we set this percentage of communicating vehicles to be greater than the vehicular density threshold for any advertisement and in the second part we lower the percentage to be less than the lowest threshold. We fix the other experiment variables. The vehicular density is set to medium 66 veh/km, the vehicular distribution is set to uniform, the speed is set to 30 m/s, and the waiting time parameter is set to 16.667 sec.

In the results shown in Figure 100 we can distinguish between the two experiment parts. In the first part, where the percentage of communicating vehicles is greater
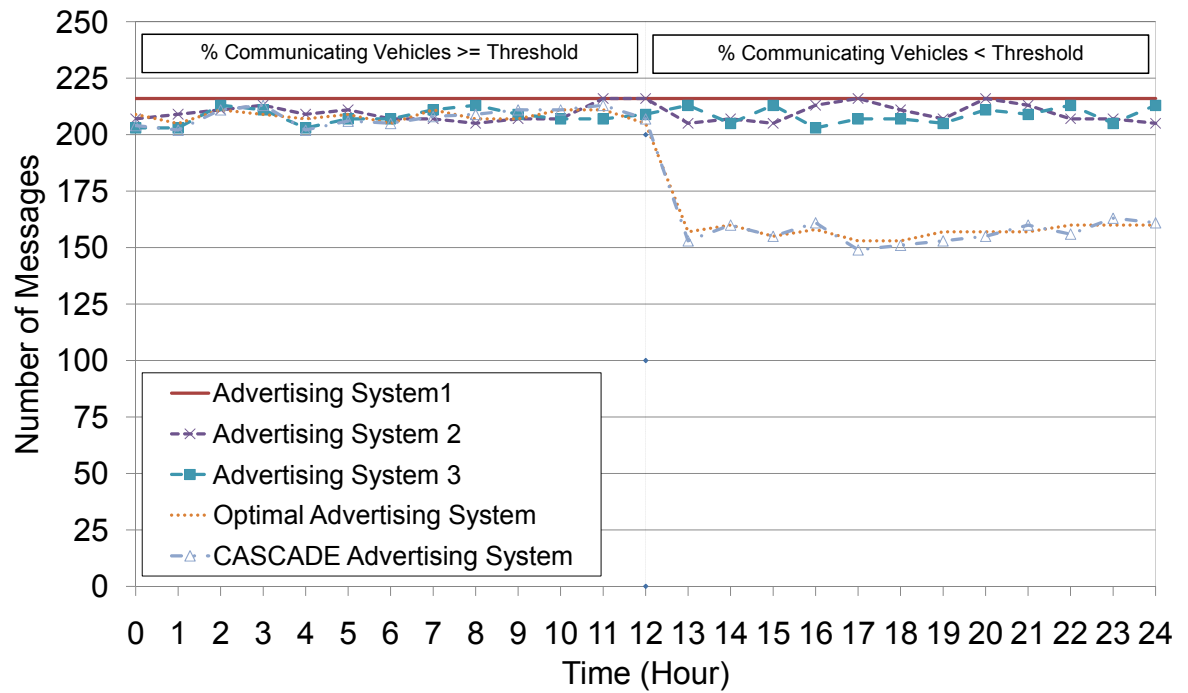
FIG. 100: Impact of Changing the Percentage of Communicating Vehicles on the Number of Messages Broadcasted

than the vehicular density threshold for any advertisement, all the vehicular advertising systems have similar performance. The reaction to changing the percentage of communicating vehicles is more clear in the second part of the experiment where we set the percentage of communicating vehicles to be less than the threshold. The performance of the first advertising system is expected and has been explained before. We notice that the performance of the second advertising system is not changed despite the drop in the percentage of communicating vehicles, because the broadcast server needs only one transmission activity to make a broadcast decision. As it is more sensitive to the transmission activity, we expected the third advertising system to have a drop with the drop in the percentage of communicating vehicles, but it did not. The reason for this is that the broadcasting server in this system cannot differentiate whether the messages are sent by the same vehicle or different vehicles, so it over-estimates the percentage of communicating vehicles and decides to broadcast the advertisement when it should not. Like in the previous experiments, the CASCADE-based system performs near optimally.

TABLE 16: Difference Between the CASCADE Based Advertising System and the Optimal Advertising System as the $matching\_percentage\_thr$ Parameter Changes

$matching\_percentage\_thr$ Parameter	The difference Range
31%	4.78% to 9.06%
74%	1.36% to 3.31%
100%	0.02% to 0.005%

Varying the $matching\_percentage\_thr$ Parameter

In this experiment we focus on studying the impact of varying the $matching\_percentage\_thr$ on the CASCADE based advertising system. In this experiment we fixed all other system parameters, and we change only $matching\_percentage\_thr$ parameter. Table 16 summarizes the differences between the optimal system performance and the performance of CASCADE based advertising system. We notice that as the $matching\_percentage\_thr$ increases, the difference decreases which implies that the CASCADE based advertising system gets closer to the optimal. The reason is that when the $matching\_percentage\_thr$ increases, the accuracy for the aggregated frames increases, so the broadcast server can make more accurate decisions on the time to broadcast the advertisements. This decreases the chance for making an early or late broadcast. From the data in Table 16, we can conclude that the value for $matching\_percentage\_thr$ parameter should be set to 100% to get the best performance out of the CASCADE-based Advertising System. But we should also consider the side effect of increasing the $matching\_percentage\_thr$ as it will lead to a shorter extended view (Figure 94) and hence limited knowledge about the traffic in between the data collector and the broadcast server. So the decision on the best value for the $matching\_percentage\_thr$ parameter should be made considering the data in Table 16 and Figure 94.

To conclude, the CASCADE-based vehicular advertising system was shown to have fewer bandwidth consumption (fewer messages broadcast) than the other vehicular advertising systems regardless of the traffic characteristic variations introduced in each experiment. The performance that the CASCADE-based vehicular advertising system achieved was very close to the optimal expected performance in all conducted experiments.

VIII.2 VEHICULAR MERGE ASSISTANT SYSTEM BASED ON CASCADE

The areas around highway entries are more vulnerable to accidents and traffic congestion than any other section in the highway, specifically in the lane adjacent to an entrance ramp. For example consider the right most lane in Figure 101. Although the vehicle entering the highway tries to gain speed while driving on the ramp, its final speed just before entering the highway will still be less than the speed of the vehicles driving on the highway. So, in Figure 101, if the highlighted area $E1$ in the right most lane is not empty, the vehicles in the entrance ramp, area A, will either have to stop and wait for safe chance to merge or to continue and merge into the highway in a risky manner. Both decisions are not optimal. If the vehicle decides to stop, this means it will lose the speed that it gained while driving on the ramp and hence it will need a longer distance to reach to the highway speed. Additionally, it will force the other vehicles on the ramp to stop, which may cause a traffic congestion. Also, the same merging problem will propagate to the vehicles waiting behind it on the ramp. The situation that results from the second option (merge into the highway in a risky way) will depend on the behavior of the vehicle driving in the right most lane, as it can either continue with its speed which will probably lead to a collision or it can slow down which will lead to a congestion on that lane. It could also change lanes (move one lane to the left), which will cause the merging problem to be shifted one lane to the left.

In the following we propose a solution that can reduce the probability of merge problem occurring. The symbols and numbering in Figure 101 will be used in explaining the solution. The merge assistance application equipped in the vehicle driving on the ramp should send a merge assistance request once it reaches the line X. As the vehicles' speed on the ramp is less than the vehicles' speed in the highway, by the time the vehicle on the ramp reaches area A, all the vehicles in the highway ahead of line X will be already passed the highway entry, so no action will be required on their part. All the vehicles before line X should be aware of the existence of a vehicle on the ramp that will need to merge. As the merge assistant application is built on top of CASCADE, they will also be aware of the traffic data ahead, through its local view. The action required from the vehicles in lane 1 is to manage in a *safe way* to change lanes to the left to guarantee that area $E1$ will be empty. If these vehicles cannot manage change lanes until they enter area $F1$, this means further
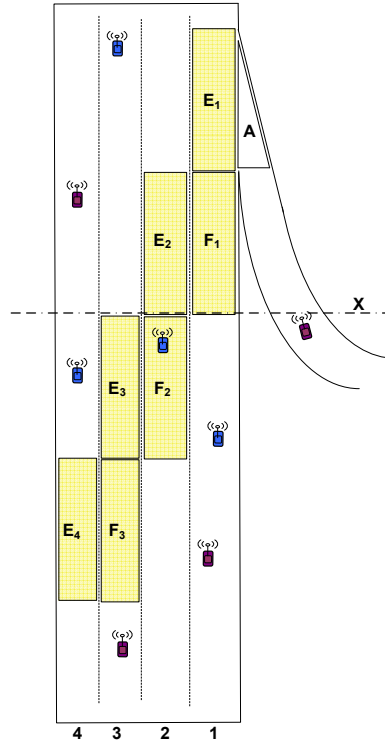
FIG. 101: Vehicular Merge Assistant System Based on CASCADE

assistant is required from the vehicles in lane 2, otherwise the traffic in lane 2 should not be impacted. If the vehicles in lane 2 detect a vehicle in area $F1$, they should manage in a *safe way* to change lanes to the left to guarantee that area $E2$ will be empty. If these vehicles cannot change lanes until they enter area $F2$, this means further assistant is required from the vehicles in lane 3, otherwise the traffic the lane 3 should not be impacted. The lane change effect propagates across all lanes until the left most lane. This implies that the merge problem is not avoidable unless a slow down is enforced in one of the lanes, but this may lead to congestion in that lane. The proposed solution reduces the probability of having a merge problem when it is possible to avoid such problem.

VIII.3   PARAMETER SETTINGS BASED ON APPLICATION CATEGORIES

The CASCADE framework has multiple parameters that control the internal behavior of its various components and hence allow it to accommodate the requirements

for most of the VANET application categories. One of these parameters is the recommended frequency for sending update messages for each application category [81]. For safety related applications, the recommended frequency for sending update messages is 10 messages/sec [81], while for the non-safety applications (i.e. entertainment) the recommended frequency for sending update messages is 1 message/sec [81]. The informational applications category lies in between the previous two categories, with respect to the frequency of sending update messages, so that the update messages should be sent out every 300-400 ms, which means almost 3 messages/sec. The frequency for sending update messages directly impacts the sending rate of primary frames so that they must be equal otherwise the application will be sending an obsolete update message multiple times. In this thesis, we have been sending the primary frames every 300-400 ms, which makes CASCADE fully compatible with both entertainment and informational applications. In order to extend CASCADE usability to include the safety related application, the primary frames sending rate should be modified to be every 100 ms.

Another parameter is *matching_percentage_thr* which controls the accuracy of the vehicular data in the extended view. The setting of this parameter is application-based, but we can provide a general guideline for the range of values based on the application category. The safety-related applications are sensitive to data accuracy, so setting the *matching_percentage_thr* parameter to be in the range 74% to 100% will guarantee highly accurate data, while it will keep the extended view to be short, ranging from 17.1 km to 3.12 km respectively. The informational applications are less sensitive to the data accuracy as they can tolerate consuming less accurate data and still generate useful results. So, setting the *matching_percentage_thr* parameter to be in the range 31% to 74% is acceptable. The corresponding extended view length will be in the range from 22.2 km to 17.1 km, respectively. The entertainment applications are usually the least sensitive to data accuracy, so the *matching_percentage_thr* parameter can to be in the range 0% to 31% and this will result in the longest extended view as it will be in range 25.48 km to 22.2 km, respectively.

VIII.4 SUMMARY

In this chapter we presented two applications, the advertising system application and the merge assistant application based on CASCADE. The advertising system has been presented in detail, and the performance of the CASCADE based advertising

system has been compared to some other advertising systems. Using CASCADE as a framework for the advertising system enhanced the system performance and made it very close to the optimal advertising system performance. The merge assistant system has been presented briefly to show that it is possible to build such application based on CASCADE and this application will benefit from the data that CASCADE will provide. Finally, we provide a general guideline for how to choose a suitable value for some of the CASCADE parameters based on the application category.

CHAPTER IX

SUMMARY AND FUTURE WORK

In this chapter, we provide a summary for the work that has been achieved in this thesis and a brief discussion for how the CASCADE framework can be extended as future work.

IX.1   THESIS SUMMARY

In this thesis, we presented a framework for data aggregation and dissemination in VANETs that consists of four main components, local view, extended view, data security and data dissemination. We collectively call this framework cluster-based accurate syntactic compression of aggregated data in VANETs, or CASCADE. This framework represents a layer that many applications can be built on top of. The main functionality of CASCADE is providing multiple versions of the vehicular data of the traffic ahead from the same incoming information. Each version of the vehicular data has different characteristics (vehicular data amount, accuracy and refresh rate) based on an application's settings. Having the ability to provide these multiple versions of vehicular data allows CASCADE to support not only informational applications, but also safety and entertainment applications. Additionally, CASCADE handles the transmission of vehicular data and hence relieves the applications from this burden and helps them to focus on their business logic. As CASCADE unifies the vehicular data processing and handling base for all consuming applications, this reduces the redundancy in the system on multiple fronts. First, the same vehicular data will not be sent multiple times by different applications running on the same vehicle, saving a huge amount of bandwidth. Second, the vehicular data processing, sending and receiving will not be the responsibility of each application and this has the potential to save tremendous amount of processing power. Hence, using CASCADE will solve the anticipated scalability problem for the VANET applications.

The success that CASCADE achieves in performing the previously mentioned functionalities is a result of a seamless integration and co-operation between the framework's various components. Each of the framework components performs multiple functions efficiently. In the following we will provide a brief summary about each component and its responsibilities.

Local View Component The local view component is the first building block in the CASCADE framework. It is responsible for building and maintaining an accurate view for the traffic ahead, but only for short distances, up to 1.5 km. This component updates its local view using the received primary frames, which are sent in signed frames containing a vehicle's position information. Frequently, the local view component compresses and aggregates the vehicular data in its local view to form an aggregated frame, which is broadcasted, so that other vehicles can use it to extend their views about about the traffic. We have investigated the alternatives for compressing the vehicular data in the local view and found that differential coding has the best compression ratio. To achieve the maximum possible compression, the local view is divided into clusters. The size of the cluster impacts the aggregated frame size which cannot exceed 2312 bytes, the IEEE 802.11 maximum frame size. The aggregated frame size controls the local view length. So, we analyzed the relationship between these three variables, aggregated frame size, cluster size and local view size, and we found that the optimal cluster size that provides the best trade-off between frame size and local view length is 16 m wide and 126 m long. Using this cluster size, we can have a local view of length 1.5 km. Having an optimal cluster size will reduce the bandwidth consumption and provide better extended driver visibility. One of the key features in this component and hence in CASCADE, is that it maintains and guarantees the high accuracy of not only the local view, but also the aggregate frames. So, whenever the aggregated frames are decompressed to reconstruct the original local view, the reconstructed view will have almost the same high accuracy as the original one. The reason for maintaining the accuracy is the lossless compression technique being used. We have shown through analysis and simulation that the local view component in CASCADE makes efficient use of the wireless channel while providing each vehicle with data that is highly accurate, represents a large area in front of the vehicle, and can be combined with aggregated data from other vehicles to further extend the covered area.

Extended View Component This component allows CASCADE to extend its view about the traffic ahead beyond the 1.5 km limit. All the vehicles in the system share their local views by broadcasting it in aggregated frames. Upon

receiving an aggregated frame, and after verifying it through the data security component, the extended view component decompresses the data in the received frame (or the RLV frame). It then reconstructs the original view and determines the intersection between the received view and its own local view. To determine whether to use the received frame to extend the local view or to discard it, CASCADE compares the vehicular data in the intersecting section between these two frames. Based on the result, the component decides what to do with the received frame. To process and compare the vehicular data in the intersecting views, we need to represent these vehicular data in a suitable data structure. Out of the possible data structures to represent the vehicular data, we selected the graph and KD-Tree data structures. Using the vehicular data comparator function, we can compare the vehicular data for two vehicles and find a measure, $diff\_percentage$, that shows how similar or different are these two vehicles. The smaller the $diff\_percentage$, the more similar the vehicular data for these two vehicles are. The comparison state of these two vehicles can be classified to be in any of the following states $TruePositive$, $FalsePositive$, $TrueNegative$ and $FalseNegative$, based on comparing the $diff\_percentage$ against $diff\_percentage\_thr$ and the vehicles IDs. Finding the optimal $diff\_percentage\_thr$ that will maximize the sum of $TruePositive$ and $TrueNegative$ and minimize the sum of $FalseNegative$ and $FalsePositive$ is an optimization problem that we solved experimentally in the case of medium vehicular density. The data structure used for representing the vehicular data introduced a small difference in the calculated optimal $diff\_percentage\_thr$. In case of using the graph data structure, the optimal $diff\_percentage\_thr$ is 16%, and in case of using the KD-Tree data structure, the optimal $diff\_percentage\_thr$ is 16.5%. At the frame comparison level, the result of comparing the received view to the local view is represented as percentage, $matching\_percentage$, that shows how similar are these two views. Based on comparing the $matching\_percentage$ to $matching\_percentage\_thr$, the received view will be accepted or discarded. Hence, the $matching\_percentage\_thr$ has an impact on the extended view length. We studied the impact of varying the $matching\_percentage\_thr$ on the extended view length. Choosing small $matching\_percentage\_thr$ will relax the matching constraint, resulting in a longer but less accurate extended view. The extended view length can reach

25.48 km if we set the *matching_percentage_thr* to 0%. While choosing a large *matching_percentage_thr* will drive the matching constraint to be more conservative, therefore the resulting extended view length will be short yet more accurate. If we set *matching_percentage_thr* to be 100%, the extended view length can be as short as 3.12 km. The decision as to what value we should set the *matching_percentage_thr* will be determined by each application based on the amount of data it needs and its required data accuracy.

Data Security Component The security fields in the frames (primary, aggregated) sent by CASCADE are calculated and appended to each corresponding frame by the data security component. This task is in addition to the component's main task which is verifying the vehicular data received in the primary frames. This component implements a light-weight position verification technique that is used within the context of CASCADE. The implemented technique is loosely coupled with CASCADE framework so that it can be used for VANETs in general. The technique is based on using the received signal strength (RSSI) and laser rangefinders for verifying the correctness of the vehicular position received in the primary frames. We have used the position verification technique to tighten CASCADE security and at the same time to evaluate the technique performance and its incurred overhead when applied to any other particular VANET system. Our defense techniques can quickly detect false inputs given to the system and quarantine those vehicles responsible, with very low rates of false positives. At the same time the extra overhead incurred due to applying this technique to CASCADE was found to be negligible.

Data Dissemination Component The data dissemination component is a core component in CASCADE framework as it is responsible for managing the data dissemination in the framework. As bandwidth is considered a scarce resource in VANETs, it is very important that this component operates efficiently. This component has two main techniques that have been designed and implemented to handle the data dissemination in CASCADE. The first is probabilistic Inter-Vehicular Geocast (*p*-IVG), and the second is On-Demand Vehicular Gap-Bridging (OD-V-GB).

In VANET, there are various data dissemination models such as same direction, opposite direction and both directions, that can be used in the V2V data

communication model. It has been proven by Nadeem *et al.* [92] and confirmed by our experiment results that the opposite direction data dissemination model is better than the other two models as it can disseminate the data to farther distances using less time and fewer hops. Due to the correlation between the data dissemination model that p-IVG uses (same direction data dissemination model) and the security enforcement component in CASCADE, changing p-IVG to use the opposite direction data dissemination model still has better performance, but its advantage over using the same direction data dissemination model has been reduced from 7% - 10% to 2% - 3%.

p-IVG was designed to alleviate the spatial broadcast storm problem that we discovered when studying standard IVG. p-IVG uses a light-weight beacon broadcasting utility to sense the local network topology and adapts itself to minimize the number of frame re-broadcasts needed to send a frame to distant areas. We have shown that p-IVG has close to 100% reception rates, regardless of the traffic density, which is a large improvement over basic flooding and IVG. In addition, p-IVG reduces the channel contention, so that vehicles with data to send have a low probability of finding the channel busy. Finally, we have shown that p-IVG achieves the goal of disseminating data to distant areas in a short amount of time in addition to having less redundancy and more coverage than IVG.

OD-V-GB is mainly designed to handle the traffic discontinuity problem that may happen in any V2V system whenever the distance between any two communicating vehicles exceeds 250 m due to either sparse or clustered traffic. To overcome this problem in CASCADE, we proposed a solution called On-Demand Vehicular Gap-Bridging or OD-V-GB. We evaluated OD-V-GB using CASCADE and considering different vehicular densities in the opposite lane. Our results showed that OD-V-GB is able to solve the discontinuity problem efficiently. As the vehicular density in the opposite lane increases, OD-V-GB takes less time to recover from the discontinuity problem and information reaches the max extended view length faster.

The main consumers for CASCADE are vehicular applications regardless of the category that they belong to. To demonstrate the effectiveness of CASCADE, we implemented a vehicular advertising system based on CASCADE and showed that

using CASCADE as part of the application boosted the application performance when compared to similar applications that are not CASCADE-based. Using CASCADE as a framework for the advertising system enhanced the system performance and made it very close to the optimal advertising system performance. For the other application, vehicular merge assistance, we explained the application and showed how using CASCADE will not only enable the application scenario, but also enhance its performance.

IX.2 FUTURE WORK

In this section, we present some ideas about how to take the CASCADE framework one step forward. One of the future work directions for CASCADE is to add more functionalities to the framework, *i.e.*, allowing the framework to communicate with roadside infrastructure so that the collected data can be stored in a centralized server for further processing and analysis. Storing the local views in centralized servers will be very useful in analyzing traffic situations especially in case of accidents or emergency situations.

Another direction is to enhance the functionality of the framework components, *i.e.*, enhancing the performance of the data dissemination component by changing the rebroadcast probability function used in p-IVG to be adaptive. The current design for p-IVG protocol uses the function $\frac{1}{N}$, where N is the vehicular density, to decide whether to perform a rebroadcast or not. This function depends only on the vehicular density in making the rebroadcast decision regardless of the current redundancy factor, which we showed how to calculate online in Section VII.4.5. Changing the rebroadcast probability function to consider the current redundancy factor should enhance the protocol performance. The proposed new rebroadcast probability function is $\frac{F(r)}{N}$, where $F(r)$ is a function that maps the redundancy factor r to a certain value. An example for such a suggested mapping is $F(r) > 1$ when $r < 0.4$, $F(r) = 1$ when $r = 0.4$ and $F(r) < 1$ when $r > 0.4$.

Currently, both the local view component and the extended view component lack the capability of handling curved roads (as we showed in Figure 6), which severely limits the extended view. So, modifying the clustering mechanism to accommodate such situations is one of the aspects that we should study to enhance the functionalities of both components.

The unlimited power source was one of the main advantages that VANETs had

over MANETs and sensor networks. All of the techniques that have been suggested and designed for VANETs have ignored the power consumption level for their operations or even did not consider it as a constraint in the solutions search space. As now the whole world is moving toward green computing and many vehicular manufacturing companies have already made huge forward step toward this goal by having the first totally electric vehicle running in the roads, all the techniques invented for VANETs should be reconsidered again after adding the power consumption constraint. So, a third direction for extending CASCADE is to optimize the operation of the framework various components toward reducing the power consumption while maintaining the same level of functionalities. An example for such optimization is to reduce the frame sizes and hence reduce the power needed to broadcast these frames. In the primary frame, each vehicle used to broadcast its location and speed as absolute values, while this can be optimized by creating a two types of primary frames, one containing these absolute values (base primary frames) that should be sent infrequently and the other one that should contain only the location and speed differences (difference primary frames) with respect to the most recent base primary frame.

BIBLIOGRAPHY

[1] Tamer ElBatt, Siddhartha Goel, Gavin Holland, Harihan Krishnan, and Jayendra Parikh. Cooperative collision warning using dedicated short range wireless communication. In *Proceedings of ACM VANET*, pages 1–9, Los Angeles, CA, September 2006.

[2] Raymond Tatchikou, Subir Biswas, and Francois Dion. Cooperative vehicle collision avoidance using inter-vehicle packet forwarding. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, pages 2762–2766, St. Louis, MO, December 2005.

[3] Ziyuan Wang, Lars Kulik, and Kotagiri Ramamohanarao. Proactive traffic merging strategies for sensor-enabled cars. In *Proceedings of ACM VANET*, pages 39–48, Montreal, Quebec, Canada, September 2007.

[4] Mahmoud Abuelela, Stephan Olariu, and Michele C. Weigle. NOTICE: An architecture for notification of traffic incidents. In *Proceedings of IEEE VTC-Spring*, pages 3001–3005, Singapore, May 2008.

[5] Jakob Eriksson, Lewis Girod, Bret Hull, Ryan Newton, Samuel Madden, and Hari Balakrishnan. The pothole patrol: using a mobile sensor network for road surface monitoring. In *Proceedings of the International Conference on Mobile systems, Applications, and Services (MobiSys)*, pages 29–39, Breckenridge, CO, June 2008.

[6] Clarus initiative. `http://www.clarusinitiative.org`, 2004.

[7] Jörg Ott and Dirk Kutscher. Drive-thru Internet: IEEE 802.11b for "automobile" users. In *Proceedings of the Annual Joint Conference Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 362–373, Hong Kong, March 2004.

[8] Joon-Sang Park, Uichin Lee, Soon Y. Oh, Mario Gerla, and Desmond S. Lun. Emergency related video streaming in vanet using network coding. In *Proceedings of ACM VANET*, pages 102–103, Los Angeles, CA, September 2006.

[9] Kevin C. Lee, Seung-Hoon Lee, Ryan Cheung, Uichin Lee, and Mario Gerla. First experience with CarTorrent in a real vehicular ad hoc network testbed. In *Proceedings of IEEE MOVE*, Anchorage, AK, May 2007.

[10] Mahmoud Abuelela and Stephan Olariu. Zipper: a zero-infrastructure peer-to-peer system for VANET. In *Proceedings of the ACM Workshop on Wireless Multimedia Networking and Performance Modeling (WMuNeP)*, pages 2–8, Crete Island, Greece, October 2007.

[11] US Department of Transportation. Standard Specification for Telecommunications and Information Exchange Between Roadside and Vehicle Systems. ASTM E2213-03, August 2003.

[12] IEEE. Status of Project IEEE 802.11p. `http://grouper.ieee.org/groups/802/11/Reports/tgp_update.htm`, November 2006.

[13] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. In *Proceedings of the ACM International Conference on Mobile Computing and Networking (MOBICOM)*, pages 151–162, Seattle, WA, August 1999.

[14] Fabio Picconi, Nishkam Ravi, Marco Gruteser, and Liviu Iftode. Probabilistic validation of aggregated data in vehicular ad-hoc networks. In *Proceedings of ACM VANET*, pages 76–85, Los Angeles, CA, September 2006.

[15] Vladimir Bychkovsky, Bret Hull, Allen Miu, Hari Balakrishnan, and Samuel Madden. A measurement study of vehicular internet access using in situ wifi networks. In *Proceedings of the ACM International Conference on Mobile Computing and Networking (MOBICOM)*, pages 50–61, Los Angeles, CA, USA, 2006.

[16] David Hadaller, Srinivasan Keshav, Tim Brecht, and Shubham Agarwal. Vehicular opportunistic communication under the microscope. In *Proceedings of the International Conference on Mobile systems, Applications, and Services (MobiSys)*, pages 206–219, San Juan, Puerto Rico, 2007.

[17] Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen Miu, Eugene Shih, Hari Balakrishnan, and Samuel Madden. Cartel: a

distributed mobile sensor computing system. In *Proceedings of ACM SenSys*, pages 125–138, Boulder, CO, 2006.

[18] Dirk Reichardt, Maurizio Miglietta, Lino Moretti, Peter Morsink, and Wolfgang Schulz. Cartalk 2000: safe and comfortable driving based upon inter-vehicle-communication. In *Proceedings of the IEEE IV Symposium*, pages 545–550, June 2002.

[19] Jing Zhao and Guohong Cao. Vadd: Vehicle-assisted data delivery in vehicular ad hoc networks. *Proceedings of IEEE Transactions on Vehicular Technology*, 57(3):1910–1922, May 2008.

[20] Gökhan Korkmaz, Eylem Ekici, and Füsun Ö zgüner. An efficient fully ad-hoc multi-hop broadcast protocol for inter-vehicular communication systems. In *Proceedings of IEEE ICC*, pages 423–428, June 2006.

[21] Gökhan Korkmaz, Eylem Ekici, and Füsun Ö zgüner. Urban multi-hop broadcast protocol for inter-vehicle communication systems. In *Proceedings of ACM VANET*, pages 76–85, Philadelphia, PA, October 2004.

[22] Christian Lochert, Björn Scheuermann, Christian Wewetzer, Andreas Luebke, and Martin Mauve. Data aggregation and roadside unit placement for a VANET traffic information system. In *Proceedings of ACM VANET*, pages 58–65, San Francisco, CA, September 2008.

[23] Per Enge. Retooling the global positioning system. *Scientific American*, May 2004.

[24] Jean-Pierre Hubaux, Srdan Capkun, and Jun Luo. The security and privacy of smart vehicles. *IEEE Security and Privacy Magazine*, 2(3):49–55, 2004.

[25] Maxim Raya and Jean-Pierre Hubaux. The security of vehicular ad hoc networks. In *Proceedings of the ACM SASN*, pages 11–21, Alexandria, VA, November 2005.

[26] JiST/SWANS. http://jist.ece.cornell.edu, 2004.

[27] Martin Treiber. Longitudinal traffic model: IDM. `http://vwisb7.vkw.tu-dresden.de/~treiber/MicroApplet/IDM.html`, April 2006.

[28] Martin Treiber. Lane change model: MOBIL. `http://vwisb7.vkw.tu-dresden.de/~treiber/MicroApplet/MOBIL.html`, February 2007.

[29] Abdelmalik Bachir and Abderrahim Benslimane. A multicast protocol in ad hoc networks: Inter-vehicle geocast. In *Proceedings of IEEE VTC-Spring*, pages 2456–2460, Jeju Island, Korea, April 2003.

[30] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TAG: a tiny aggregation service for ad-hoc sensor networks. *ACM SIGOPS Operating Systems Review*, 36(SI):131–146, 2002.

[31] Bartosz Przydatek, Dawn Song, and Adrian Perrig. SIA: Secure information aggregation in sensor networks. In *Proceedings of ACM SenSys*, pages 255–265, Los Angeles, CA, November 2003.

[32] Yi Yang, Xinran Wang, Sencun Zhu, and Guohong Cao. SDAP: A secure hop-by-hop data aggregation protocol for sensor networks. In *Proceedings of ACM MobiHoc*, pages 356–367, Florence, Italy, May 2006.

[33] Tamer Nadeem, Sasan Dashtinezhad, Chunyuan Liao, and Liviu Iftode. TrafficView: Traffic data dissemination using car-to-car communication. *ACM Mobile Computing and Communications Review (MC2R), Special Issue on Mobile Data Management*, 8(3):6–19, July 2004.

[34] Christian Lochert, Björn Scheuermann, and Martin Mauve. Probabilistic aggregation for data dissemination in VANETs. In *Proceedings of ACM VANET*, pages 1–8, Montreal, Canada, September 2007.

[35] Bo Yu, Jiayu Gong, and Cheng-Zhong Xu. Catch-up: A data aggregation scheme for VANETs. In *Proceedings of ACM VANET*, pages 49–57, San Francisco, CA, September 2008.

[36] Hanan Saleet and Otman Basir. Location-based message aggregation in vehicular ad hoc networks. In *Proceedings of IEEE AutoNet*, pages 1–7, Washington, DC, November 2007.

[37] C. L. Robinson, L. Caminiti, D. Caveney, and K. Laberteaux. Efficient coordination and transmission of data for cooperative vehicular safety applications.

In *Proceedings of ACM VANET*, pages 10–19, Los Angeles, CA, September 2006.

[38] Maxim Raya, Adel Aziz, and Jean-Pierre Hubaux. Efficient secure aggregation in VANETs. In *Proceedings of ACM VANET*, pages 67–75, Los Angeles, CA, September 2006.

[39] Maxim Raya, Panos Papadimitratos, and JEAN-PIERRE Hubaux. Securing vehicular communications. *IEEE Wireless Communications Magazine*, 13(5):8–15, October 2006.

[40] Srdjan Capkun and Jean-Pierre Hubaux. Secure positioning of wireless devices with application to sensor networks. In *Proceedings of the Annual Joint Conference Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 3, pages 1917–1928, Miami, FL, March 2005.

[41] Bryan Parno and Adrian Perrig. Challenges in securing vehicular networks. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, College Park, MD, 2005.

[42] Frederik Armknecht, Andreas Festag, Dirk Westhoff, and Ke Zeng. Cross-layer privacy enhancement and non-repudiation in vehicular communication. In *Proceedings of the IEEE International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, Bern, Switzerland, 2007.

[43] Jong Youl Choi, Philippe Golle, and Markus Jakobsson. Tamper-evident digital signature protecting certification authorities against malware. In *Proceedings of the IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC)*, pages 37–44, Indianapolis, IN, 2006.

[44] Naveen Sastry, Umesh Shankar, and David Wagner. Secure verification of location claims. In *Proceedings of ACM WiSe*, pages 1–10, San Diego, CA, 2003.

[45] Tina Suen and Alec Yasinsac. Peer identification in wireless and sensor networks using signal properties. In *Proceedings of the IEEE WSNS*, pages 1–8, Washington, DC, November 2005.

[46] Tim Leinmüller, Elmar Schoch, Frank Kargl, and Christian Maihöfer. Influence of falsified position data on geographic ad-hoc routing. In *Proceedings of European Workshop on Security and Privacy in Ad hoc and Sensor Networks (ESAS)*, 2005.

[47] Tim Leinmüller and Elmar Schoch. Greedy routing in highway scenarios: The impact of position faking nodes. In *Proceedings of the Workshop On Intelligent Transportation (WIT)*, 2006.

[48] Philippe Golle, Dan Greene, and Jessica Staddon. Detecting and Correcting Malicious Data in VANETs. In *Proceedings of ACM VANET*, pages 29–37, Philadelphia, PA, October 2004.

[49] Gongjun Yan, Stephan Olariu, and Michele C. Weigle. Providing VANET security through active position detection. *Computer Communications: Special Issue on Mobility Protocols for ITS/VANET*, pages 2883–2897, 2008.

[50] Bin Xiao, Bo Yu, and Chuanshan Gao. Detection and localization of sybil nodes in VANETs. In *Proceedings of ACM DIWANS*, pages 1–8, Los Angeles, CA, 2006.

[51] Tiffany Hyun-Jin Kim, Ahren Studer, Rituik Dubey, Xin Zhang, Adrian Perrig, Fan Bai, Bhargav Bellur, and Aravind Iyer. Vanet alert endorsement using multi-source filters. In *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking*, VANET '10, pages 51–60, New York, NY, USA, 2010. ACM.

[52] Yu-Chee Tseng, Sze-Yao Ni, and En-Yu Shih. Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network. *IEEE Transactions on Computers*, 52:545–557, 2003.

[53] Chunyu Hu, Yifei Hong, and Jennifer Hou. On mitigating the broadcast storm problem with directional antennas. In *Proceedings of IEEE ICC*, pages 104–110, Anchorage, AK, May 2003.

[54] NawaPorn Wisitpongphan, Ozan Tonguz, Jayendra Parikh, Priyantha Mudalige, Fan Bai, and Varsha Sadekar. Broadcast storm mitigation techniques in vehicular ad hoc networks. *IEEE Wireless Communications*, 14(6):84–94, December 2007.

[55] Linda Briesemeister and Günter Hommel. Role-based multicast in highly mobile but sparsely connected ad hoc networks. In *Proceedings of ACM MobiHoc*, pages 45–50, Boston, MA, August 2000.

[56] Marc Torrent-Moreno, Daniel Jiang, and Hannes Hartenstein. Broadcast reception rates and effects of priority access in 802.11-based vehicular ad-hoc networks. In *Proceedings of ACM VANET*, pages 10–18, Philadelphia, PA, October 2004.

[57] Ozan Tonguz, NawaPorn Wisitpongphan, Fan Bai, Priyantha Mudalige, and Varsha Sadekar. Broadcasting in VANET. In *Proceedings of IEEE MOVE*, pages 7–12, Anchorage, AK, May 2007.

[58] Stefan Krauß. *Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics*. PhD thesis, Mathematisches Institut, Universität zu Köln, 1998.

[59] Kai Nagel and Michael Schreckenberg. A cellular automaton model for freeway traffic. *J. Phys. I France*, 2:2221–2229, 1992.

[60] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, 2000.

[61] Arne Kesting, Martin Treiber, and Dirk Helbing. MOBIL: General lane-changing model for car-following models. In *Proceedings of the Transportation Research Board Annual Meeting*, Washington, DC, January 2007.

[62] Marco Fiore, Jérôme Härri, Fethi Filali, and Christian Bonnet. Understanding vehicular mobility in network simulation. In *Proceedings of the IEEE MoVeNet*, Pisa, Italy, October 2007.

[63] `http://sumo.sourceforge.net`.

[64] `http://canu.informatik.uni-stuttgart.de`.

[65] Xiang Zeng, Rajive Bagrodia, and Mario Gerla. Glomosim: a library for parallel simulation of large-scale wireless networks. *SIGSIM Simulation Digest*, 28(1):154–161, 1998.

[66] `http://www.omnetpp.org`.

[67] `http://isi.edu/nsnam/ns`.

[68] `http://www.scalable-networks.com`.

[69] `http://www.opnet.com`.

[70] Stuart Kurkowski, Tracy Camp, and Michael Colagrosso. MANET simulation studies: The incredibles. *ACM Mobile Computing and Communications Review (MC2R)*, 9(4):50–61, 2005.

[71] Frank Kargl and Elmar Schoch. Simulation of MANETs: A qualitative comparison between JiST/SWANS and ns-2. In *Proceedings of the International Workshop on System Evaluation for Mobile Platforms (MobiEval)*, pages 41–46, San Juan, PR, June 2007.

[72] Feliz K. Karnadi, Zhi H. Mo, and Kun-Chan Lan. Rapid generation of realistic mobility models for VANET. In *Proceedings of IEEE WCNC*, pages 2506–2511, Hong Kong, March 2007.

[73] Jérôme Härri, Fethi Filali, Christian Bonnet, and Marco Fiore. VanetMobiSim: Generating realistic mobility patterns for VANETs. In *Proceedings of ACM VANET*, pages 96–97, Los Angeles, CA, September 2006.

[74] David R. Choffnes and Fabián E. Bustamante. An integrated mobility and traffic model for vehicular wireless networks. In *Proceedings of ACM VANET*, pages 69–78, Cologne, Germany, September 2005.

[75] Amit Kumar Saha and David B. Johnson. Modeling mobility for vehicular ad-hoc networks. In *Proceedings of ACM VANET*, pages 91–92, Philadelphia, PA, October 2004.

[76] Rahul Mangharam, Daniel S. Weller, Daniel D. Stancil, Ragunathan Rajkumar, and Jayendra S. Parikh. GrooveSim: a topography-accurate simulator for geographic routing in vehicular networks. In *Proceedings of ACM VANET*, pages 59–68, Cologne, Germany, September 2005.

[77] S. Y. Wang, C. L. Chou, Y. H. Chiu, Y. S. Tzeng, M. S. Hsu, Y. W. Cheng, W. L. Liu, and T. W. Ho. NCTUns 4.0: An integrated simulation platform

for vehicular traffic, communication, and network researches. In *Proceedings of IEEE VTC-Fall*, pages 2081–2085, Baltimore, MD, October 2007.

[78] Michal Piorkowski, Maxim Raya, Ada Lugo, Panos Papadimitratos, Matthias Grossglauser, and Jean-Pierre Hubaux. TraNS : realistic joint traffic and network simulator for vanets. *ACM Mobile Computing and Communications Review (MC2R)*, 12(1):31–33, 2008.

[79] Hadi Arbabi and Michele C. Weigle. Highway mobility and vehicular ad-hoc networks in ns-3. In *Proceedings of the Winter Simulation Conference*, Baltimore, MD, December 2010.

[80] Stephan Olariu and Michele C. Weigle, editors. *Vehicular Networks: From Theory to Practice*. CRC Press / Taylor & Francis, March 2009.

[81] US DoT, National Highway Traffic Safety Administration. Identify Intelligent Vehicle Safety Applications Enabled by DSRC. DOT HS 809 859, Vehicle Safety Communications Project Task 3 Final Report, March 2005.

[82] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ECDSA). *International Journal of Information Security*, 1(1):36–63, August 2001.

[83] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*, chapter 10. MIT Press, Cambridge, London, 2nd edition, 2001.

[84] Jijun Yin, Tamer ElBatt, Gavin Yeung, Bo Ryu, Stephen Habermas, Harihan Krishnan, and Timothy Talty. Performance evaluation of safety applications over DSRC vehicular ad hoc networks. In *Proceedings of ACM VANET*, pages 1–9, Philadelphia, PA, October 2004.

[85] SICK - Sensor Intelligence. http://www.sick.com.

[86] ibeo Automobile Sensor. http://www.ibeo-as.com.

[87] Robert A. MacLachlan and Christoph Mertz. Tracking of moving objects from a moving vehicle using a scanning laser rangefinder. In *Proceedings of IEEE ITSC*, pages 301–306, Toronto, Canada, September 2006.

[88] Fadi Fayad and Veronique Cherfaoui. Tracking objects using a laser scanner in driving situation based on modeling target shape. In *Proceedings of the IEEE IV Symposium*, pages 44–49, Istanbul, Turkey, June 2007.

[89] Stefano Basagni, Marco Conti, Silvia Giordano, and Ivan Stojmenovic, editors. *Mobile Ad Hoc Networking*. Wiley-IEEE Press, December 2003.

[90] Siddhartha Saha, Kamalika Chaudhuri, Dheeraj Sanghi, and Pravin Bhagwat. Location determination of a mobile device using IEEE 802.11b access point signals. In *Proceedings of IEEE WCNC*, pages 1987–1992, New Orleans, LA, March 2003.

[91] Ronald Miller and Qingfeng Huang. An adaptive peer-to-peer collision warning system. In *Proceedings of IEEE VTC-Spring*, pages 317–321, Birmingham, AL, May 2002.

[92] Tamer Nadeem, Pravin Shankar, and Liviu Iftode. A comparative study of data dissemination models for vanets. *Annual International Conference on Mobile and Ubiquitous Systems*, 0:1–10, 2006.

[93] NawaPorn Wisitpongphan, Fan Bai, Priyantha Mudalige, Varsha Sadekar, and Ozan Tonguz. Routing in sparse vehicular ad hoc wireless networks. *IEEE Journal on Selected Areas in Communications*, 25(8):1538–1556, October 2007.

[94] Berkeley highway lab (BHL). http://bhl.calccit.org/, 2006.

[95] Peter Deutsch. DEFLATE Compressed Data Format Specification version 1.3. RFC 1951, May 1996.

[96] Khaled Ibrahim and Michele C. Weigle. Optimizing CASCADE data aggregation for VANETs. In *Proceedings of the IEEE MoVeNet*, pages 724–729, Atlanta, GA, September 2008.

[97] Alok Nandan, Alok N, Saurabh Tewari, Shirshanka Das, Mario Gerla, and Leonard Kleinrock. Adtorrent: Delivering location cognizant advertisements to car networks. In *The Third International Conference on Wireless On Demand Network Systems and Services(WONS 2006)*, January 2006.

[98] Rimon Barr, Zygmunt Hass, and Robbert van Renesse. *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad hoc Wireless, and Peer-to-Peer Networks*, chapter 19, pages 297–311. CRC Press, 2005.

[99] Khaled Ibrahim and Michele C. Weigle. ASH: Application-aware SWANS with highway mobility. In *Proceedings of IEEE MOVE*, Phoenix, AZ, April 2008.

[100] Khaled Ibrahim and Michele C. Weigle. p-IVG: Probabilistic inter-vehicle geocast for dense vehicular networks, September 2008. In Submission.

APPENDIX A

ASH SIMULATOR

Most VANET simulators do not allow for feedback between the vehicle mobility model and the network simulator. This limits the realistic simulation of safety and traffic information applications that might cause drivers to change their routes. Because the feedback between the vehicle mobility model and the network simulator is one of the key requirements for CASCADE we chose to develop extensions to the Scalable Wireless Ad hoc Network Simulator (SWANS) [26, 98] to make it feedback-capable in addition to adding some other features. Our SWANS modules, which we collectively call ASH (Application-aware SWANS with Highway mobility) [99], make several contributions:

- Methods to provide feedback between the application layer and the mobility model, allowing for the needed two-way communication

- A customizable highway topology, supporting multiple lanes, entrances, and exits

- An implementation of the Intelligent Driver Model (IDM) [60] car-following model

- An implementation of the MOBIL [61] lane change model

- An implementation of the Inter-Vehicle Geocast (IVG) [29] broadcasting technique, as well as an implementation of the probabilistic IVG (p-IVG) [100] enhancement to IVG

- An enhanced node model allowing for the addition of non-communicating vehicles, road-side units, and obstacles to a simulation

- Statistical and logging facilities for reporting simulation metrics

These additions to the scalable SWANS simulator will allow for realistic VANET simulations of important safety and traffic information applications.

A.1   NETWORK MODEL

There are large number of network simulators currently being used in the network research community. Some of these are open source like GloMoSim [65], OMNet++ [66], and ns-2 [67], while the others are commercial like QualNet [68] and OPNET [69]. Of these, ns-2 is the most widely-used in the mobile ad-hoc network research community [70]. But, ns-2 suffers from problems when simulating large numbers of nodes.

SWANS [26] was developed to be a scalable alternative to ns-2 for simulating wireless networks. Based on comparisons [71] between SWANS, GloMoSim and ns-2, SWANS was found to be the most scalable, the most efficient in memory usage, and fastest in runtime. In addition, Kargl *et al.* [71] validated the network model in SWANS against ns-2. They showed that along with better performance, SWANS delivered similar results as ns-2, at least for the network components that were implemented in both.

A.2   MOBILITY MODEL

There are four basic mobility models provided in SWANS: static, teleport, random walk, and random waypoint. These models are suitable for representing the mobility of mobile ad-hoc network nodes, but they are not applicable to vehicular mobility.

Vehicular traffic models are typically classified into three categories based on traffic granularity: macroscopic, mesoscopic, and microscopic. Macroscopic models deal with traffic as flows, while mesoscopic models are concerned with the movement of whole platoons of vehicles. Microscopic models handle the movement of each vehicle in the traffic flow, thus they are the most suitable for VANET applications.

Many microscopic models have been developed. The most widespread ones are the SK model [58], the Cellular Automaton (CA) model [59], and the IDM/MOBIL model [60] [61]. Fiore *et al.* [62] evaluated the realism of several mobility models and recommended that only realistic car-following models, such as IDM, be used in VANET simulations. Thus, we have chosen to extend SWANS by implementing IDM/MOBIL to control vehicular mobility (acceleration, deceleration, and lane change). In the future, we plan to implement other car-following models in ASH to give the research community the freedom to select different models.

TABLE 17: IDM Variables [27]

Symbol	Definition
v_0	Desired velocity when driving on a free road
T	Desired safety headway when following other vehicles
a	Acceleration in everyday traffic
b	Comfortable braking deceleration in everyday traffic
s_0	Minimum bumper-to-bumper distance to the front vehicle
δ	Acceleration exponent

TABLE 18: MOBIL Variables [28]

Symbol	Definition
p	Politeness factor
b_{safe}	Maximum safe deceleration
a_{thr}	Threshold (must be below IDM parameter a)
δ_b	Bias to the right lane

The Intelligent Driver Model (IDM), developed by Treiber *et al.* [60], is a car-following model. The general characteristic of car-following models is that the traffic state at any time is characterized by the positions, velocities, and the lane indices of all vehicles. Car-following models are also known as safety-distance models, because the inter-vehicle distance cannot be less than a specific threshold. Thus, there are no collisions when using car-following models.

IDM has several parameters that can be adapted to represent different types of drivers as shown in Table 17. For instance, aggressive drivers may have an increased desired velocity v_0, acceleration a, and deceleration b, with a decreased safety headway T (distance between them and the car directly ahead).

Minimizing Overall Braking deceleration Induced by Lane changes (MOBIL), developed by Kesting *et al.* [61], controls how vehicles change lanes. According to this model, there are two criteria that must be satisfied before any vehicle performs a lane change. The *incentive criterion* states that the targeted new lane must be more attractive than the current lane, and the *safety criterion* states that this lane change should be done safely. In other words, the lane change should be both advantageous to the driver and should not lead to an accident or discomfort to nearby drivers.

The variables used to customize MOBIL are given in Table 18. The politeness factor, p, allows MOBIL to model various types of drivers, from altruistic ones who favor others over themselves to malicious drivers who only want to thwart others. Note that no matter the politeness factor, the safety criterion must always be obeyed. As the exit destination is predetermined for each vehicle at its creation time, the politeness factor for each vehicle's driver can change dynamically as the vehicle approaches its destination exit.

A.3   ASH EXTENSIONS

SWANS is a scalable wireless network simulator that has been built on top of JiST [26], Java in Simulation Time. As SWANS uses JiST as its run platform, it inherits all JiST's capabilities of being very efficient in memory usage, faster than the others simulator and even more scalable [71]. SWANS has been architected in component-wise fashion, so it has many components that implement different categories of applications; networking, routing and media access protocols; radio transmission, reception and noise models; signal propagation and fading models; and node mobility models. Each of these components is encapsulated as a JiST entity, which makes it self-contained. The interaction between these components is performed through event-based interfaces. Based on SWANS's components architecture, designing a new competent to extend SWANS or to replace one of its existing components is straight forward. Also, the integration of any newly implemented component with the other components in SWANS is seamless as long as the new component adheres to the component design guidelines and implements the pre-defined interfaces [26].

Based on what we mentioned above, we selected SWANS to be the code base for our new vehicular simulator, ASH [99]. Here we present an overview of the features included in ASH and each of these features is explained in details in the following sections:

- Two-way communication

- Customizable highway topology

- Enhanced node model

- Implementations of IDM and MOBIL

- Implementations of Inter-Vehicle Geocast (IVG) and probabilistic IVG (*p*-IVG)

- Statistical and logging facilities

A.3.1 Two-Way Communication

The ability for two-way communication between the network and the application is the most important feature we have added in ASH. Through this feature, the application running at each vehicle can affect the vehicle's mobility. This two-way communication is achieved by providing the application layer with two sets of primitives. The first set of primitives allows the application layer to retrieve the received messages from the network stack in the network sub-simulator. The application should provide a message handler and associate it with a specific port. When a new message arrives at the specified port, the network stack will deliver the message to the appropriate handler.

Based on the information received from the network, the application can choose to affect the vehicle's mobility. Therefore, the application can override the IDM/MOBIL normal behavior through the second set of primitives which is a set of mobility control primitives e.g., *accelerate*, *decelerate*, *change-lane*. This means the mobility decisions will be adaptive according to the information available at the application level. In Figure 102, we show an example cooperative collision avoidance system where such control by the application is needed. The driver of car C receives an alert about a suddenly braking vehicle ahead (car A). Since car B is still outside of the safety distance of car C, IDM would not cause car C to slow down. But, the driver might choose to begin slowing down early. The two-way communication provided by ASH would allow the application running on the car C to emulate decisions that a driver might make. This feature is the most important needed feature in any simulation used by those interested in developing safety-related applications for VANET.

A.3.2 Highway Topology

ASH supports a simplified design of a highway segment. The segment can be configured as a one-way or two-way highway. The key point of supporting the two-way highway option is that some VANET applications require the vehicles in the opposite directions to communicate with each other. The characteristics of the highway segment can be specified in a configuration file. The supported characteristics are
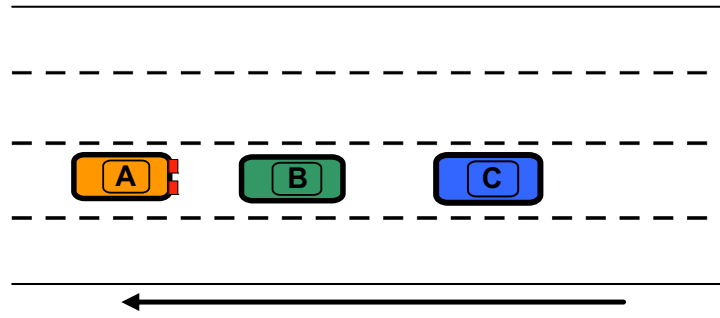
FIG. 102: Cooperative Collision Avoidance System Example

segment length, number of directions, number of lanes, and number of entries and exits, as well as their locations along the segment. Although this highway design is simple, it satisfies most of the requirements needed to implement scenarios in highway-based VANET applications.

In Figure 103 we show an example highway configuration file; the corresponding graphical representation is shown in Figure 104. In this example, we setup a highway segment of length ten kilometers with two directions and four lanes in each direction. The keyword `Highway` denotes the beginning of the configuration. This is followed by the specification of the highway length and number of lanes (`Length`, `Two-Ways`, and `Lanes-Count`). Once the basic characteristics of the highway have been set up, the number and position of exits and entrances can be defined. Each highway direction is specified separately, using the `Direction` and `Opposite Direction` keywords. Entries are specified using the `Entry` keyword and exits by the `Exit` keyword. Both entries and exits are defined by their ID, location (offset from the beginning of the highway), and type (left or right), defined by the keywords `ID`, `Location`, and `Type`, respectively. The highway can be set to have only one direction by setting `Two-Ways: no`. In this case, the characteristics of the `Opposite-Direction` should not be included in the configuration file.

A.3.3 Node Model

Currently in SWANS, all nodes in a simulation have the same mobility, *i.e.* nodes can all be mobile or all be static. We have enhanced the node model to allow each node to have its own mobility state, so that a mix of mobile and static nodes can be present in the same simulation. By default in SWANS, all nodes communicate. We

```
Highway
{
  Length: 10          // in km
  Two-Ways : yes      // yes or no
  Lanes-Count : 4     // Number of Lanes per direction

  Direction
  {
    Exit
    {
      ID : 1
      Location : 1    // Offset in km from the highway start
      Type: R         // R=Right Exit ; L=Left Exit
    }

    Exit
    {
      ID : 2
      Location : 3    // Offset in km from the highway start
      Type: R         // R=Right Exit ; L=Left Exit
    }

    Entry
    {
      ID : 1
      Location : 1.5  // Offset in km from the highway start
      Type: R         // R=Right Entry ; L=Left Entry
    }

    Entry
    {
      ID : 2
      Location : 3.5  // Offset in km from the highway start
      Type: R         // R=Right Entry ; L=Left Entry
    }
  }

  Opposite-Direction  // exists only if Two-Ways : Yes
  {
    Exit
    {
      ID : 1
      Location : 1    // Offset in km from the highway start
      Type: R         // R=Right Exit ; L=Left Exit
    }

    Exit
    {
      ID : 2
      Location : 3    // Offset in km from the highway start
      Type: R         // R=Right Exit ; L=Left Exit
    }

    Entry
    {
      ID : 1
      Location : 1.5  // Offset in km from the highway start
      Type: R         // R=Right Entry ; L=Left Entry
    }

    Entry
    {
      ID : 2
      Location : 3.5  // Offset in km from the highway start
      Type: R         // R=Right Entry ; L=Left Entry
    }
  }
}
```

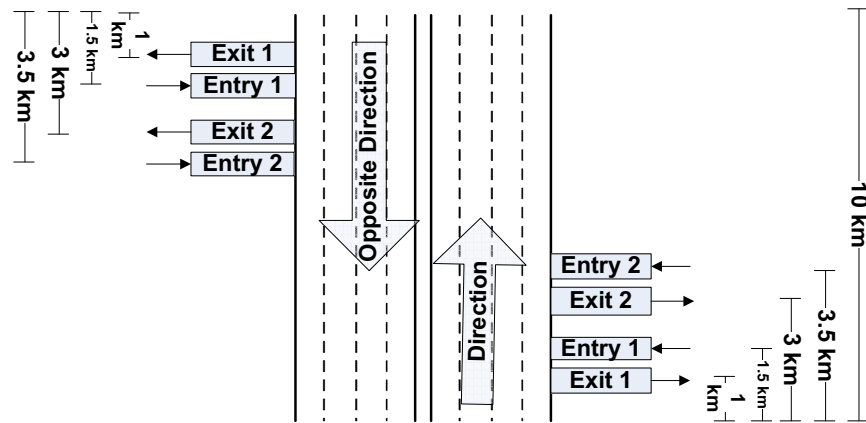FIG. 103: Example Highway Configuration File

FIG. 104: Graphical Representation of Example Highway Configuration

have added the option for nodes to be silent. This results in four possible node types that can be present in the same simulation (illustrated in Figure 105):

- *Mobile Communicating Node*: Represents a participating vehicle that should execute a user-defined application. This application specifies how the vehicle should behave.

- *Mobile Silent Node*: Represents a non-participating vehicle that should execute a null application so that it will not be able to send or receive any messages.

- *Static Communicating Node*: Represents road-side infrastructure that should execute a user-defined application. This application specifies how the road-side unit should behave. Also, this kind of node may have different physical layer characteristics, *e.g.* transmission power, than the mobile communicating nodes,

- *Static Silent Node*: Represents a road obstacle that should execute a null application.

In particular, the addition of the mobile silent node is important to allow testing of protocols under different penetration rates where not all vehicles are equipped with communicating devices. The location of the static silent node can either be pre-determined before running the simulation or can be determined at runtime, in order to simulate an accident, for example.
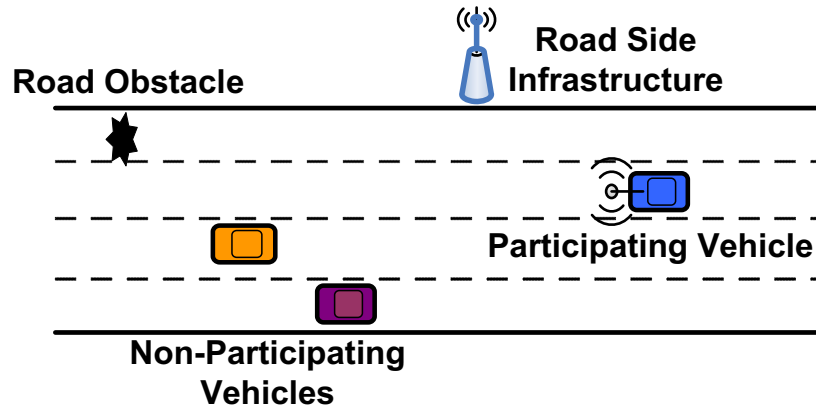
FIG. 105: Possible Nodes Types in ASH

A.3.4 Enhanced Dissemination Techniques

Because most VANET applications use flooding-based techniques to disseminate data, we implemented the Inter-Vehicle Geocast protocol (IVG) [29] in ASH. In IVG, whenever a vehicle receives a message to broadcast, it starts a timer. When the timer value expires, it broadcasts the message only if it has not heard another node re-broadcasts the same message. The time value T_x for vehicle x is

$$T_x = T_{max} \cdot \frac{(R^\epsilon - D_{sx}^\epsilon)}{R^\epsilon},$$

where R is the transmission range and D_{sx} is the distance between vehicle x and vehicle s, the sender of the message. The authors of IVG suggest using $\epsilon = 2$ to generate a uniform timer value between $[0, T_{max}]$ where $T_{max} = 200$ ms. ASH also supports our modification to IVG, p-IVG, which we will describe in Chapter VI. As part of our extension to ASH's extensions, we have added a new enum, $Dissemination\_Techniques$, to define the implemented dissemination techniques flooding, IVG and p-IVG, and as part of the simulation startup settings, we should set the $Dissemination\_Technique$ in the configuration file to one of the $Dissemination\_Techniques$ enum values.

A.3.5 Supporting Utilities

We have also implemented statistical and logging utilities to support simulations in ASH. The utilities provide information about the simulation entities at different granularities. This information can be retrieved at the simulation level, lane level,

vehicle level, or message type level. For example, to log the mobility traces for all the vehicles in the simulation, the mobility log switch should be turned on at the simulation level (`Mobility_Log = true`). The mobility traces of all vehicles will be stored in a single file, set by `Mobility_Log_File = filename`. To log the mobility traces for a specific vehicle, the mobility log switch for this vehicle should be turned on when the vehicle is created (`Vehicle.Mobility_Log = true`). In this case the log file in which the vehicle's mobility traces will be stored should have the vehicle ID as part of its name (`Vehicle.Mobility_Log_File = filename + Vehicle.ID`). Similarly, any message type can be logged at either the vehicle level or the simulation level. Also, the data can be projected to the time domain, meaning that the information can be provided for the whole simulation time, any simulation period, or a specific simulation instance. The provided utilities are customizable so they can be turned on or off according to the user's needs.

The statistics utility provides statistics about all possible events that can occur in the simulation. For example, it keeps track of the number of messages sent or received for each message type. This count is maintained at both the vehicle level and at the simulation level. Moreover, the statistics utility keeps track of the number of vehicles in the simulation and how many of them are in each lane. This utility works automatically without the needing to be turned on or off. All of the statistics are stored in variables and arrays that are accessible system-wide. For example, the number of current vehicles in direction dir and lane $lane$ is stored in `Vehicles_Per_Lanes`$[dir]$ $[lane]$. As another example, the variable `Vehicles.`$[V\_ID]$`.Messages_Sent` stores the total messages sent by vehicle V_ID.

A.4 SUMMARY

In this appendix we have presented ASH, comprising our enhancements to the SWANS network simulator to provide feedback between the application layer and mobility model, allowing for two-way communication. We have also implemented the IDM/MOBIL mobility model as well as enhancing SWANS' node model and adding a highway topology. ASH also has a complete implementation for the three dissemination techniques, flooding, IVG and p-IVG that we used in various experiments. The source code of ASH is available through our project website: `http://www.cs.odu.edu/~vanet/`

VITA

Khaled Ibrahim

Department of Computer Science

Old Dominion University

Norfolk, VA 23529

Khaled received his Bachelors Degree in Computer Science with honors from Alexandria University in 1999. In 2004, he received the Masters Degree in Computer Science from Alexandria University. His Masters thesis was focused on enhancing the performance of content-based switching and was titled "Modified Protocol for Content-Based Switching". He joined the PhD program in the Computer Science Department at Old Dominion University in May 2006. Since he joined the PhD program, he was supported by the Dominion Scholarship. In 2008, he received the outstanding teaching assistant award given by College of Science for his excellent teaching and continuing good work in the Spring 2008 semester. He joined the VANET research group in April 2007 and decided to pursue his dissertation under the supervision of Dr. Michele C. Weigle in Vehicular Ad-Hoc Networks. Khaled's research interests include a wide variety of interesting vehicular ad-hoc networks related topics such as lossless data aggregation, efficient data dissemination, communication discontinuity detection and recovery, and vehicular related applications. His research work and ideas were well received by the VANET research community, with 11 publications in various IEEE and ACM conferences and workshops.

Typeset using LaTeX.