2021

Routing and Applications of Vehicular Named Data Networking

Bassma G. Aldahlan
University of Kentucky, baldahlan@hotmail.com
Author ORCID Identifier:
https://orcid.org/0000-0002-0548-4353
Digital Object Identifier: https://doi.org/10.13023/etd.2021.309

Bassma G. Aldahlan, Student

Dr. Zongming Fei, Major Professor

Dr. Zongming Fei, Director of Graduate Studies

Routing and Applications of Vehicular Named Data Networking

DISSERTATION

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the College of Engineering at the University of Kentucky

By
Bassma Aldahlan
Lexington, Kentucky
Director: Dr. Zongming Fei, Professor of Computer Science
Lexington, Kentucky
2021

https://orcid.org/0000-0002-0548-4353

ABSTRACT OF DISSERTATION

Routing and Applications of Vehicular Named Data Networking

Vehicular Ad hoc NETwork (VANET) allows vehicles to exchange important information among themselves and has become a critical component for enabling smart transportation. In VANET, vehicles are more interested in content itself than from which vehicle the content is originated. Named Data Networking (NDN) is an Internet architecture that concentrates on what the content is rather than where the content is located. We adopt NDN as the underlying communication paradigm for VANET because it can better address a plethora of problems in VANET, such as frequent disconnections and fast mobility of vehicles. However, vehicular named data networking faces the problem of how to efficiently route interest packets and data packets.

To address the problem, we propose a new geographic routing strategy of applying NDN in vehicular networks with Delay Tolerant Networking (DTN) support, called GeoDTN-NDN. We designed a hybrid routing mechanism for solving the flooding issue of forwarding interest packets and the disruption problem of delivering data packets. To avoid disruptions caused by routing packets over less-traveled roads, we develop a new progressive segment routing approach that takes into consideration how vehicles are distributed among different roads, with the goal of favoring well-traveled roads. A novel criterion for determining progress of routing is designed to guarantee that the destination will be reached no matter whether a temporary loop may be formed in the path.

We also investigate applications of vehicular named data networking. We categorize these applications into four types and design an NDN naming scheme for them. We propose a fog-computing based architecture to support the smart parking application, which enables a driver to find a parking lot with available parking space and make reservation for future parking need. Finally we describe several future research directions for vehicular named data networking.

KEYWORDS: Named Data Networking, Vehicular Named Data Networking, Routing, NDN applications, Smart parking Application

Bassma Aldahlan

May 30, 2021

Routing and Applications of Vehicular Named Data Networking

By

Bassma Aldahlan

Zongming Fei

Director of Dissertation

Zongming Fei

Director of Graduate Studies

May 30, 2021

Date

To my parents, husband, brother, and sons
I couldn't have done it without you

ACKNOWLEDGMENTS

This dissertation could not have been written without the support of the following people and institutions.

I would like to express my deepest gratitude to my advisor, Professor Zongming Fei, for his ongoing guidance, patience, motivation, and support. He offered me his knowledge and helpful comments throughout my time in the Ph.D. program. During my studies, it was a great honor to work with him and receive his guidance. I am extremely thankful to him for his empathy, kindness, and great sense of humor. This dissertation would not have been completed without his support and advice. I also gratefully acknowledge the assistance of my committee members and their comments, assessments, and questions.

I am also grateful to the Yanbu University College and the Saudi Arabian Cultural Mission (SACM) for giving me the opportunity to pursue my studies. I would like to extend my sincere thanks to them for their financial support. Given the responsibility and trust they have bestowed upon me, I hope to serve my country with loyalty and sincerity.

Special thanks to my friend in the computer science department, Zainab Almula, for being with me through the easy times and the hard times, sharing the same aspirations and difficulties. I would also like to thank Cory Siler for his friendship and support.

There are many people outside of school who have supported me with their love and patience. I am greatly indebted to my parents, Badoor and Ghazi Aldahlan, for all the help and love they had given me from when I was a child to this moment. I will never forget my parents' words that motivated me to pursue my studies. I will also never forget their prayers and their help with my kids; they have proven that they are truly the best parents.

I must also express my very profound gratitude to my husband, Awad Alomari, for giving me love and support. Awad, you are the most wonderful husband. You were and are still with me, supporting me in all the steps to complete this dissertation. Whenever I faced an obstacle or challenging time, you were always there to offer solutions, love, and kindness. You have been happy when I succeed, and you have supported me during my struggles. You have been immense support while I wrote this dissertation by taking a leave from your job and taking care of our kids. I will never forget your love and support, and there are no words to express my deep, heartfelt thanks.

I would also like to thank my youngest brother, Abdullah Aldahlan, for being with me when I started the Ph.D. program. Abdullah, I thank you for your patience, high spirits, and all your supports. I appreciate your financial help as well as your help with childcare. Whenever I needed help, I found you there next to me.

Finally, many thanks to my sons, Firas, Maher, and Ghazi Alomari, for giving me the time to write this dissertation. My apologies for the times we spent apart. My kiddos, it was so hard for my heart to have you away from me, but I had no other choices. After completing this dissertation, I would like to dedicate this honor to you with all my love. You are all my heart and my breath, so without you, I could not live.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Vehicular Ad hoc NETwork (VANET) is a communication architecture that has emerged from Mobile Ad hoc NETwork (MANET) to support communication among vehicles. Vehicles in a VANET organize themselves in an ad hoc fashion with regards to the neighbor relationship among them and help each other by forwarding packets from other vehicles to the desired destination. VANET makes traffic information and other important information available to vehicles and has become a critical component for enabling smart transportation. It can improve road safety and traffic efficiency and provide other information such as parking, gas station, and entertainment information. The quality of experience for using vehicles can be greatly improved with the assistance of VANET.

There are several variants of VANET. If each participating vehicle is allowed to communicate only with other vehicles via wireless connections and exchange information, it is called Vehicle to Vehicle (V2V) communication. If a participating vehicle can also connect to Road Side Units (RSUs), which are considered part of the infrastructure, it is called Vehicle to Infrastructure (V2I) communication. Finally, Vehicle to Everything (V2X) communication means that a vehicle can additionally communicate with any entity, including other vehicles, infrastructures, gateways, pedestrians, and other Internet of Things (IoT) devices.

Implementing VANETs with the traditional IP-based network is challenging because frequent motions of vehicles result in rapid changes in the topology, and network connectivity is short-lived and intermittent. In an IP-based network, the sender needs to specify the IP address of the destination. It works fine in the traditional wired network because the IP address is associated with the logical location inside networks, and we have a way to forward the packets to the location. In VANETs, the address associated with a vehicle will not give us an efficient way to forward packets to the destination because the IP address no longer gives the clue of the location due to the mobility of the vehicles.

Named Data Networking (NDN) [1] is a recently proposed future Internet architecture based on the Information-Centric Networking paradigm [2]. NDN specifies what the content is rather than where the content comes from, focusing on the name of the content instead of the address of the packet. NDN does not need to route packets to a specific address but only needs to find the producer of the content or whoever caches the content in the network. Because of caching of the data inside the NDN network, a mobile node can get the data from any inside network cache without the need to specify the IP address of the producer. Therefore, it can naturally support the mobility of the producer and is more suitable for VANETs.

In NDN, the specification of the name of content uses a hierarchical, variable-length, and human-readable structure, similar to URLs. A consumer interested in some content sends out an *interest packet*, which contains the name prefix of the content with other parameters. The Interest packet is usually sent out by broadcast and thus may cause the flooding problem in the network. If either a producer or an intermediate router with cached content receives the Interest packet, it will send back the matched *data packet* to the consumer along the reverse path the interest packet has traveled.

NDN supports the mobility of consumers and producers because they are not assigned any IP addresses that have to be attached to the Internet at some fixed point. They can move to different places without worrying about changing the addresses – the problem with the current IP-based network. NDN also enables in-network caching of content so that the content can be fetched from any available cache, even though the producers may have moved to different places at the time. NDN maintains self-certification of every data packet. Each data packet is signed by the producer with its private key, and therefore, can be verified by consumers who receive the data packet. Consumers verify the signature of the received data packets to check the integrity and authenticity by using the key locator, which includes the signing key information, carried in the data packet to get the corresponding public key certificate [3]. This is important for VANET because we often need to make sure that packets forwarded by some intermediate nodes in VANET come from a valid source. All these features make NDN a good fit as a fundamental architecture for the VANET environment [4].

While using NDN for VANET is a promising approach, it also brings several issues. Early work on Vehicular Data Named Networking (V-NDN) [4] depends on broadcasting interest packets to find the producer of the required content. This can cause flooding issues inside the network. For those applications that focus on location-associated information, such as traffic information, geographical routing can help direct the interest packet to its desired destination. For geographical routing, there is a "local maximum" problem when a node may not find a neighbor closer to the destination than itself. While perimeter routing partially solved the problem, all these solutions are proposed in the context of MANET, where they consider that mobile nodes form the topology of a graph, in which each node is a vertex, and the communication link between two nodes is an edge. In VANETs, mobile vehicles have to travel on the road. The topology formed by considering the city map is at a higher abstraction level, where intersections are vertexes and roads represent edges. The question becomes how to take advantage of this and make the geographical routing more efficient.

Another issue with VANETs is the high mobility of vehicles. In NDN, the data packet for satisfying an interest is sent to the consumers in the reserve path of what the interest packet traveled. The path was set up when the interest packet was sent from the consumer to the producer. Due to the mobility in VANET, the vehicles on the reverse path may have moved to other places. The problem becomes how we can still get the data packet back to the consumers. The high mobility may even lead to partitioning of VANETs when a vehicle cannot find any other vehicles in its transmission range. This can interrupt the transmission of both interest packets and data packets. Routing in these scenarios becomes even more challenging.

To address these issues, we propose two approaches in this thesis. One is based on geographical routing. We enhance it with Delay Tolerant Networking (DTN) to deal with the network partition problem by allowing a vehicle to carry the packets for a short period of time. We propose a hybrid geographic routing solution with restricted greedy, greedy, perimeter, and DTN modes in packet forwarding. To deal with the interruption of forwarding of data packets, we update the data packets with location information so that they can be forwarded in the correct direction even if some intermediate nodes are dislocated. We also allow the consumers to update requests when their location changes.

The other approach is based on source routing. To use the road graph made available by the map on the vehicles, we propose a progressive segment routing approach that allows multiple segments to be formed based on the observed density information about roads, in order to choose the most efficient path for the packets. We need to make sure that progress is made when the routing path can be dynamically changed by multiple segments. The general idea is to choose well-traveled roads over less-traveled roads to reduce the failure rate of packet delivery. A novel criterion for determining the progress of routing is designed to guarantee that the destination will be reached even if a temporary loop may be formed in the path.

After dealing with the routing problems, we turn our focus on V-NDN applications. Many applications have been investigated, and they have different behaviors, features and characteristics. We categorize these applications and discuss their communication requirements. Some applications are more time sensitive than others and may need better quality of service (QoS). We present an NDN naming scheme for these applications and design a priority model to prioritize the delivery of packets based on their types and names. Finally, we develop a smart parking application to demonstrate how location-based routing approaches can be used in V-NDN applications. The application is designed based on the fog computing framework, which brings the computation and storage services close to the end-users and allows sensor data collected to be processed locally. The smart parking application enables drivers to query the availability of parking space and make reservations ahead of time.

1.1 Overview of NDN System Architecture

Since our work is based on the NDN architecture, we present an overview of NDN here. In NDN, names are used to identify the content and are the most critical part. NDN uses a hierarchical, variable-length, and human-readable structure to represent names. NDN can support many functionalities such as multicast, mobility, content distribution, and delay-tolerant networking. In NDN, there are two types of packets, interest packets and data packets. Interest packets are sent by a consumer and contain the name prefix of the content with other parameters. Usually, a consumer application uses consumer/producer API to request content. It sends an interest packet to the network to get the desired content, usually by broadcast. The data packet contains the name prefix of the content with some parameters plus the data. The data packet will be returned to the consumer if either the intermediate router or a producer has received the interest packet. The data packet is forwarded to the consumer using the reverse path of the matched interest packet.

The forwarding engine in NDN has three main data structures as shown in Figure 1.1: Content Store (CS), Pending Interest Table (PIT), and Forwarding Information Base (FIB).

- CS: CS is composed of the name of the content and cached data. NDN uses the caching ability in its intermediate nodes to reduce required upstream bandwidth and downstream latency. It can also increase the probability that data content can be

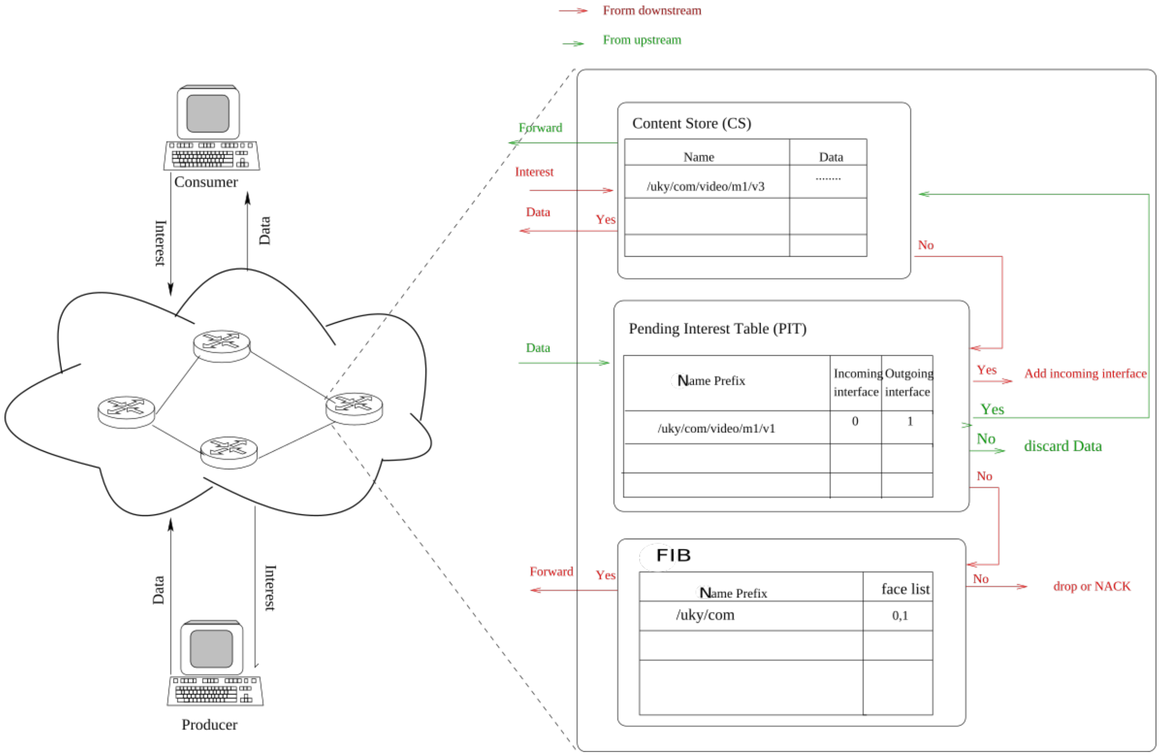

Figure 1.1: NDN Forwarding Engine

shared in the mobile environment. NDN packets are independent, self-identifying, and self-authenticating so that consumers can fetch the content from CS and verify the origin of the content.

- FIB: NDN uses FIB to forward the interest packet upstream toward the source(s). The FIB is populated by name prefixes based on the routing protocol. Each entry of FIB contains a list of the outgoing interface(s) for each name prefix as forwarding entries. NDN FIB is similar to IP FIB. However, an NDN FIB entry contains a list of interfaces rather than having only one best next hop as in IP FIB. Forwarding strategies have been used to retrieve the Longest Prefix Matching (LPM) entry from the FIB and decide when and where to forward the interest.

- PIT: It contains all the interest names with their associated interface entries, which have been forwarded upstream but are not yet satisfied. The PIT entries are removed if the matching data packet is found and the corresponding interest can be satisfied, or if the timer expires without any satisfaction.

1.2 Contributions

Our contributions can be highlighted in several aspects:

1. **GeoDTN-NDN: a Geographic Routing Strategy with DTN Support for Vehicular Named Data Networking:** an improved V-NDN routing strategy that enhances geographical routing to efficiently forward interest and data packets.

    - We propose a hybrid geographical-routing scheme. The proposed hybrid geographical routing strategy includes restricted greedy, greedy, perimeter, and DTN modes. We explore the details of how the modes should be transitioned from one to another.
    - We design a method to direct data packets propagation toward the consumer, aiming to facilitate NDN data packets propagation when facing high mobility of vehicles in VANETs.
    - We evaluate the GeoDTN-NDN performance in terms of efficiency and effectiveness.

2. **Progressive Segment Routing (PSR) for Vehicular Named Data Networks:** a routing strategy that utilizes Dijkstra's algorithm and source routing to reduce delivery failure.

    - We propose an innovative routing algorithm that considers the density of vehicles on the road when deciding which path to take. This achieves the goal of avoiding less-traveled roads.

- We allow intermediate nodes to dynamically change the routing path based on locally measured density information so that a more efficient routing path can be found.
- We define a novel criterion for determining the progress of routing to guarantee that the destination will be reached even if a temporary loop may be formed in the path.
- We evaluate the effectiveness and efficiency of the proposed PSR.

3. **Applications of Vehicular Named Data Networking**.

   - We present a classification to categorize different V-NDN applications and introduce the associated NDN naming scheme.
   - We propose a priority model to prioritize delivering the data content based on the NDN data type and name.
   - We develop a smart parking application using a fog computing architecture that supports IoT data collection and avoids the overcrowdedness of the central cloud approach.
   - We conduct simulations to evaluate the performance of the smart parking application. The results show that the proposed solution achieves better performance than existing solutions.

1.3 Thesis Organization

The remainder of this dissertation is organized as follows. In chapter 2, we review the literature related to routing in V-NDN and VANET applications. Chapter 3 discusses the hybrid geographical routing scheme for efficient forwarding of interest and data packets. Chapter 4 presents the progressive segment routing approach for using multiple segments for packet forwarding in vehicular named data networking. Chapter 5 analyzes different NDN applications and presents a naming scheme. It also describes the smart parking application to allow drivers to find parking space and making reservations. Finally, Chapter 6 concludes the thesis and points out future research directions.

Chapter 2

Related Work

This dissertation builds upon a considerable body of previous work. Our discussion on related work consists of two parts. Section 2.1 discusses work related to routing techniques for VANETs. Section 2.2 discusses work related to applications using VANETs.

2.1 Related Work on Routing in VANETs

Early work on routing for VANETs does not use any framework of NDN. Using geographical location information was first introduced in MANETs in general, not specifically for VANETs. Several papers dealt with various problems associated with geographical routing [5, 6]. To deal with the potential partition problems caused by wireless networks in extreme condition, such as deep space exploration and wireless networks in mobile or extreme terrestrial environments, Delay Tolerant Networking (DTN) was proposed. Clustering techniques have been used in NDN to improve the routing efficiency. Finally, we will discuss work on routing in vehicular named data networks.

2.1.1 Geographical Routing

Geographical routing depends on the knowledge of the geographical location of the destination to guide the forwarding of packets. The basic strategy is to use Greedy Routing [5], in which each node always forwards a packet to the neighbor that is the closer (greedy strategy) to the final destination than itself. Collectively, the packet will finally arrive at the destination. Each node maintains only one-hop information for neighbors instead of per-destination routing entries. However, the greedy strategy has its limitation. It is possible that a node may not find a neighbor that is closer to the destination than itself, even though a path to the destination exists. This typically happens when there is an obstacle (such as a building), which affects the distribution of wireless nodes in the space, resulting in a void area in the final topology formed. This is the well-known "local maximum" problem with the greedy routing.

Several solutions have been proposed to solve the local maximum problem. Greedy Perimeter Stateless Routing (GPSR) [7] is the most prominent one among them. GPSR is a geo-location routing protocol that has two modes: greedy and perimeter. GPSR uses the greedy mode until the packet arrives at a node that has no neighbors that is closer to the destination than itself. At that point, it switches to the perimeter mode. If the packet arrived at the forwarding node N from node M, the edge that node N will forward to is the next edge sequentially counter-clockwise about N from edge (M, N). This is usually called right-hand rule. It switches back to the greedy mode when a neighbor node that is closer to D is found. GPSR requires the topology of nodes to be a planar graph. It can be a complicated task to transform a non-planar graph into a planar graph. Several follow-up studies explore how to make sure the topology is a planar graph or how to transform a non-planar graph into a planar graph by removing edges in the graph.

Greedy Perimeter Coordinator Routing (GPCR) [6] adopts an alternative approach to address the planar graph issue. It uses the urban street map as the forwarding graph, instead of the graph generated by the neighboring relations among wireless nodes. The solution considers each intersection as a vertex and the road between two intersections as an edge in the graph. In addition to greedy and perimeter modes, GPCR introduces the restricted greedy mode into the forwarding process. The greedy and perimeter modes are only applied at the intersections where the forwarding decision is made. A node forwards packets along the street to the next intersection uses the restricted greedy mode until it finds a node at an intersection. Vehicles at the intersection are called "coordinator vehicles" and are responsible for making routing decisions. The routing decisions at intersections can be either greedy or perimeter. If a "local maximum" is reached, the perimeter forwarding is chosen at the intersection, and the node has to decide to forward packets to the next street by applying the right-hand rule. Beacon messages are used to determine if the node is at a road or intersection. The main issue with GPCR is that when no vehicle is at the intersection, it can lead to the cross-link issue with possible routing loops.

VANET Cross-Link Corrected Routing (VCLCR) [8] proposed a routing protocol based on the natural planner graph of urban areas where the intersections are considered vertices and the streets between two intersections are considered edges. VCLCR avoids routing to empty intersections by detecting cross-links, resulting from the perimeter mode in the GPCR algorithm. It proposed a simple strategy by recording a path (a list of streets and intersections) a packet traveled during the perimeter mode. When a packet is routed to a road or intersection on the list, the node will know that there is a loop or cross-link and try to avoid that.

2.1.2 Delay-Tolerant Networking (DTN)

In VANET, communication between vehicles can be disconnected when vehicles move apart. This is the well-known intermittent connectivity problem. When a forwarder cannot find any other nodes within its communication range, the network is essentially partitioned. DTN [9] is a paradigm of routing proposed to deal with the disconnectivity issue in wireless networks. To cope with intermittent connectivity, DTN uses store-carry-forward routing. The packet can be buffered at the intermediate node in the routing path if there is no available next-hop forwarder to forward the packet. This duration that the intermediate node has to buffer the packet depends on how long it takes to find an appropriate next-hop node that can continue to forward the packet. When there are multiple next-hop nodes to forward the packet, selecting the next-hop forwarder can determine the fate of the packet. It is a hard decision to determine the best forwarder so that the packet can have the highest chance for successful delivery.

The DTN routing can be categorized into two main categories, deterministic routing, and stochastic routing. Based on these two types, different routing protocols have been designed. In deterministic routing, the routing protocols know the future movement and have the complete network topology. In stochastic routing, the routing protocols are not sure of the mobility pattern of the network. Usually, stochastic routing depends on historical data to predict the future behavior of mobile nodes and makes decisions based on some stochastic model.

Mobile Relay Protocol (MRP) [10] is a relay-based approach used in conjunction with traditional ad hoc protocol. It takes advantage of node mobility to disseminate messages to mobile nodes. MRP protocol combines message routing and storage in the network. The packet follows the normal routine process until it gets to an intermediate node without neighbors. Then that intermediate node stores the packet and enters into relaying mode.

Model-Based Routing (MBR) [11] uses a mobility model of mobile nodes, which contain the location information that can be used to select a relaying node without flooding. It also uses the motion of vehicles as an approach to contribute to the delivery of the message.

MaxProp [12] attempts to provide a routing protocol for DTN messages. MaxProp approach aims to increase the reliability of data delivery in DTN by determining which packets to be transferred to other nodes and which packets to be dropped. It is based on determining the data delivery likelihood to discover the path to a destination.

2.1.3 Clustering in NDN

For NDN, one of the techniques proposed to improve the routing efficiency is clustering. M. Gohar et al. [13] introduced clustering-based routing to solve the mobility issue of producers. They proposed a cluster-based device mobility management (CB-DMM) technique to manage the producer's movement from one access router to another. A Cluster Head (CH) is selected based on storage capacity, energy, and the CH's location. When a producer moves to a new location, it first informs the new access router about its information. The new access router informs the CH about this producer's information. The CH adds/updates the new location of the producer and sends a binding acknowledgment to the new access router. When the CH receives a content request, it knows the new location of the producer and can forward to it.

Huang et al. [14] introduced a cooperative caching strategy in NDN to improve caching efficiency. The authors argue that the caching in the current NDN is non-cooperative. So the content can be cached anywhere, which can cause higher latency and overhead in data delivery. They proposed a cluster-based method to divide vehicles into clusters based on their location and movement. They introduced a cooperative caching mechanism to enable the CHs to cache the content by cooperating with RSUs. Cluster members can benefit from fetching the content from CHs. They also designed a popularity-aware cache replacement method to make popular content more likely to stay in the cache for a longer period of time. This work achieves good performance in reducing the delay and average hop count, and improving the hit ratio.

2.1.4 Routing in Vehicular Named Data Networking

The first work that applies NDN to VANETs is V-NDN [4], which took advantage of the content-centric approach and caching capabilities of intermediate nodes in NDN and made it work coherently with the vehicular networks. It encodes the Geo-location information into data names and uses greedy forwarding to get packets to the destination. It may encounter the "local maximum" problem. Another issue is that it may cause the flooding issue when the Interest packet is broadcast in the network without specifying any destination.

C. Bian et al. [15] proposed a geographical-based design of the NDN forwarding strategy for urban areas in VANETs to handle the issues caused by dynamic topology. The authors employ a timer-based forwarding decision mechanism to promote reliability, multi-path forwarding, and caching ability. The forwarding strategy is based on selecting the furthest node from the sender.

Last Encounter Content Routing [16] proposed an opportunistic routing method, which forwards interests to vehicles that have the content based on the information collected before. They target applications with content not related to a specific location, such as retrieving music or sharing video. To decrease the content exploration overhead, it lets each node maintain the content locations using a data structure called "Last Encounter List (LEL)," which contains the content list and content locations. These last encounter lists are exchanged when two vehicles encounter. When vehicles get LEL information, they will update their lists. A timer-based rebroadcast mechanism is used to reduce the chance of rebroadcasting the same interest, in which vehicles are allowed to rebroadcast interest after a timer expires, and they have not heard other vehicles broadcasting the same interest.

Maryam, et al. [17] studied how to mitigate the flooding storm of interests and address link disconnectivity issues in Vehicular Content-Centric Networking. They proposed an Intersection Based Forwarder Selection (IBFS) scheme, which tries to find the content within one hop, and if not found, the interest will be broadcast to the network. The method consists of two phases. The first phase is the intersection sharing phase, in which performance metrics are exchanged between neighbors within the transmission range. The second phase is the interest/data packet forwarding phase, in which a consumer selects a relay node whose trajectory information is the best. The goal of IBFS is to reduce the delay and minimize the number of forwarders in the forwarding path.

Navigo [18] is a more recent work that considers the urban area situation and exploits Geo-location information for forwarding the interest packets to the destination. It assumes that every node has access to a static map. Navigo considered two types of applications: 1) the applications where the consumer can determine their geographical location, such as searching for road traffic in a specific area and finding the available parking, and 2) the applications where the consumer cannot determine the location of data, such as music sharing. It uses Dijkstra's algorithm to calculate the shortest path to the destination by considering each intersection as a node and the street between two intersections as an edge. Navigo uses the exploration phase (flooding) when there is no information about the data location in the FIB table. So it can suffer from the interest flooding problem in the exploration phase. Another issue is that it assumes that Data packets will be sent to the consumers in the reverse path of the interest packets, which can get interrupted when the nodes on the path move. When calculating the shortest path, Navigo prefers wider roads (with more lanes) over narrower roads (with fewer lanes) because they are likely to have more traffic. Therefore, it adjusted the costs of edges by making them inversely proportional to the number of lanes. In contrast, our approach measures the density of traffic on the roads and adjusted the costs of edges accordingly that can more accurately reflect the situation of road conditions.

2.2 Related Work on Applications of VANETs

There is quite a lot of work related to applications of using VANETs. Some of them have been mentioned in Section 2.1.4. In this section, we will focus on related work in three aspects, NDN-based Internet of Things (IoT), fog cloud computing and the smart parking application.

2.2.1 NDN-based IoT

Integrating NDN with IoT has been investigated by some researchers in the last few years. The following related work focuses on NDN-based IoT applications.

The authors in [19] proposed a vehicular information network architecture that relied on NDN and used IoT technology. Each vehicle is equipped with a sensor(s) to collect data such as weather conditions, location information, etc. The authors focus on designing a hierarchical naming scheme that supports both the pull-based and push-based models for disseminating the sensed data in V2V communication. The hierarchical naming scheme includes the geographical-location of the sensed device so it can be accessed by names. The authors introduced collectors that can be gateways or smartphones. The main function of collectors is to collect data. In addition, they define the metadata that can also be generated by the collectors. The authors include aggregator servers that provide support to collect, process and disseminate the results at different granularity. The authors proposed a method to support data packet aggregation and interest packet segregation by modifying the PIT table. They introduced timers for PIT table entries, and each timer can be set based on the requirement of each application. The authors introduced a good solution to support distributed mobility management and a location-based scheme. However, using several aggregator servers at several levels can be expensive.

Kumar et al. [20] aimed to enhance the Routing Protocol for Low power loss networks (RPL) commonly used in IoT and use it as a routing protocol for mobile nodes such as vehicles. The authors proposed a solution to enhance the mobility feature by allowing mobile nodes to broadcast to neighbors. Each parent mobile node contains a table that stores mobile node information about all of its children. Once the mobile child node has moved, the parent node broadcasts the mobile node information to its neighbors. They also integrated the content-centric feature into the RPL protocol to reduce energy consumption and latency. This solution first implemented the enhanced mobility feature by using IP addresses and used RPL for vehicles.

François et al. [21] introduced the idea of pushing data content in Content-Centric Networking (CCN), in which only one-way interests can carry the small embedded sensed information by using the CCN namespace. The intermediate nodes do not cache or create a PIT entry, and only the FIB forwards the interest packets. The solution was discussed in the context of CCN and did not address the issue of the mobility in VANETs.

Amadeo et al. [22] studied using NDN for IoT in terms of easy data access, energy efficiency, security, and mobility support. However, dealing with IoT devices can be challenging, especially in a wireless environment such as VANET. NDN supports receiver-driven communications, so the IoT pull-based solutions are suitable to deal with the IoT in NDN because the nature of NDN supports it. Amadeo et al. [23], proposed a pull-based solution to collect data in WSN. The authors introduced a smart forwarding scheme to efficiently retrieve data from different sensors to allow the sink node to retrieve data efficiently. They also discussed how NDN can support reliable push-based IoT traffic such as periodic data. For example, sensors for environmental monitoring may want to send measurements at fixed intervals about the observed parameters in the environment. Sensors for health monitoring may want to send event-triggered data, such as the measured blood pressure if it exceeds a safety threshold. These data are suited for using the push-based model.

2.2.2 Fog Cloud Computing

In this Section, we explain related work of using fog cloud computing. Khattak et al. [24] introduced the integration of fog nodes into VANETs. The authors designed an architecture to implement this integration with an infotainment application scenario. They used the cache size of fog nodes as a parameter to evaluate its impact on this integration on the fog-enabled VANET environment.

Both [25] and [26] have studied the ability of cloud computing, especially fog computing, on VANET, to support the computational demands and reduce the response time. The fog computing at edges has the benefits of bringing computing, processing, and storage closer to the sensor devices. Hou et al. [27] introduced the idea of exploiting fog computing in VANET. Xiao and Zhu [28] also introduced a similar idea of vehicular fog computing. They presented cost-effective and on-demand fog computing for vehicular applications. Grover et al. [29] demonstrated that real-time applications of VANET can be implemented using fog computing.

Tang et al. [30] proposed a parking slot allocation strategy that considered the real-time parking slots information by exploiting the fog computing-based capabilities on a smart parking architecture, aiming to enhance smart parking in real-time. The deployed fog nodes (RSUs) at parking lots. These fog nodes can communicate with each other using wired communication, allowing them to process parking requests collectively. Meanwhile, the centralized cloud can promote smart parking capability by enforcing global optimization on parking requests. The authors provided an allocation strategy that considered the comprehensive factors, such as walking costs, driving costs, and waiting costs. The main issue with the approach is that installing, deploying, and maintaining RSUs for every parking area to achieve one-hop communication can be expensive.

Yi et al. [31] proposed a parking reservation auction system that uses cloud computing with the association of parked vehicle cloud. The authors aim to guide mobile vehicles to the available parking spaces with less effort. They focused on monetary rewards to compensate for the service cost of vehicles and allocate the workload to each CPU. The solution results in improved performance due to the fog node contribution.

2.2.3 Smart Parking Applications

Developing smart parking solutions can reduce the parking search time and traffic congestion. It can also alleviate environmental pollution and fuel consumption. Some existing projects such as SFpark (San Francisco) and LA Express Park (Los Angeles) [32] take advantage of smart parking applications to allow drivers to locate available parking spaces using smartphones. Many sensors were deployed to collect the sensed data in those projects and send them to nearby meters. Drivers with smartphone devices can request the availability of parking spaces. However, real-time availability data is only available for vehicles located near the parking location. Also some desired functions are missing from both systems.

Recently, smart parking applications have also attracted many researchers in academia and industries. Several survey papers [32, 33, 34, 35] have been published with a focus on smart parking applications. Lin et al. [32] presented a survey paper about the smart parking ecosystem. They introduced a comprehensive classification of smart parking applications by identifying functionalities and problems of existing work. They divided existing work into three macro-themes based on the target of the work. The first category is the information collection tools that assist in gathering data such as sensors, actuators, and parking meters. The second category is system deployment, which includes the software system that assists with the statistical analysis of the collected data and provides data prediction. The third category is service dissemination that analyzes the relationship between the gathered information and some social feature related to the driver's behaviors.

Hassoune et al. [34, 33] discussed some technologies about parking availability monitoring, parking reservation and dynamic pricing. The authors in [34] investigated different technologies regarding parking availability monitoring. These technologies help the dissemination of the associated information regarding parking space availability and parking reservation. The authors aimed to ensure higher customer satisfaction and increase revenue from parking services. The authors studied the parking infrastructure from different perspectives, including the various parking lots infrastructure.

Faheem et al. [35] presented a review paper showing different Intelligent Parking Services (IPS), which are used for parking guidance and parking facility management. They represent all the techniques that contribute to having an efficient and modern parking system. The authors provided economic analysis for assessing and exploring the project's feasibility.

Parking reservations have also been investigated by several previous studies. Integrating reservation policies with smart parking provides many benefits for both the drivers and managers of parking areas. Delot et al. [36] proposed a reservation protocol that allows vehicles to search for the available parking slots related to their requested event. Vehicles with similar interests in a relevant event will work cooperatively. Besides that, the relevant direction of vehicles is taken into account to support the reservation strategy. They aimed to avoid competition between the vehicles by introducing a probability function that includes a threshold to verify whether an event is relevant for a vehicle or not.

Doulamis et al. [37] introduced an intelligent parking reservation management system that relies on an optimal strategy, aiming to promote service quality of drivers and increase the parking spaces utilization. The authors proposed an approach to utilize the interval scheduling principles. Reservations are represented as a list of parking requests with interested time intervals. They introduce a scheduler that can determine whether to accept the task and assign it to some resource or deny it. They also introduced an adaptive pricing policy such that the price is proportional to the rejection probability of a parking request.

Karbab et al. [38] proposed a scalable and low-cost car parking framework (CPF). The authors introduced driver guidance, automatic payment, parking lot retrieval, security, and vandalism detection. They used smartphone applications to reserve parking slots with hybrid wireless communications. The GPS helps in getting the real-time location and guides the drivers to the destination. All these parking applications are based on using the traditional network, without considering the possibility of getting help from the NDN architecture.

Chapter 3

GeoDTN-NDN: a Geographic Routing Strategy with DTN Support for Vehicular Named Data Networking

3.1 Overview

We consider VANET in an urban environment and assume that vehicles do not have access to a static global map. Each vehicle is equipped with wifi in the ad-hoc mode for communicating with other vehicles. We also assume that each vehicle knows its location, speed, and moving direction. We apply NDN to VANET because each vehicle in need of some content will be considered the consumer of the content, and there may be one or multiple producers of the content. The consumer sends out interest packets specifying the content it needs. The corresponding data packets will be sent back to the consumer by the producer or some intermediate node inside the network.

We observe that VANET applications may be interested in location-associated information, such as the traffic situation along the road to the destination or the parking information near the destination, or any point of interest such as gas stations and restaurants at a specific location or some point along the path to the location. So we adopt an approach similar to Navigo [18] to include geographical location in the interest packets, so that we can use geographical routing to get the interest packet to a producer node in the location. We do not even need to name a specific node but any node in that location. This addresses the flooding problem with the forwarding of interest packets in NDN.

Data packets for satisfying the interest are forwarded in the reverse path in NDN, which is fine in the original wired network environment. However, it can cause problems in the mobile environment of vehicular networks because the nodes along the path may have moved and the sender of the interest packet may also have moved. In this paper, we will include the Geo-location of the sender in the interest packet so that data packets can be forwarded back to the original sender of the interest packet using its location information. If the location of the sender has changed, we can let the sender resend the interest packet toward the source with current (updated) location. We expect that the interest packets will meet the data packet at some node in the middle. This will make sure that we can deliver the packets to the original sender with a higher success rate.

To further address the network partition problem caused by high mobility in VANET, we propose to use Delay-Tolerant Networking (DTN) to enhance the geographical routing for vehicular named data networks. When a node cannot find a neighbor that can make progress for the interest/data packet forwarding, we can let the current node carry the packet for a while until it encounters a neighbor. At that time, we can let the forwarding process continue.

3.2 System Design and Forwarding Strategy

To use geographical routing strategies for forwarding interest and data packets, we include the geographical location of the destination (either a producer of the content or the location of an area of interest) in the interest packet. The intermediate nodes can use this location information to determine how to forward the packet instead of flooding it in all directions. To facilitate the geographical routing of corresponding data packets back to the consumer, we include the location information of the consumer in the interest packets. This location information will be copied to the data packets when they are generated as an answer to the interest packets.

Each vehicle maintains a table of current neighbors. Each node periodically sends out a beacon message. The name of a beacon message has a special prefix "/*hello*" to be distinguished from other messages. Other nodes within the transmission range will record the information in their neighboring tables. Based on the entries in a neighboring table, a node can find the best neighbor as the forwarder. A node can also determine whether it is on the road or at the intersection by using beacon messages.

3.2.1 Selecting a Neighbor

Among all neighbor modes, we can select a neighbor that is the best to forward the packet, based on a score that measures the distance and the velocity. Both the distance and velocity can be positive and negative. If the neighbor is ahead with regard to the moving direction of the current node, the distance is positive, and otherwise, it is negative. If the neighbor is moving in the same direction with that of the current node, the velocity is positive, and otherwise, it is negative.

We use the following formula (3.1) to calculate the neighbor score (NS) and the node with the largest NS will be indicated by the sender for doing the next hop forwarding.

$$NS = \alpha \times V + \beta \times D \tag{3.1}$$

where,

- α, β are system parameters,

- V is the velocity of the neighbor node (only positive and negative as the direction plus the speed),
- D is the distance from the current node.

3.2.2 Splitting Space into Geographical Areas

In traditional IP forwarding, we need to specify the destination IP address, which contains the location information because the IP address is not only used as an identification but implies where it is attached to the Internet as well. Meanwhile, packets in NDN carry data names instead of destination addresses. In geographical routing, we need to steer an interest packet to where the data is located.

In order to specify the geographical location, we adopt the idea of splitting the space into Geographical Areas (Geo-areas) [18]. More specifically, the space is divided into grids in our scheme. Producers that generate the content are located in certain Geo-areas. Consumers are interested in the content related to certain geographical locations. They may not know the ID or the exact location of the producers, but they know the Geo-area(s) they are interested in. Therefore, consumers can specify the Geo-area while sending interest packets. As mentioned, the interest packet is extended to carry more fields, including the location information specified by the Geo-area. The interest packet will be forwarded toward the center position of the destination Geo-area. Then the node near the center will broadcast the packet to all nodes in the same Geo-area. Thus, the interest packet will be finally forwarded to a corresponding producer.

Figure 3.1 shows how to split the space into grids (Geo-areas) and how to steer an interest packet toward a specific Geo-area. A consumer (Yellow) is located in $GeoArea\_1\_1$ and wants to get the traffic information of $GeoArea\_2\_1$. The consumer does not know the position of the producer (green). At first, the destination position field in the interest packet will be the center position (blue) of $GeoArea\_2\_1$. The forwarders (red) will use this information to forward the packet toward the destination. The last forwarder will broadcast to all nodes in the Geo-area and to reach a corresponding producer.

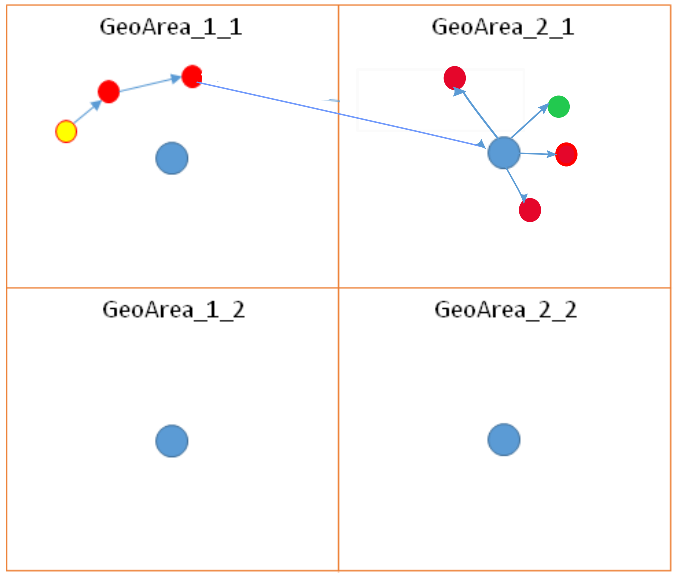

Figure 3.1: Forwarding to Geo-Area

3.2.3 Forwarding Strategies

Geographical routing uses location information to forward the packet toward the destination, typically consisting of two modes – greedy mode and perimeter mode. In the greedy mode, packets are forwarded to a neighbor that is closest to the destination. When a node does not have a neighbor that is closer to the destination than itself, the forwarding will change to the perimeter mode, in which packets are forwarded to the neighbor based on the right-hand rule [7]. The perimeter forwarding is only successful if the graph is a planar graph. In the urban environment, we can consider the streets as the edges and the intersections as the vertices of the graph, which is naturally a planar graph. The only restriction is that when a packet is forwarded on the streets, we can no longer use the general greedy forwarding. Rather we will use the restricted greedy forwarding introduced in GPCR [6].

One of the issues in VANET is that we do not have an always-connected network because of the high mobility of vehicles. It is normal that the network becomes partitioned for a period of time. Therefore, in addition to greedy, perimeter, and restricted greedy, we introduce the DTN mode to the packet forwarding process. When a node cannot find any desired neighbor, it will carry the packet for a while until it can forward the packet to some desirable neighbor. The state transitions among different modes are shown in Figure 3.2.

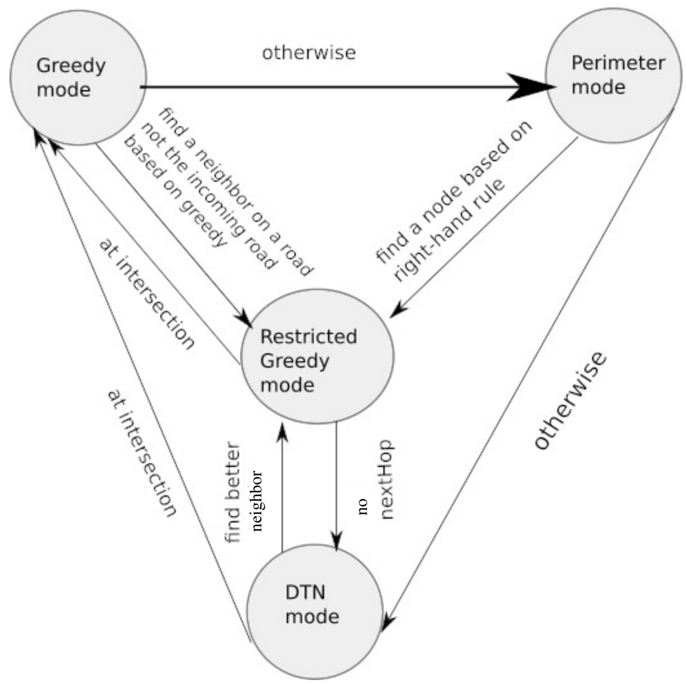

Figure 3.2: Transitions between restricted greedy, greedy, perimeter and DTN modes in the proposed forwarding strategy

Initially, when a vehicle is about to send out an Interest packet, it specifies the location of the destination and also includes its own location in the packet. It will find the best neighbor that is closest to the destination or carry it until it finds such a neighbor. Then it will forward the interest packet to this neighbor.

3.2.3.1 Restricted Greedy Forwarding

When a node on the street (not at intersections) receives a packet, it will use the restricted greedy forwarding strategy. It will forward the packet along the street in the same direction the packet has traveled. This implies that the current node only selects neighbors in the forwarding direction, opposite to the direction the packet comes from. It will select the neighbor that is farthest from the current node so that it can make the biggest progress for forwarding the packet. The only exception is that one of these neighbors is located in the intersection, in which case the neighbor in the intersection is preferred and the forwarding is transitioned to greedy mode. If it cannot find any neighbors in the direction, it will change to DTN mode by carrying the packet along the street.

Figure 3.3 shows such a scenario. A sender vehicle S, located on the road between intersection $J3$ and $J4$, wants to send an interest to $GeoDest\_1\_1$. It selects the best forwarder A from its neighboring table based on the best neighbor score and forwards the interest packet by specifying the desired forwarder within its transmission range along the street toward $J3$. Vehicle A forwards the interest packet to vehicle B and so on until the interest packet gets to the intersection. At that time, a decision has to be made to determine which street the packet should be forwarded to until the interest packet reaches the final destination.

3.2.3.2 Greedy and Perimeter Forwarding

The node at the intersection will make a decision to do greedy forwarding or perimeter forwarding. It will first use a greedy forwarding strategy by finding one among the neighbors closer to the destination than itself and not on the incoming street. Then, the packet will be forwarded along the street where the selected forwarder is located toward the next intersection. The mode is transitioned to restricted greedy forwarding. If it cannot find such a node, the mode will be transitioned to perimeter mode. In this case, it will use the right-hand rule to locate a street, excluding the one where the packet comes from. It will find one of the neighbors on that street that is farthest away from the current node as the forwarder. After that, the forwarding mode will also be transitioned to the restricted greedy. If there is no neighbor on that street, the forwarding mode will transition to DTN

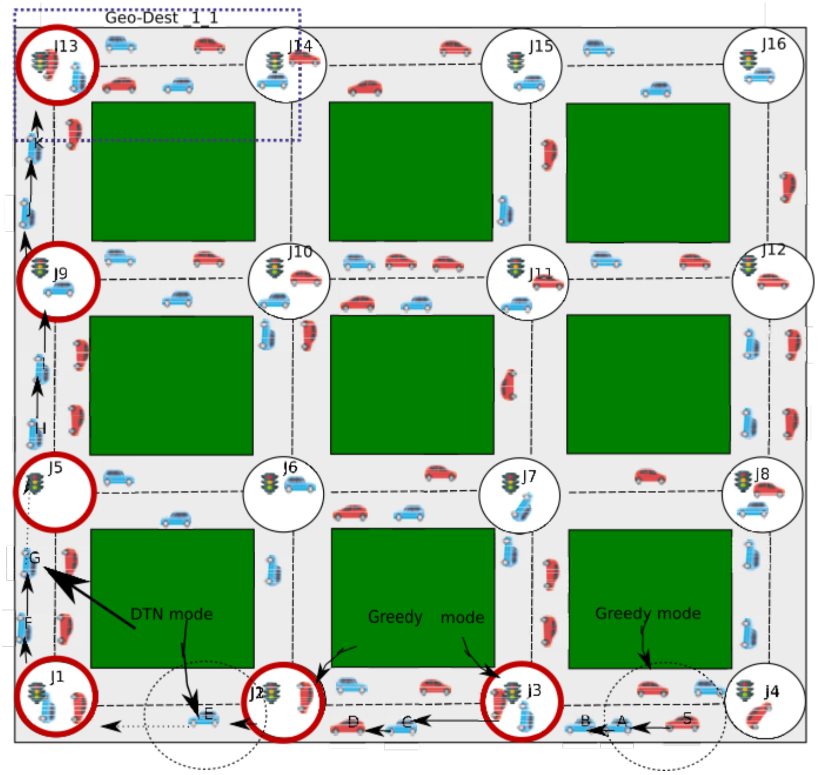

Figure 3.3: A Scenario of Forwarding Strategy

mode. The current node will carry the packet.

In Figure 3.4, sender S sends an interest packet to $GeoDest\_1\_1$. It first forwards to a coordinator node at $J2$, where the forwarding decision is made to decide on what street to follow. The mode, in this case, is greedy. When the interest packet reaches intersection $J8$, it switches its mode to perimeter mode because there is no intersection vertex closer to the destination. Intersection $J8$ uses the right-hand rule and forwards the interest packet to intersection $J7$. By using the right-hand rule, the chosen street is the next one counter-clockwise. Also, at intersection $J7$, the mode is perimeter mode. When the interest packet reaches $J10$, the mode is switched to greedy again. The purple circles around the intersections ($J8$ and $J7$) indicate the intersections that decide to use perimeter mode, while the red circles around the intersections indicate the greedy mode.

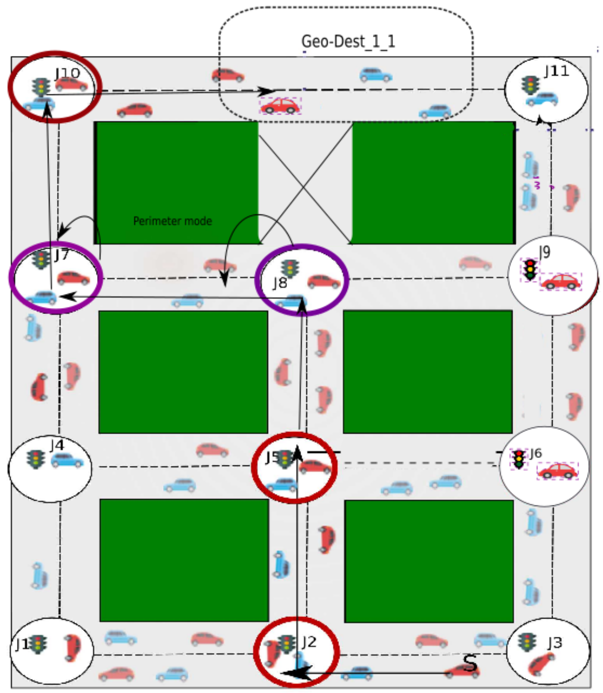

Figure 3.4: A scenario of forwarding strategy (Perimeter mode)

3.2.3.3 DTN Forwarding

Our proposed strategy can decide to switch to DTN mode after checking the availability of forwarders from a neighboring table. If the next forwarder node is missing, meaning that there is no available vehicle to forward the packet from neighbors in the road or the current node is scored better than its neighbors as mentioned in 3.2.1, then it switches to DTN mode.

In DTN mode, the current node will carry the packet for as long as it takes. When the node travels into the next intersection, the forwarding mode will change to greedy forwarding. Before it gets to the next intersection, it will forward the packet to that neighbor if it finds a neighbor along the street in the forwarding direction. If the neighbor is not at the intersection, the mode will transition to restricted greedy; otherwise, it will transition to greedy forwarding.

The DTN mode uses "hold/carry and forward." It will change some fields in the interest packet.

- M: a flag field that indicates the mode of the packet. The flag will be changed to DTN mode.
- DTN_EXP: if the mode is set to DTN mode, an expiration time will be set to indicate how long the selected node buffers and carries this packet. Once the timer expires, the node will drop the packet.

The idea behind our solution is that after entering DTN mode, it takes advantage of vehicle motion to hold/carry the packet toward the destination. If this vehicle finds a suitable next node to forward the packet, the mode will be changed to restricted greedy or greedy/perimeter, depending on whether the next node is on the same road or at the intersection. If this vehicle cannot find such a time within the expiration time, the packet will be simply dropped.

Figure 3.3 shows that when a vehicle E cannot find any neighbor node within its transmission range to forward the packet, it will switch to the DTN mode and carry/hold the interest packet until it finds a next neighbor.

Here is the summary of the algorithm for forwarding interest packets.

1. The vehicle looks up the neighboring table and check whether it is approaching an intersection. If not, it will use the restricted greedy forwarding to choose the neighbor on the same road with the highest score to forward the packet to the next intersection.
2. When the vehicle is approaching the intersection, the routing mode is changed to greedy or perimeter, depending on whether it can find a neighbor on the road that leads to the destination based on greedy forwarding. In the perimeter mode, the vehicle find a next neighbor node on the road based on the right-hand rule.
3. When a node does not find a forwarder node with a better score than itself on a road or intersection, it switches to DTN mode.

4. When a node is in DTN mode, if its DTN_EXP expires, the packet will be dropped. If it finds a neighbor that can forward the packet further, the forwarding will exit DTN mode depending on the location of the neighbor.

3.2.4 Data Packet Forwarding

In the NDN architecture, the path the interest packet traveled will be recorded in the Pending Interests Tables (PIT) in the intermediate nodes. The data packet for satisfying the interest will be sent in the reverse path by looking at the PIT tables. However, it sometimes may be difficult for the data packet to take the reverse path because of the high mobility of vehicles. We have included the location information of the consumer in the interest packet. When a data packet is forwarded back from the producer to the consumer, we can take advantage of this location information to help direct the forwarding of the data packet. Our solution is based on the push-based model that allows the producer to push the data packet toward the consumer.

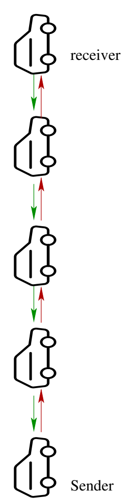

Figure 3.5: Sender, receiver and intermediate nodes remain in their position

We consider three scenarios. First, when the producer (receiver of the interest packet), the consumer (sender of the interest packet), and intermediate nodes remain in the same position. Once the producer receives the interest packet, it can forward the data packet via the reverse path using the NDN forwarding mechanism. Figure 3.5 shows how the sender sends an interest packet to the producer (receiver), and then the data packet is returned in the reverse path to the sender. This scenario is the simplest case that can directly use the existing NDN mechanism.

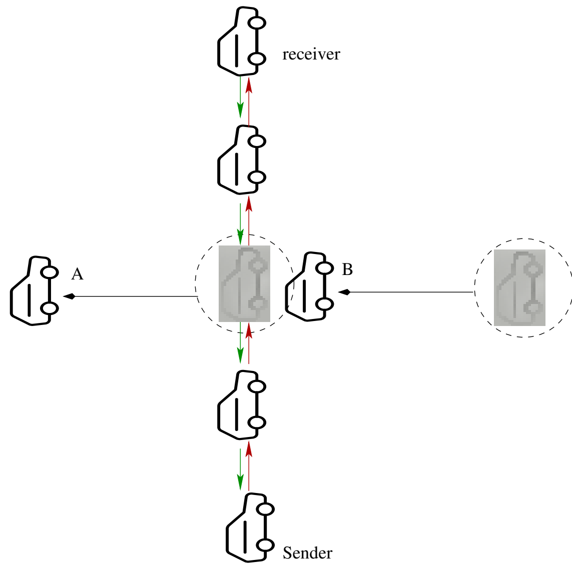

Figure 3.6: A changes its position and B comes in A's previous position

The second scenario is the case in which some intermediate node changes its position, and it is possible that some new node comes to the position to replace the node that has moved. In this situation, we can use the push-based model using the location information of the consumer. The data packet will be pushed toward the geographical location of the consumer using a similar method for interest forwarding. Once the packet reaches a node that has information about the interest in its PIT table, it can use the NDN mechanism to forward the packet. Figure 3.6 shows that an intermediate node A moves to a new position and node B is a new an intermediate node that comes to the position of A. Node B did not participate in forwarding of the interest packet. However, the data packet will be forward to B by the push model, and it will in turn forward the packet toward the sender of the interest packet, even if it did not have a PIT entry for the request.

The third scenario is when the sender of an interest packet moves away from its original position. If the distance is far enough to be out of range of the upstream neighbor and

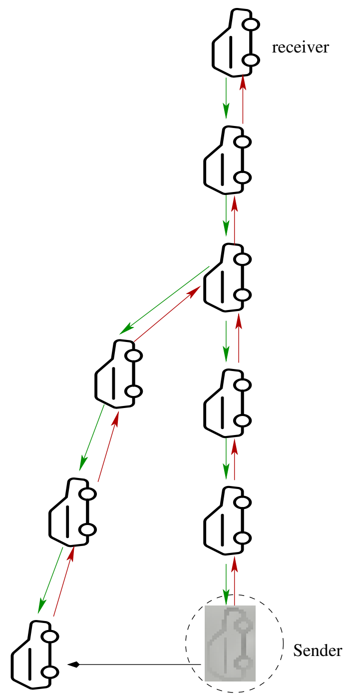

Figure 3.7: Sender changes its position before receiving the Data back

yet the requested data has not been satisfied; it should resend the interest packet toward the producer, including its new position in the packet. This new interest packet may reach some intermediate node from the path taken by transmitting the previous interest packet. Under the NDN model, those nodes will cache the content if the data packet has gone through it toward the original location of the sender. Therefore, the consumer can exploit the NDN feature of caching and receives data from the first node that has the data in its cache instead of traveling all the way to the producer. Figure 3.7 shows that the sender updates its position by sending a new interest packet to the producer and gets the data packet from the first intermediate node met on the original path. This can reduce the time the consumer gets the content from the producer in the mobile environment.

3.3 Simulation

We evaluated our proposed forwarding strategy (GeoDTN-NDN) for vehicular named data networking by simulation and compared its performance with the original implementation of applying NDN to VANETs called V-NDN [4]. We used an ns-3 based open-source NDN simulator ndnSIM [39] (Version 2.6) and the SUMO microscopic traffic simulation for urban mobility [40] as a starting point, and implemented the forwarding strategies for interest packets and data packets. We used a map of the University of Kentucky Campus in the city of Lexington, KY as the area for moving vehicles. There are lanes in two directions for all streets. The vehicle speed limit is 30 km/h. We assume that the transmission range of signals of all vehicles is 150 meters. Figure 3.8 shows the University of Kentucky campus map in SUMO-GUI.

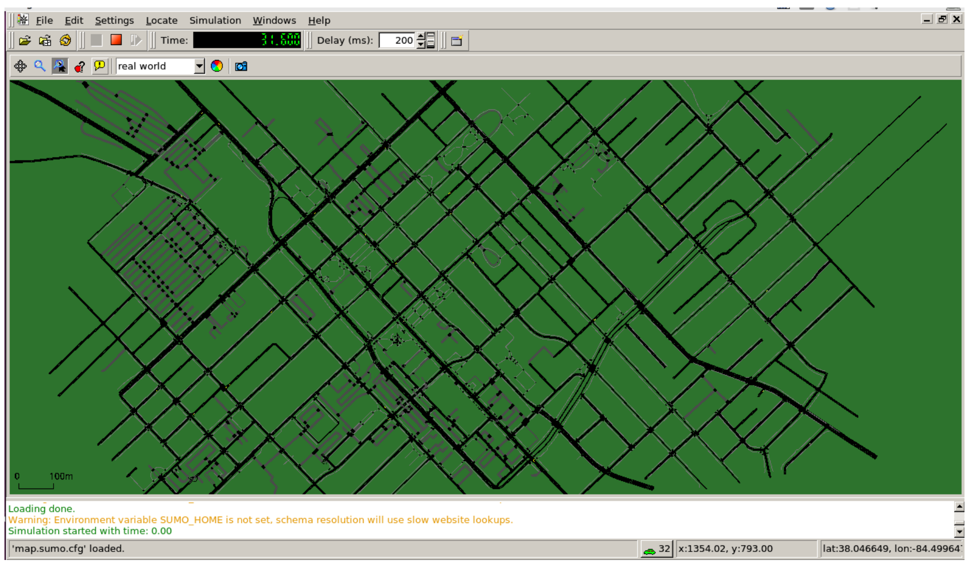

Figure 3.8: University of Kentucky campus map in SUMO

3.3.1 Simulation Scenarios

To evaluate GeoDTN-NDN performance, we considered two scenarios. In scenario 1, we set the number of vehicles to be 50, 100, 150, 200, 300, 400, 500, 600, 700, 800, and 900. We took into consideration parameters such as location, moving speed, and the direction of vehicles. We also considered the number of neighbors within a transmission range. Based on the topology, we set the system parameter for calculating the best score as $\alpha = 0.6, \beta = 0.4$. Table 3.1 shows the parameters of Scenario 1.

Table 3.1: Simulation Step Parameters of Scenario 1

Scenario 1	
Maximum transmission range	150 m
Number of nodes	50, 100, 150, 200, 300, 400,..., 900
Vehicle speed	0-30 km/h
Geo-area size	$200 \times 200m$
Simulation duration	200 ms
MAC type	IEEE802.11

Table 3.2: Simulation Step Parameters Scenario 2

Scenario 2	
Maximum transmission range	150 m
Number of nodes	100
Number of consumers	10, 20, 30, 40, 50, 60, 70, 80, and 90
Vehicle speed	0-30 km/h
Geo-area size	$200 \times 200m$
Simulation duration	200 ms
MAC type	IEEE802.11

In Scenario 2, we considered that not all vehicles send interest packets to get the data. Those vehicles that send interest packets are called consumers, which will get the data from vehicles acting as producers. In Scenario 2, the total number of vehicles is 100, and the number of consumers are set to be 10, 20, 30, 40, 50, 60, 70, 80, and 90. Table 3.2 shows the parameters of Scenario 2.

3.3.2 Simulation Results

We compare the performance of GeoDTN-NDN with V-NDN [4]. The performance metrics used include the packet delivery rate, end to end delay, packet loss rate and throughput. We first present results in Scenario 1.

1) Packet Delivery Rate. It is defined as the ratio of data packets successfully received to the total interest packets sent. Figure 3.9 shows that our solution achieves higher delivery rate than the original V-NDN. The delivery rate of our solution increases from 30% to 65 % when the number of vehicles increases from 50 to 900, while V-NDN remains steady and

then increases up to 40%. When the number of nodes increases, the density of nodes in the simulated area also increases. The higher density means better connectivity among nodes. Furthermore, the distance between nodes will be smaller. This results in higher delivery rate if we have more vehicles in the area. Because GeoDTN-NDN introduces DTN mode and a more robust data packet delivery mechanism, its delivery rate is nearly 22% higher than V-NDN.

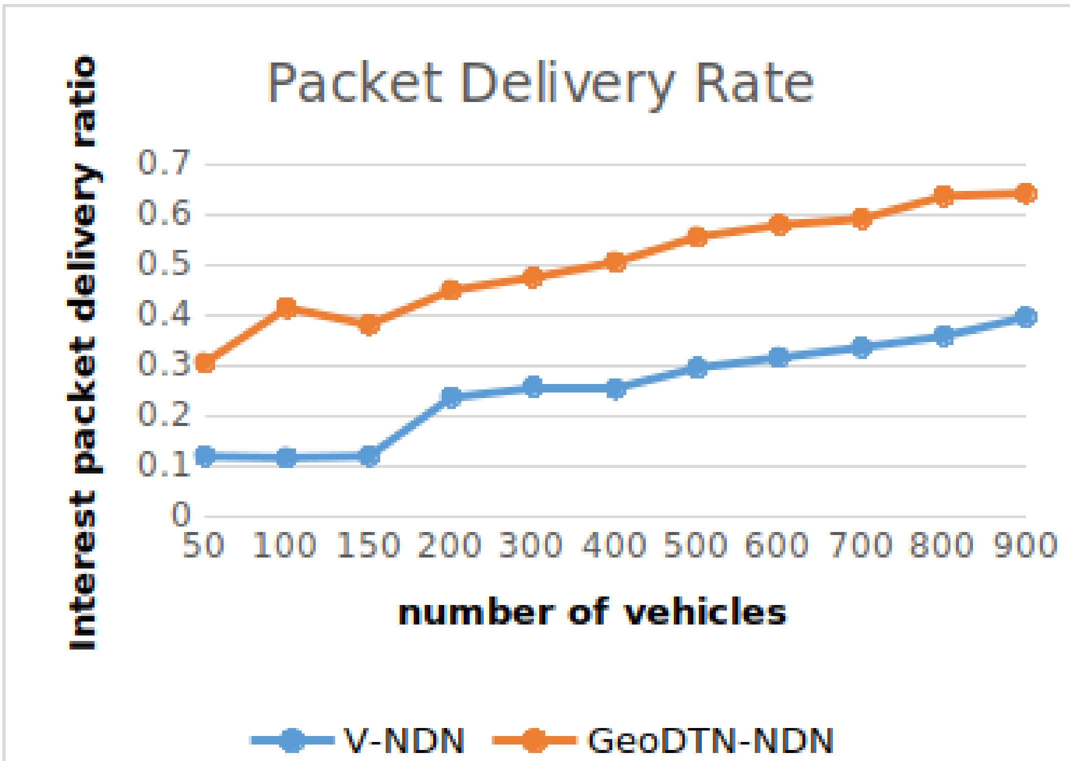

Figure 3.9: PDR for the number of vehicles

2) End to end delay. It is defined as the time from sending the request to receiving the request. Figure 3.10 shows the delay when we have different numbers of vehicles. Overall, the graph shows that when the number of vehicles increases, the delay decreases, partly due to caching in NDN. When there are more vehicles in the network, the chance of finding the content is higher. GeoDTN-NDN has lower delay than V-NDN. However, the improvement is not very significant.

3) Throughput. It is defined as the number of successfully transmitted packets per

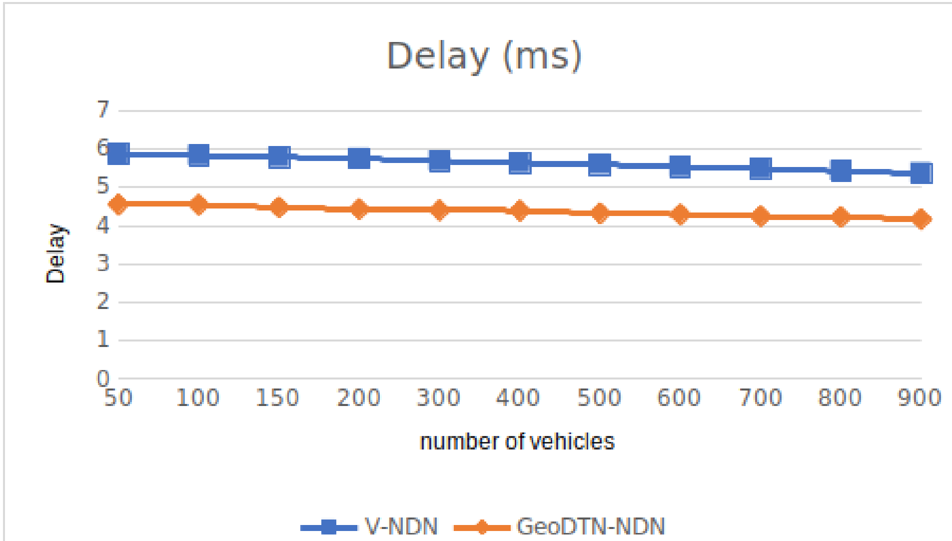

Figure 3.10: Delay for the number of vehicles

unit time. Figure 3.11 shows that the throughput of V-NDN increases from 0 to 8000, while GeoDTN-NDN throughput increases from 1000 to 14000. The throughput in both schemes increases when the number of vehicles increases. GeoDTN-NDN achieves better throughput because the forwarding strategy of GeoDTN-NDN selects the best forwarded among all neighbors.

In Scenario 2, we used a fixed number of vehicles (100) and a variable number of consumers. We used the same performance metrics as in Scenario 1.

1) Packet Delivery Rate (PDR). Figure 3.12 shows that the delivery rate of interest packets. GeoDTN-NDN performs much better, compared with V-NDN. The PDR of V-NDN remains stable at approximately 10%. Meanwhile, the PDR of GeoDTN-NDN is 25% when the number of consumers is 10%. Then it has a jump when the number of consumers is 20%. After that, when the number of consumers increases, there is a slight decrease in the delivery rate, probably because more requests sent to the network cause a higher loss rate.

2) End to end delay. Figure 3.13 shows that the delay for GeoDTN-NDN is less than that of V-NDN. From the graph, we can the delay increases when the number of consumers increases. When the number of consumers is higher, the number of requests sent to the

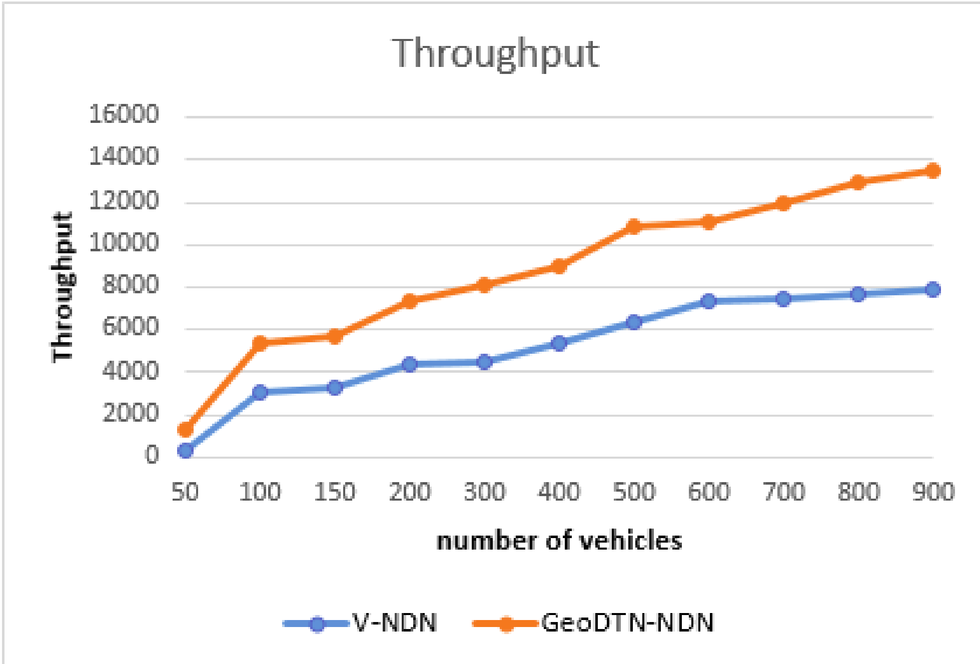

Figure 3.11: Throughput for the number of vehicles

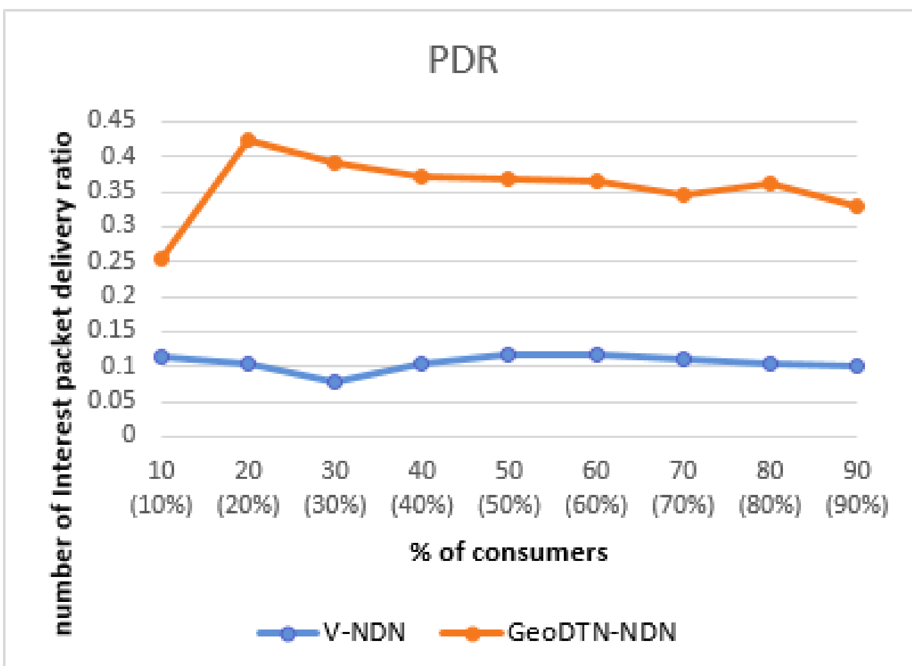

Figure 3.12: PDR for the number of consumers

network increases. Thus the traffic in the network is higher.

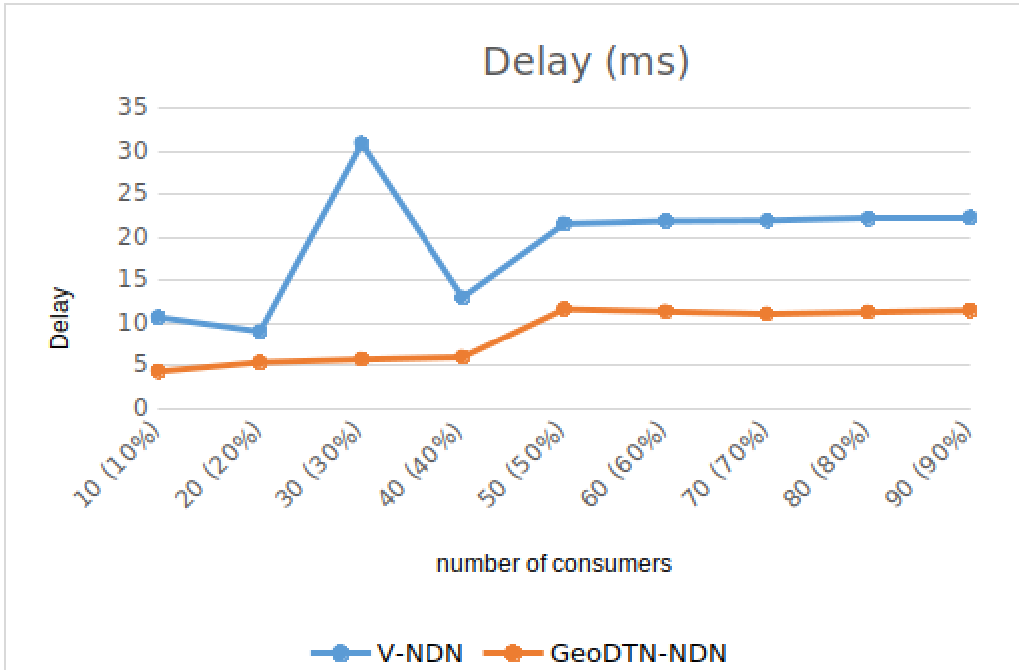

Figure 3.13: Delay for the number of consumers

3) Throughput. Figure 3.14 shows that as the number of consumers increases, the throughput for both schemes increases significantly. This is because the throughput is simply the aggregated rate over all consumers. We can observe that GeoDTN-NDN shows a significant improvement in throughput, especially when the number of consumers are from 50 to 90.

In summary, our proposed solution GeoDTN-NDN can improve performance with regard to packet delivery rate, end to end delay and throughput, in both evaluated scenarios.

3.4 Summary

In this Chapter, we discussed the main challenges when applying NDN to VANETs. We proposed a hybrid geographic routing solution (GeoDTN-NDN) combining the restricted greedy, greedy, perimeter, and DTN modes to improve packet delivery forwarding in urban area. Our proposed solution incorporates the strengths of both Geographical routing

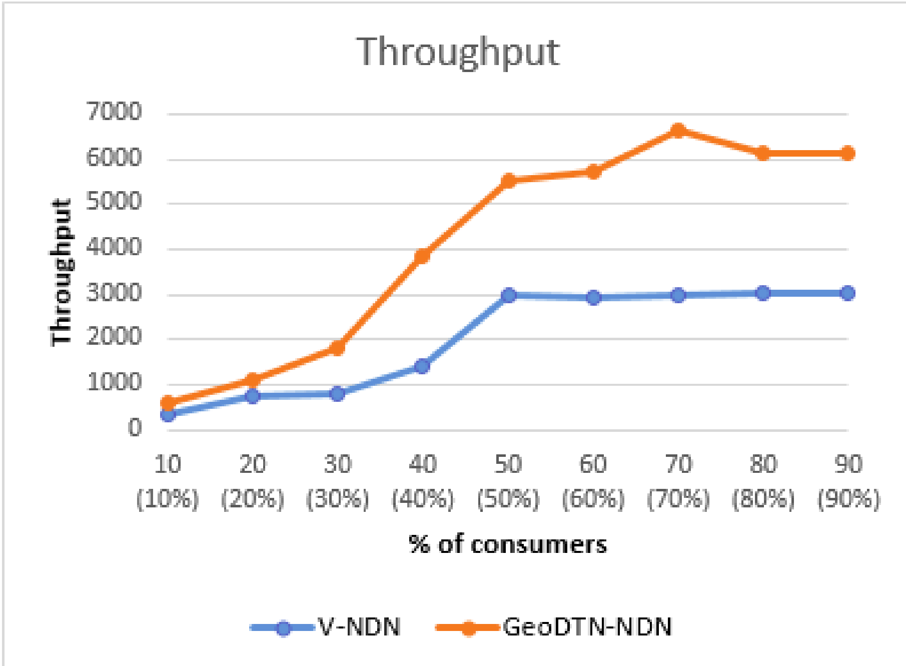

Figure 3.14: Throughput for the number of consumers

and DTN forwarding. We introduced a mechanism for data packet propagation and considered the scenarios where the sender changes its location. Our simulation results showed significant improvement in packet delivery rate and throughput of the proposed forwarding strategy.

Chapter 4

Progressive Segment Routing for Vehicular Named Data Networks

4.1 Design Overview

In this chapter, we shift our focus on an urban environment where vehicles use the vehicle to vehicle (V2V) communications and each vehicle has a map to get the road information, including the attributes of roads and intersection locations.

We have the same targeted category of VANET applications (transportation) as in the previous chapter. An observation is that the producers of the content for these applications are typically located in a certain geographical area. Using the location information of potential sources can direct the interest packets in a certain direction. Therefore the flooding caused by a broadcast of interest packets can be avoided. In particular, with the assumption that the vehicles in these applications are equipped with a map of an area of interest (e.g., in urban areas), a simple approach is to use Dijkstra's algorithm to calculate the shortest path from the consumer to the producer of the content as the routing path for forwarding interest packets.

We have to address several issues with regard to the routing of interest packets. First, we need to determine the metric of cost of edges used in Dijkstra's algorithm. The physical distance may be a good measure if we are figuring out a path for a vehicle to travel to the destination. However, we are interested in forwarding the interest packets to the destination. It is the nature of VANETS that communication may experience frequent disconnectivity due to the fast mobility of vehicles. When selecting a path, we may want to have a path with the least chance of being disconnected.

Second, the path condition may change over time. Letting the source node calculate the whole path to the producers of the content may lead to a sub-quality path. We need to make the decision on how far the source should dictate the forwarding of the interest packets.

Third, if multiple points along the path make decisions, we may end up with a loop in the forwarding process. We have to make sure the algorithm can avoid such situations.

To address these problems, we propose a progressive segment routing (PSR) approach for handling interest packets in vehicular NDNs. The new routing algorithm takes into consideration how vehicles are distributed among different roads and choose well-traveled roads over less-traveled roads. The goal is to reduce the failure rate of packet delivery. The whole routing path consists of multiple segments. The source will calculate the routing path to some intermediate target intersections as the first segment, and the rest of the path will use the shortest path as the default. After reaching the intermediate target, the default shortest path can be re-calculated by relay nodes based on changed conditions. At this time, another segment can be formulated, followed by another default shortest path to the destination. The process continues until the producer of the interest is reached.

One problem with the multi-segment routing is that it may generate a loop because the intermediate targets calculate the path independently from the previous path. To avoid an infinite loop in the path, we propose the idea of progressive segment routing. When a node generates a source routing path, not all nodes can re-calculate and update the path in this source routing path. Instead, only those nodes with a shorter distance on the map graph than the current node can update the path and send the interest packet toward the producer, including its new position in the packet.

4.2 Progressive Segment Routing Strategy

Most routing problems can be reduced to finding the shortest path in a graph. If we can build a graph among all vehicles, called *vehicle graph*, we can use the same algorithm to find the routing path, except that this graph is constantly changing. Notice that this vehicle graph is different from the *map graph*, in which vertices are intersections and edges are the roads connecting these intersections.

The basic idea of our approach is to reduce the routing problem to the problem of finding the shortest path in the map graph. The starting step is to use source routing. When the consumer needs to send out an interest packet, it computes the shortest path using the map graph. The remaining problem is how to forward a packet from one intersection to another. When a vehicle on one road approaching the intersection needs to relay packets to the next vehicle, it should choose one on the correct road, under the assumption that the GPS information is accurate enough and neighboring nodes have exchanged location information. Once the message is received by a vehicle on the new road, we assume that the restricted greedy forwarding mechanism [6] will be used. That is, nodes will always forward the packet along the road in the same direction the packet has traveled. This restricted greedy forwarding will continue until the node relaying the message is approaching the intersection.

As we mentioned before, the physical distance may be a good measure for finding a path for a vehicle to travel to the destination, and may not necessarily be the best for finding a path for packet forwarding. We need to take into consideration the frequent disconnectivity due to the fast mobility of vehicles. We will use the approach to avoid less traveled roads to reduce the chance of getting into the situation in which a vehicle cannot find the next relay node. To that end, we will estimate the density of vehicles on each road and adjust the distance by the density as the cost of edges used in the algorithm.

Our proposed routing mechanism consists of two major steps, density estimation of roads and progressive routing.

4.2.1 Collection of Density Information

To achieve the goal of routing interest messages along the roads that have constant traffic so that a node will likely find the following relay nodes, we developed a mechanism to estimate how frequently each road is traveled. We used density to measure how busy a road is, defined as the number of vehicles traveling on the road per distance unit. We normalized it to the range of 0 to 1. We will give a definition next and explain how it will be used to adjust the distance information used for calculating shortest paths. We do not expect to have density information of all the roads on a map for scalability reasons. The total number of roads may be too large to keep the status up to date. The states maintained at each node should also be minimal. Therefore, we will let each vehicle keep track of the number of vehicles encountered for the last n roads it traveled, where n is a design parameter and can be adjusted. For example, we can let a vehicle keep track of $n = 8$ roads most recently traveled. When a vehicle encounters another vehicle on a road, the exchange of beacon messages between these vehicles will allow each identifies that another vehicle is on the same road. A vehicle maintains a list of IDs of other vehicles on the current road. Once it moves on to the next road, it only needs to remember the number of vehicles encountered. When a vehicle needs to record the number of vehicles on a new road and has already had n road information stored, the oldest road information will be discarded. The stable state is that each vehicle has the information of the number of vehicles on last n roads it traveled. Note that the number recorded by a vehicle on a road is an undercount of the actual number of vehicles on the road because it most likely cannot meet every vehicle on the road. Nonetheless, it is the best estimate we can get that actually reflects how busy each road is.

For each road, the density is determined by the number of neighboring vehicles N_i, the length of the road L_i, and transmission range of wireless signal R. We can define the density of road i as follows.

$$Den_i = \min(\frac{N_i}{\beta * \frac{L_i}{R}}, 1) \tag{4.1}$$

where β is a parameter that defines the target of saturation level and $\beta \geq 1$.

Under the assumption of transmission range being R, for a road of length L, we need at least $\frac{L}{R}$ vehicles to cover the whole road length, if the vehicles are evenly distributed on the road with equal distance among themselves. However, that will not happen in reality. The redundancy parameter β states the target number of vehicles (i.e., β times $\frac{L}{R}$) to accomplish smooth communication on the road. In practice, β can be somewhere in the range of 5 to 20. The ratio of the actual number (N_i) of vehicles on the road over this target number ($\beta * \frac{L_i}{R}$) is defined as the density of the road. To limit the density number to the range from 0 to 1, we put an upper limit of 1 to get the density value.

When a consumer node or a relay node needs to calculate the shortest path to the destination, it asks its neighbors about the number of the vehicles recorded on their last n roads traveled. The current node calculates the average number of vehicles for each road it has information. Using formula (4.1), it can calculate the density of roads for which it has information. If the node does not have information about the number of vehicles recorded, it just considers the density to be 0.

After getting the density information, the node can calculate the adjusted distance for each road as follows:

$$Dis_i = L_i * (1 - \alpha * Den_i) \tag{4.2}$$

where α ($0 \leq \alpha \leq 1$) is a parameter that defines how aggressively we want to adjust distance values.

We can look more closely at how the values of α will adjust the distance values; if $\alpha = 0$, then $Dis_i = L_i$. The density information does not affect at all the length value. In the unlikely case when $\alpha = 1$, we will have $Dis_i = 0$ if the density is $Den_i = 1$; that is, we will consider the length of the road is 0 if density is 1. The larger the values of α, the higher the density's effect on the road length. More likely, we expect that α get a value close to 1, such as 0.8 or 0.9. This reflects our preference for higher-density roads over lower-density roads.

4.2.2 Progressive Routing Algorithm

When a consumer (source) node needs to send an interest packet, it calculates the entire path using Dijkstra's algorithm on the underlying map with the length of road adjusted by the collected density information. Using source routing, it attaches the interest packet with the path, defined as a sequence of intersections that the interest packet should travel. The relay node selectively chooses a forwarder when approaching intersections to ensure that the packet is forwarded to the next node on the right road. Assisted with the restricted greedy forwarding [7] on the road, the interest packet can be forwarded to the destination locations.

However, the source node only has a limited view of the whole network topology because its neighbors only store densities of up to n roads. It may find the better path close-by and choose a sub-optimal path far away. Also, the density information may change over time. Therefore, the source routing path may not necessarily be optimal for the interest packets to travel. To provide more flexibility, we allow the relay nodes on the source routing path to make new decisions to update the source routing path based on their density information collected at the location closest to where the packet will travel with more up-to-date information [41]. The path traveled so far from the consumer node to

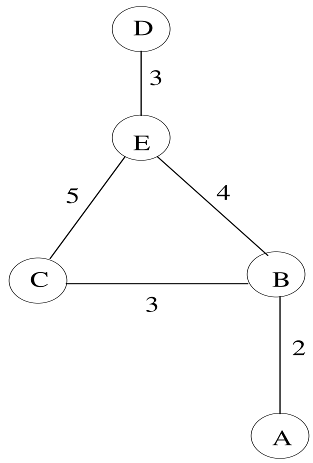

Figure 4.1: Updating source routing path

the relay node is called a *segment*. The rest of the path will be another segment, which can potentially be divided into more segments if other relay nodes downstream update the routing path.

For example, in Figure 4.1, source node A calculates the path to destination D is $[A, B, E, D]$. However, when the packet arrives at B, B finds that a better path $[B, C, E, D]$ based on its density information. It will replace the original path with $[B, C, E, D]$. Finally, the packet will go through $[A, B, C, E, D]$ to get to the destination D. We have two segments $[A, B]$ and $[B, C, E, D]$ in this routing path.

One possible problem with the dynamic updating of the source routing path is that we may end up with a loop in the routing path. For example, in Figure 4.1, after B updates the routing path $[A, B, E, D]$ with its newly found path $[B, C, E, D]$, it forwards the packet to C. However, C may find the path $[C, B, E, D]$ is the best path to destination D based on the information it collected; thus, it will forward the packet back to B. This process can continue forever.

To deal with this problem, we propose a progressive approach for segment routing. While we still want relay nodes to be able to update the routing path, we will impose a condition on who can update to make progress toward the destination at each step.

After calculating the source routing path from the current node to the destination with density-adjusted lengths, the progressive segment routing algorithm uses the unadjusted graph with the original lengths of roads to calculate the distance from each node in the path to the destination. In the source routing path, these nodes having a longer distance than the current node to the destination (both distances using the unadjusted lengths) will be marked. When approaching these marked intermediate nodes in the path, the relay node will not update the source routing path. Only when approaching unmarked intermediate nodes, the relay node may possibly update the source routing path.

We go back to the example in Figure 4.1. At node A, all nodes in the source routing path, including B, E, and D have a shorter distance than A. So, no nodes are marked. When approaching intersection B, the relay node will update the path because it finds a better path to D, i.e., $[B, C, E, D]$. However, because C has a longer distance to D than that from B to D in the unadjusted graph, C is marked. So, the updated source routing path is $[C*, E, D]$. When the relay node approaches C, because C is marked, it will not update the routing path. Rather, it will just forward the packet to a node on the road from C to E. When approaching E, a relay node will not be able to find a better path to D. Finally, the packet will be delivered to D. By imposing these restrictions, we can avoid the routing loop problem.

We present the progressive segment routing algorithm that a relay node will execute upon receiving a packet to forward it to the next node in figure 19. Step 2 determines whether to update the source routing path; only unmarked intersections will trigger the update. Steps 9 to 15 decide which nodes in the updated path get marked based on the distance to the destination compared to the distance from the current intersection to the destination.

We give another example in Figure 4.3 to show that using the progressive segment routing can guarantee that the packet will be forwarded to the final destination, even if a node may be visited twice (a loop). When the original source A forwards the packet to D, the source routing path is $[A, B, C, F, D]$. Since no nodes in the path have a shorter

Data: current node n_c, destination D, source routing path $P = [L_1, L_2, \cdots, L_k]$

if *the current node approaches an intersection* L_1 **then**
    if *the intersection* L_1 *is marked in the source routing path* **then**
        remove L_1 from the routing path
        forward the packet to a node on the next road (L_1, L_2)
    else
        collect density information from neighboring nodes
        calculate the adjusted length for each road edge based on densities
        calculate the shortest path P' from L_1 to D using the adjusted lengths
        if P' *is different from* P **then**
            use the unadjusted map graph to calculate the distance d_c from L_1 to D
            for *each node* n_i *in* P' *except* L_1 **do**
                use the unadjusted map graph to calculate the distance d_i from n_i to D
                if $d_i > d_c$ **then**
                    Mark n_i
            replace P by P'
        remove L_1 from P
        forward the packet to a node on the road from L_1 to the first node in P
else
    forward the packet to the next vehicle using restricted greedy forwarding

Figure 4.2: Progressive Segment Routing Algorithm

distance than A, they are not marked. Approaching intersection C, the updated routing path based on density can be $P' = [C, E, G, D]$. The distance of the current node C to D is 3+6=9 using the map graph. We calculate the distances of nodes in P'. The distance from E to D is 7, which is smaller than C's distance to D. So, E is not marked. Similarly, G and D are not marked. So, the updated routing path is $P = [E, G, D]$.

When the packet approaches E, the updated routing path based on density can be $P' = [E, B, C, F, D]$. The distance of the current node E to D is 3+4=7 using the map graph. We calculate the distances of nodes in P'. The distance from B to D is 12, which is greater than E's distance to D. So, B is marked. Similarly, C is marked; however, F is not marked. So, the updated routing path will be $P = [B*, C*, F, D]$. Based on the algorithm, when the packet approaches B, no update of the routing path will happen since B is marked. Similarly, when the packet approaches C, no update of the routing path will

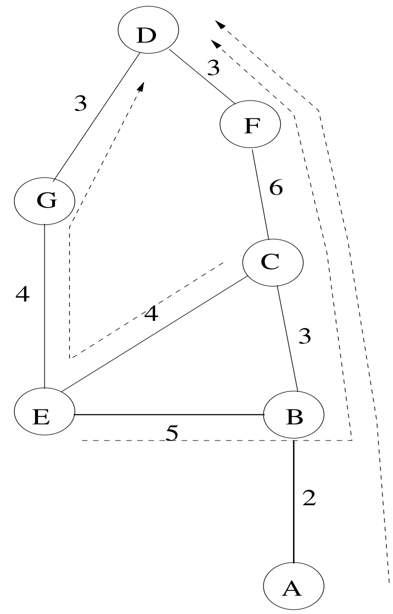

Figure 4.3: An example of progressive segment routing

happen since C is marked. It is fine that the path may be updated when approaching F, but that will not cause a problem because we guarantee that progress is made when we do an update each time. The routing path in this example consists of three segments, $[A, B, C]$, $[C, E]$ and $[E, B, C, F, D]$.

We will make one more observation. Just from the temporary loop ($[C, E, B, C]$) in which the packet travels, it seems that the routing path is not optimal. Actually, based on the states collected by the decision making relay nodes, the decisions are the best they can make at the time. The changes of the network states or the limitation of information they can get lead to inconsistent views of the network. However, the algorithm will still make sure that the packet can be delivered to the final destination.

4.3 Simulation

4.3.1 Simulation Environment

We use an ns-3 based open-source NDN simulator ndnSIM (Version 2.6) [39] and the SUMO microscopic traffic simulation for urban mobility [40] as the starting point. We

Table 4.1: Simulation Step Parameters of Scenario 1

Scenario 1	
Maximum transmission range	150 m
Maximum transmission range of RSU	400 m
Number of nodes	50, 100, 150, 200, 300, 400,..., 900
Vehicle speed	30-40 km/h
Simulation duration	200 ms
MAC type	IEEE802.11

use OpenStreetMap [42] to get a map of our University Campus that spans a 1500×1000 meter area. Vehicles move in both directions on all streets. The vehicle speed limit is 30-40 km/h. We assume that the transmission range of signals for all vehicles is 150 meters. We implemented our proposed algorithm using ndnSIM to meet our requirements. To evaluate the performance, we compare the results with two previous schemes, V-NDN and GeoDTN-NDN. The evaluation uses several metrics including packet delivery ratio, throughput, and delay.

4.3.2 Simulation Scenarios

We generated two scenarios to compare our system with V-NDN and GeoDTN-NDN. In Scenario 1, we set the number of vehicles to be 50, 100, 150, 200, 300, 400, 500, 600, 700, 800, and 900. Table 1 shows the simulation setup parameters of Scenario 1. In Scenario 1, we investigated the performance of original V-NDN, GeoDTN-NDN, and PSR when using different numbers of vehicles.

In Scenario 2, we evaluate PSR by comparing it with the original V-NDN and GeoDTN-NDN. The number of vehicles is fixed (100), and the number of consumers is set to 10, 20, 30, 40, 50, 60, 70, 80, and 90. Table 4.2 shows the simulation setup parameters of Scenario 2.

Table 4.2: Simulation Step Parameters Scenario 2

Scenario 2	
Maximum transmission range	150 m
Maximum transmission range of RSU	400 m
Number of nodes	100
Maximum transmission range of RSU	400 m
Number of consumers	10, 20, 30, 40, 50, 60, 70, 80, and 90
Vehicle speed	30-40 km/h
Simulation duration	200 ms
MAC type	IEEE802.11

4.3.3 Simulation Results

In this section, we present the simulation results of both Scenario 1 and Scenario 2. We use several performance metrics in the evaluation, including packet delivery ratio, delay and throughput.

We first present the results in Scenario 1.

- Packet Delivery Ratio (PDR). It is defined as the ratio of the number of packets successfully received to the total number of packets transmitted. The graph in figure 4.4 illustrates the PDRs of V-NDN, GeoDTN-NDN, and PSR. From the figure, we find that PSR achieves the highest packet delivery ratio because it avoids less-travelled roads. When the number of vehicles is 50, the PDR is the lowest for all three schemes because the vehicle density is low and the connectivity is the worst. All are gradually get better performance when the number of vehicle increases. We observed that the PDR almost grows linearly proportional to the number of vehicles due to vehicles availability and the nature of NDN. It shows that the PDR dipped for both V-NDN and GeoDTN-NDN when the number of vehicles is 150 because they might face some disconnectivity at this number. Overall, V-NDN and GeoDTN-NDN to have a lower PDR compared with PSR. Specifically, the PDR for GeoDTN-NDN is about 20% higher than that of V-NDN, due to the hybrid forwarding algorithm with the prorogation algorithm in GeoDTN-NDN. The PDR of our proposed PSR is about $5\% \sim 25\%$ higher than that of GeoDTN-NDN.

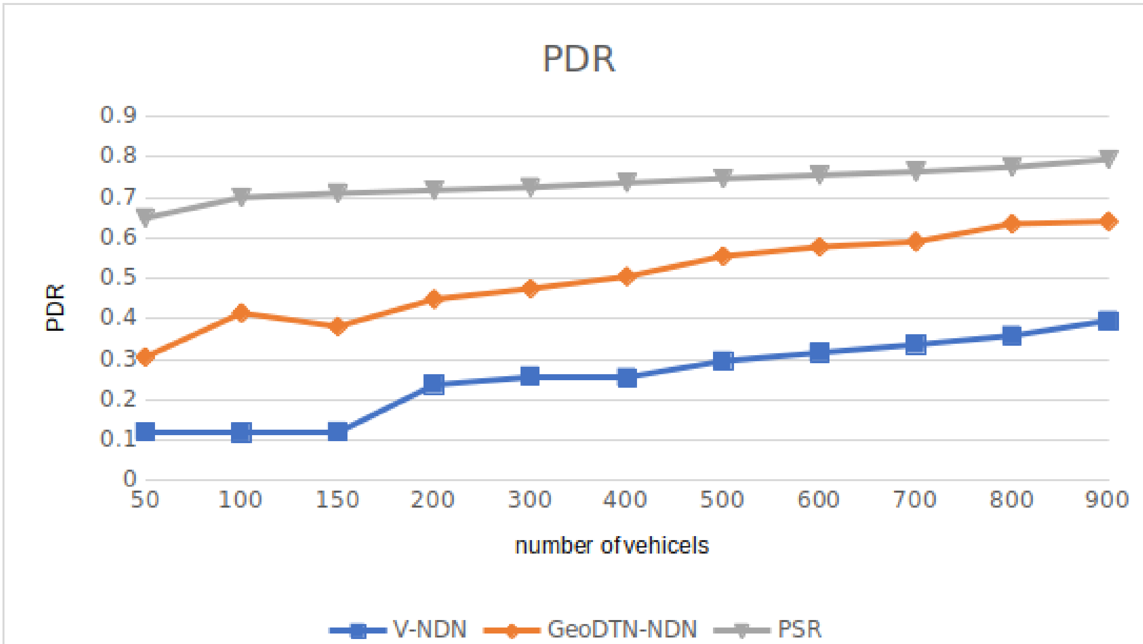

Figure 4.4: PDR for the number of vehicles

- Delay. It is defined as the end-to-end delay of all successful packets sent. Figure 4.5 shows the average delay when the number of vehicles changes from 50 to 900. The results indicate that whenever the number of vehicles increases, the average delay decreases due to the better connectivity in NDN. Figure 4.5 illustrates that PSR attains better performance regarding the average delay than previous solutions due to the proposed routing strategy.

- Throughput. It is defined as the number of successfully forwarded packets per unit time. As expected, all three schemes get better performance when the number of vehicles increases. Figure 4.6 shows that V-NDN and GeoDTN-NDN achieve better throughput than PSR when the the number of vehicles is below 500. We observed some overhead, which is caused by control messages in PSR. When the number of vehicles is 500 or greater, PSR achieves larger throughput than V-NDN and GeoDTN-NDN. The throughput of PSR is 56.62% higher than that of V-NDN and 26% higher than

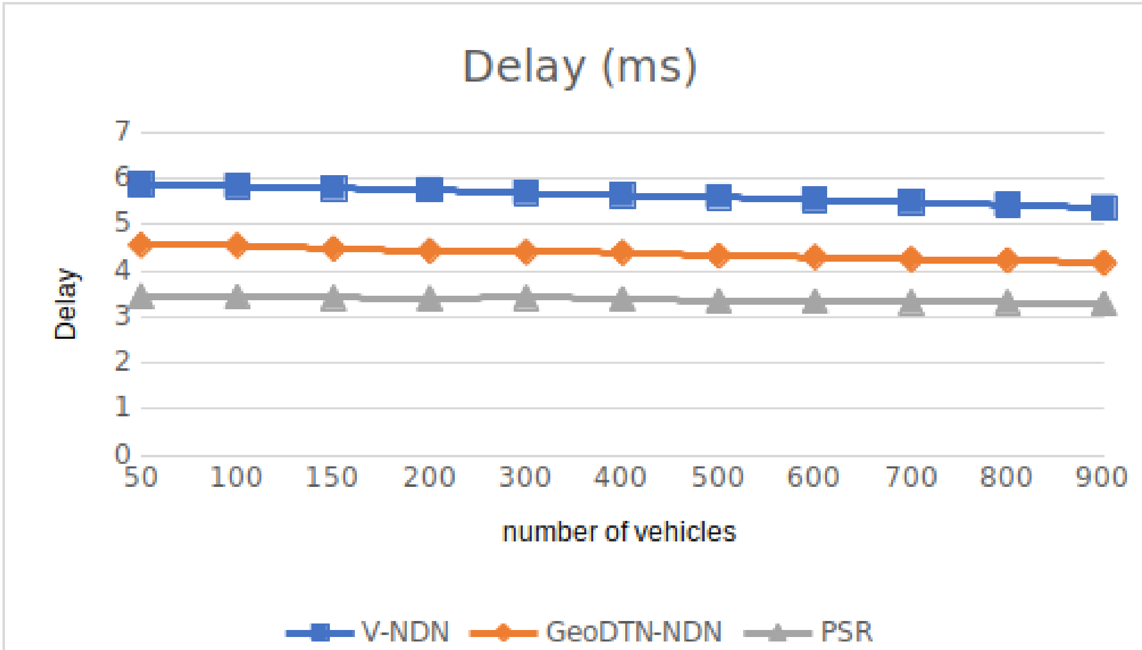

Figure 4.5: Delay for the number of vehicles

that of GeoDTN-NDN when the number of vehicles is 900. This suggests that PSR is more efficient because the forwarding strategy utilizes the naming and PIT features in NDN, and especially it considers forwarding packets using roads with higher density to avoid network disconnectivity.

In scenario 2, we used a fixed number of vehicles (100) and changed the number of consumers. We presented the results for scenario 2, using the same performance metrics as in scenario 1.

- Packet Delivery Ratio: Figure 4.7 shows the packet delivery ratios of three schemes PSR, V-NDN, and GeoDTN-NDN. The PDR of V-NDN remains steady at approximately 10%. Meanwhile, GeoDTN-NDN achieves around 25% when the number of consumers is 10%. The ratio grows to 42% when the number of consumers is 20%. PSR achieves a similar shape to the GeoDTN-NDN's curve but with substantial improvement. It starts at 47% when the number of requests is 10% and it grows to 52% when the number of customers is 20%. After that, both curves remain constant with a bit of decline. When the number of customers is 90%, it means a lot packets

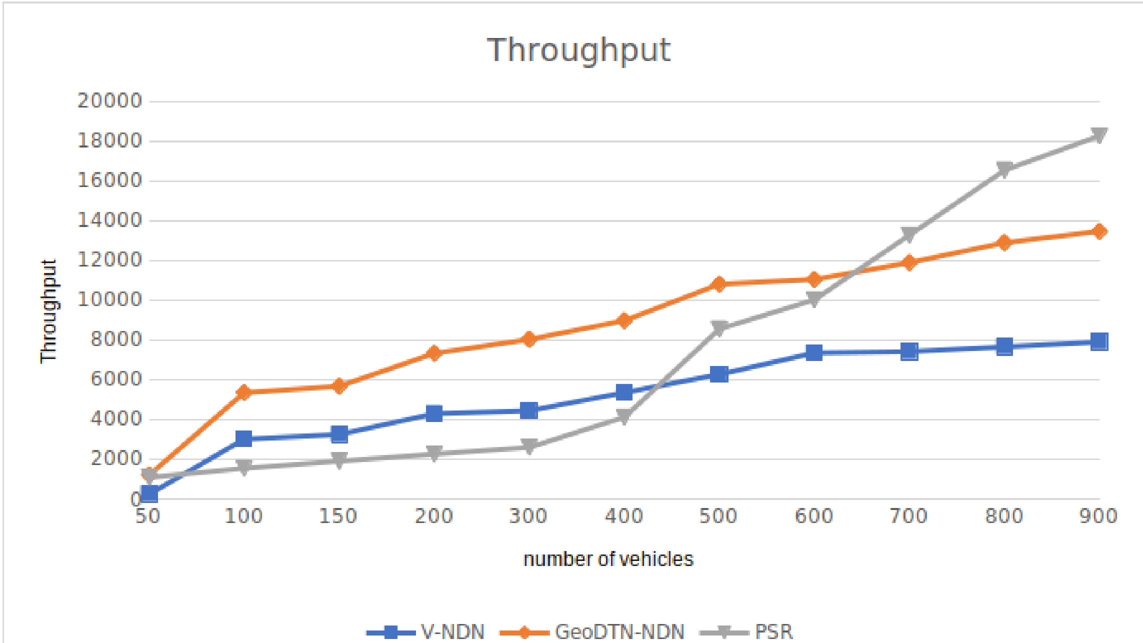

Figure 4.6: Throughput for the number of vehicles

are sent through channels. When the number of consumers increases, it means that there are more requests sent to the network and this can cause the delivery ratio to decrease. However, PSR outperforms other two schemes with regard to the metric of packet delivery ratio.

- Delay. The delay is presented as a function of the number of consumers. Figure 4.8 shows that PSR achieves smaller delay than V-NDN and GeoDTN-NDN no what how many consumers we have. The delay increases when the number of requested packets increases. GeoDTN-NDN has a lower delay than V-NDN but higher than our proposed PSR.

- Throughput. Figure 4.9 shows that as the number of consumers grows, the throughput for all three solutions increases, especially when the number of requests is 30%, the throughput grows significantly. This is because the caching ability in NDN leads to higher throughput. We can observe that PSR shows significant improvement, especially when the number of consumers changes from 40% to 90%. PSR achieves

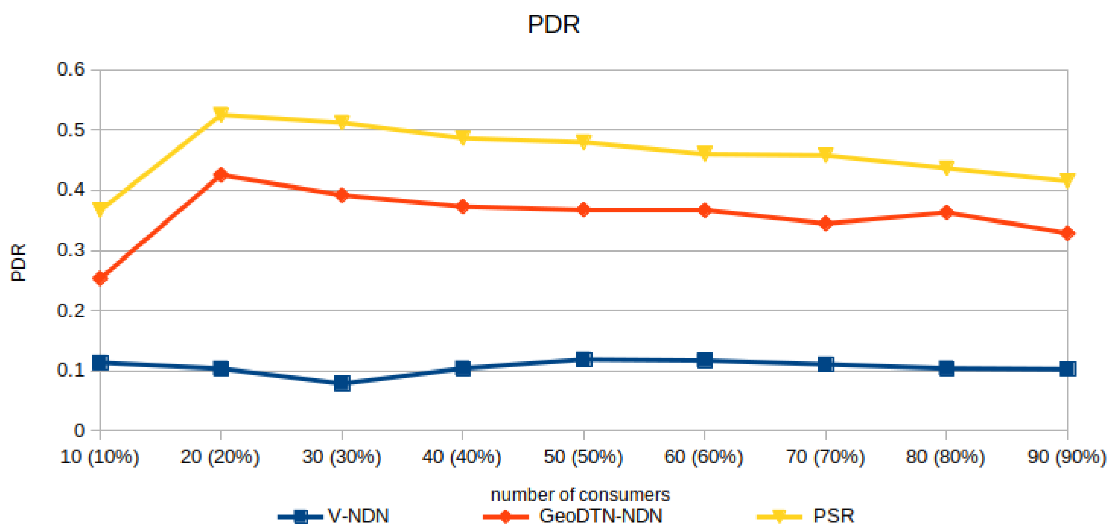

Figure 4.7: PDR for the number of consumers

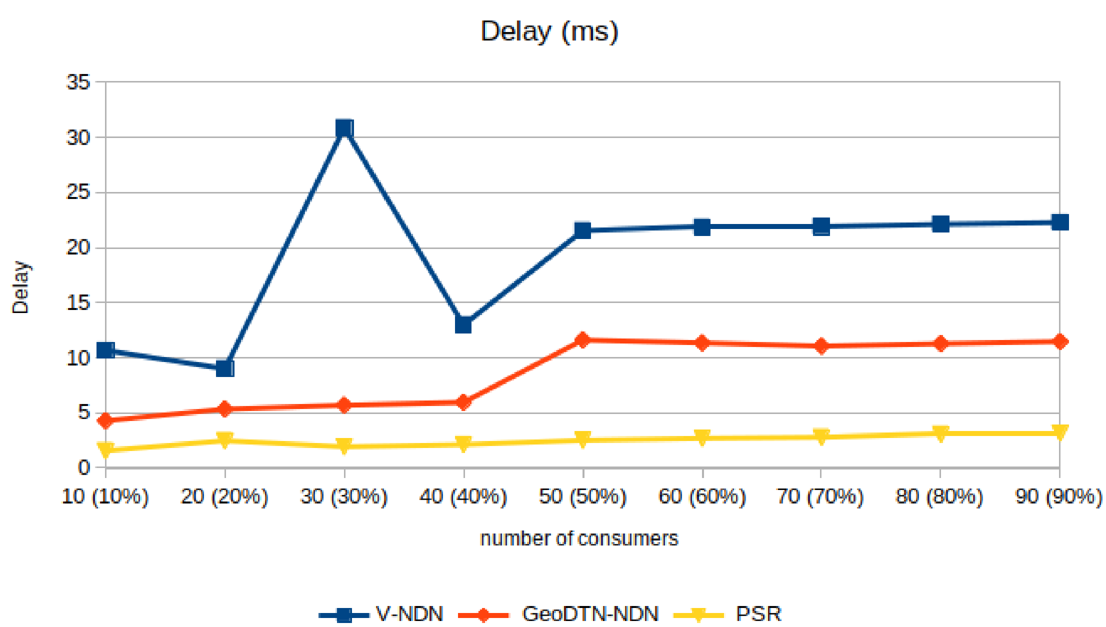

Figure 4.8: Delay for the number of consumers

35.67% higher throughput than GeoDTN-NDN and 65.03% higher throughput than V-NDN, when the number of consumers is 50%. PSR obtained 61.2% and 80.89% increase when the number of consumers is 90% compared to V-NDN, and GeoDTN-NDN, respectively. The PSR outperforms when the number of requests is higher because its selective forwarding strategy.

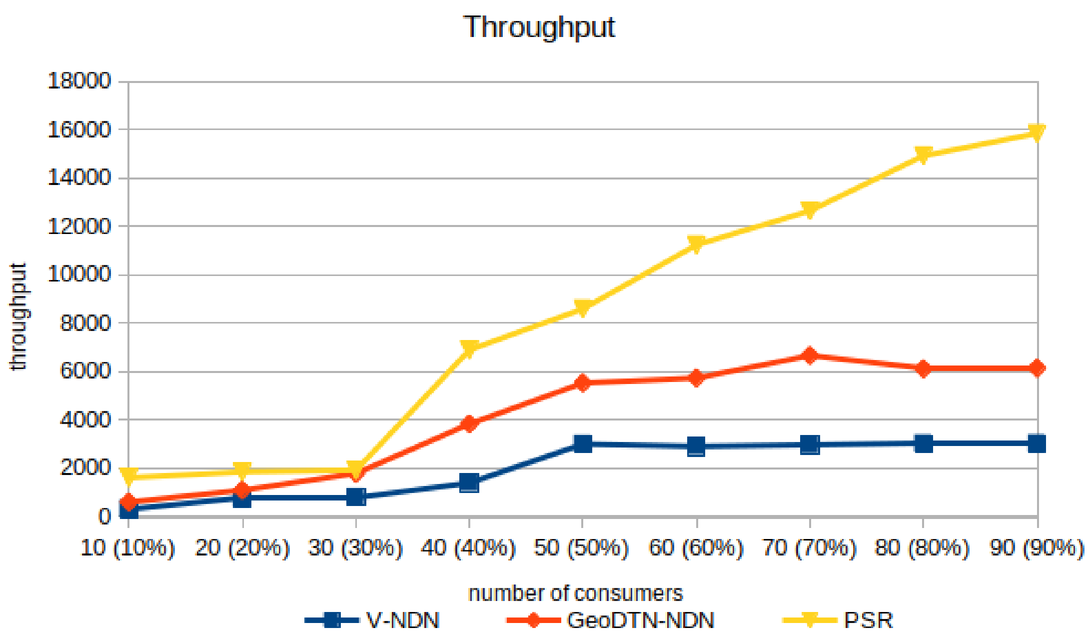

Figure 4.9: Throughput for the number of consumers

To conclude, PSR provides an efficient solution that improves the overall performance of Vehicular Named Data Networking. It achieves higher packet delivery ratios, lower delays, and higher throughput in both evaluated scenarios.

4.4 Summary

We proposed a progressive segment routing approach for interest forwarding in vehicular Named Data Networking. Our approach is based on the source routing solution using the map graph, with the cost of the edges adjusted by the density information collected. By doing so, we can avoid empty or less traveled roads that have fewer vehicles to forward the traffic. We allowed intermediate nodes to update the routing path based on updated conditions in its proximity, thus creating multiple segments in the routing path. The progress condition imposed on who can update the source routing path guarantees that each routing segment makes progress toward getting to the final destination. Simulation results show that progressive segment routing can improve the packet delivery ratio, reduce the packet delivery time and improve the throughput.

Chapter 5

Applications of Vehicular Named Data Networking

5.1 Overview

In this chapter, we explore applications of Vehicular Named Data Networking (V-NDN) and study a particular application, smart parking application, in detail.

V-NDN applications have a variety of behaviors, characteristics, features and requirements and some applications may have Quality of Service (QoS) requirements. We present a classification of NDN-based VANET applications and introduce an NDN naming scheme to meet the requirement of these applications. We observe that these applications may produce a high volume of traffic in the network and some safety-related applications may require on-time packet delivery to handle emergency situations. Other applications such as video streaming may be bandwidth-sensitive because the quality of user experience may be degraded if the minimum bandwidth requirement is not met. To deal with these problems, we design a priority model to handle different types of applications and utilize the NDN naming scheme to prioritize different types of applications.

The second part of this chapter focuses on a particular application – the smart parking application. We design an NDN-based framework for the application to answer the query from drivers about whether there is parking space available in a particular geographical area. Further, the application allows drivers to make a reservation for a period of time ahead of time. We assume sensors are installed in the parking lot to collect availability information. Sensors on vehicles can also contribute to the parking application. In order for sensor nodes to deliver the information proactively, we introduce a "push-based model" to integrate sensor nodes into the system. We adopt a fog computing architecture for collecting, processing, storing, and computing data for the system and bring cloud services close to the edges.

Section 5.2 presents the categorization of NDN-based VANET applications. Section 5.3 gives a naming scheme designed for these applications. Sections 5.4 and 5.5 present the design of the smart parking application and simulation results, respectively.

5.2 Categorization of NDN-based VANET Applications

NDN applications use the Application Data Unit (ADU) [43] to allow the producer to publish its content with a unique naming scheme. Meanwhile, the consumer can request the data content by using the application name. There are a wide variety of applications that use NDN in VANET applications to bring the benefits of NDN to these applications. In some applications, the consumer has to make the request before the producer can send the content to the consumer, while other applications allow the producer to broadcast content no matter whether there are requests for the content. Some content is associated with a geographical location, while other content is location independent. Another difference we observe is how long the content is propagated inside the network, whether it is limited to the local vehicle-to-vehicle communication. Based on these factors, we can categorize NDN-based VANET applications into four groups.

1. Applications that are interested in location-associated information (type A). Vehicles in these applications want to get the information associated with a geographical location, such as requesting the traffic situation on a specific destination or querying

the availability of parking slots in a geographical area. The consumers have to make the request by sending the interest packet. The producers can respond to the request from the consumers to meet their needs. The content usually is limited to local transmission. The forwarding strategies for these applications can take advantage of geographical location information. For example, the proposed GeoDTN-NDN and PSR forwarding strategies in the previous chapters are targeting these applications.

2. Applications that aim to increase the passenger comfort and provide entertainment such as Internet access, video streaming, and audio streaming (type B). These applications are similar to type A applications except that the content typically is not associated with any geographical location. This makes the forwarding of NDN interest packets harder because no geographical information can be used to guide their forwarding. However, these applications typically take advantage of the caching ability of the NDN architecture. When the interest packet reaches intermediate nodes with the content, the request can be satisfied by using the cached content. These intermediate nodes can be normal vehicles, road side units, or some gateways that are a part of the NDN network. There are several kinds of applications in the group.

   (a) Live media streaming for comfort purposes. In live media streaming applications, media content (video and/or audio) is playing out immediately when the content arrives at the consumer side, rather than waiting to download the whole media file completely. The producers of the streaming content organize the content into frames and send over the communication channel to the consumers. Many factors can degrade the quality of media, such as noise in the transmission channel and coding/encoding techniques at the producers and consumers [44]. Another important factor is the bandwidth available for the application. When it is below a certain threshold, the video/audio can be illegible. It is critical to provide a certain level of quality of service guarantee or give priority to the traffic that is sensitive to the available bandwidth on the path.

(b) Video on Demand streaming (VoD) of prerecorded video. VoD is a type of media application that allows a consumer to select a video from their preferred set of videos and enables them to watch it at any time. The consumer can control the playing, such as pause, fast forward, and jump backward. VoD in VAENT is still challenging due to the similar quality requirement as live media streaming applications.

(c) Interactive video/audio applications. These applications support two-way communications between two vehicles such as video/audio conferencing and online video gaming. Interactive video applications require high bandwidth and short end-to-end latency so that the experience is similar to real-life interactions.

(d) Video/audio downloading. In these applications, a video/audio file is downloaded from its producer to the consumer's device. The file should be fully downloaded to the hard drive before it can be played out. Having a copy of the file allows the vehicle to transmit it to others. Vehicles can get the content from RSUs or nearby vehicles via V2V communications.

(e) Internet access. These applications include Internet browsing and email access, etc. Accessing the Internet in VANET allows vehicles to get information at any time.

3. Push-based applications (type C). These applications aim to improve safety and reduce accidents, such as real-time traffic and post-crash/hazard notifications. The real-time data should be pushed in NDN to the intended receivers in a short period. So the consumers of the content do not need to make the request by sending interest packets. Instead of pulling the content by the consumers, the content is pushed by the producers to a certain scope of consumers. The content can include the following.

(a) Accident/hazard notifications. Once an accident happens or a hazard is found, vehicles close to the event can inform other vehicles about the accident/hazard location via broadcast. The content pushed can warn other vehicles to avoid the location, reduce collisions, and request help.

(b) Video broadcast for the safety propose. Smart vehicles are equipped with cameras that can capture real-time events and push them to other vehicles, especially first responders who may evaluate the seriousness of the situation and take appropriate actions. Video broadcast applications are very similar to the live media streaming applications of type B. The difference is that video broadcast applications are push-based applications in which the content is pushed to consumers without their requests, while live media streaming is pull-based applications in which consumers make the request and the corresponding content will be live streamed to them.

For type C applications, the content can also be published to the nearest RSU first. Then, the RSU will make the content available for other vehicles.

4. Local vehicle-to-vehicle communication applications (type D). Communications in these applications are limited between vehicles in the same area within their transmission range. Vehicles can communicate with each other about changing lanes, the distance between vehicles, and adjusting speed, etc. This one-hop communication is the simplest, but still important because it may reduce the possibility of accidents and improve safety.

5.3 NDN Naming Scheme for VANET Applications

NDN naming uses unique, hierarchical, global, and human-readable names. The hierarchical namespace enables name aggregation and scales the routing system. Designing a naming scheme in NDN is the basis for identifying the desired data and retrieving them.

We design a naming convention that can cover the types to applications we categorized above. The structure of data names is based on the application type. Each type may have its own special routing approach.

- Type A. For type A applications, the naming scheme needs to represent the Goe-area information in the hierarchical name structure. It can be encoded based specific applications and routing algorithms. The format of type A applications is /type of application/data type/data name/timestamp/Geo-area. In this format, the “type of application” component indicates it is type A. “Data type” component represents the type of data the consumer requests, for example, a request for traffic information or a request for parking information. “Data name” represents the specific data the consumer requested for the data type. “Timestamp” can use the same UNIX timestamp to indicate the time the request is made or the data is sent out. “Geo-area” component uses an encoded format to represent the geographical locations the consumer is interested in or the data comes from.

- Type B. For media applications (live streaming, VoD, etc.), there are two types of names. One is the meta information about the media (video/audio). This includes information such as frame number, frame rate, frame width and height, and encoding format. The name format for meta informaion is (/type of application/data type/stream-ID/video or audio/stream-info). In this format, the “type of application” shows that the type of application is type B. “data type” represents whether it is live streaming, pre-recorded video streaming, interactive applications, media downloading or Internet access. “stream-ID” is used to distinguish different streams. The “video or audio” component represents whether it is video stream or audio stream. And “stream-info” indicates what specific stream information the request is trying to get.

  The second type of names is about the content of either video or audio streaming. It can be represented as /type of application/data type/stream-ID/video or audio/segment number/frame number/timestamp. “type of application”, “data type”, “stream-ID”, and “video or audio” represent the same as in the meta-information name. “segment number” represents the interested segment. “frame number” identifies a specific frame in a video or audio. And “timestamp” indicates the time in the same format as UNIX.

The format of the Internet access names is /type of application/data type/timestamp. In this format, “type of application” represents it is type B, “data type” represents it is for Internet access, and “timestamp” component represents the time.

- Type C. For pushed-based applications, it can be accident/hazard notifications or broadcast video. For broadcast video, we can use the similar format as in the media applications of type B. The only different is that the type of application is type C. Further, the broadcast message may limit the scope of broadcast to a certain geographical location by adding a geo-area in the name. For accident/hazard notifications, the name format is /type of application/data type/timestamp/Geo-area. In this format, “type of application” indicates the type of application and in this case, it is type C. “data type” represents push-based notifications. It also includes a “timestamp” and “Geo-area”, which is used to limit the notification scope in a certain geographical area.

- Type D. In type D applications, the fundamental communication will be between nearby vehicles, without involvement of remote vehicles or the infrastructure. The naming format is /type of application/data type/position/timestamp. In this format, “type of application” indicates it is type D. “data type” represents what type of the data is included, e.g., information about changing lanes, the distance between vehicles, and adjusting speed, etc. “position” represents the location of the sending vehicle so that the receiving vehicle can know the relative locations. “timestamp” represents the time it is sent out.

5.3.1 A Priority Model

In VANETs, some applications are more time sensitive than others, such as applications related to safety proposes. For those time sensitive applications, it is preferable to give the traffic from them a higher priority so that their packets can arrive at the consumers in time. While the classification of applications into different types give a sense of time-sensitivity, there is no direct one-to-one mapping between the types and the priority level. For example, Internet access is not that time sensitive as other multimedia applications, even though they belong to the same type.

We define three priority levels, so that the forwarders in VANET can give preference to traffic from high-priority applications when making forwarding decisions. These priority levels and other features can be incorporate in the PIT table in the NDN architecture. High priority level means that the forwarder should use as much resources available as possible to handle their messages so that it can achieve higher bandwidth allocation and shorter delays. Low priority level means that messages can be delayed in the PIT by waiting until the higher priority level traffic is delivered. Medium priority level is something in between. In NDN, the PIT table maintains incoming interest packets until the request for that interest is satisfied or a timer expires. We can modify the PIT so that even if there are earlier messages in the PIT, higher priority traffic will be forwarded even if it arrives later in the PIT than existing earlier messages in the PIT.

For the four categories of VANET applications, we can analyze the priority levels we should give based on the nature of applications. For example, local vehicle-to-vehicle communications tries to avoid car accidents and should be given a high priority. The following is the priority level for each type of application:

- High: all safety applications (type C and type D applications).
- Medium: multimedia applications that are time sensitive (type B media applications).

- Low: applications such as requesting a traffic situation on a specific Geo-area and or simply getting access to the Internet (type A and type B Internet access applications).

5.4 Smart Parking Application

Smart parking application is developed to provide two functions. One is to allow a driver to find parking lots that have available park spaces. The driver can send out the request as an interest packet in the NDN architecture and the replies can come from multiple parking lots. It is up to the driver to pick any specific parking lot. Normally, the driver wants to find available parking space at this moment. We will discuss a case in which the driver may be interested in the availability in the future. This is related to the function we will discuss next. Typically, the driver wants to find the parking lots in a particular geographical area. Based on the categorization, it is a type A application.

The other function is to allow a driver to reserve a parking spot in a particular parking lot for a period of time. This can happen after the driver finds a list of parking lots with available space in some future time. Then the driver picks one to make the reservation with the arrival time and departure time specified. Similar to the previous function, the driver can also provide geo-area information in the request packet so that it can be forwarded to the correct location, rather than using the broadcast.

The smart parking application adopts a fog cloud architecture to provide a flexible structure and improve the response time. NDN naming scheme is designed to identify the request, either for finding an available parking space or making a reservation for the future. We introduce a push-based model into the NDN architecture to streamline the process for allowing sensor nodes installed to get the information into the system.

5.4.1 Components of the Smart Parking Application

We consider a topology with roads with traffic on both directions and interactions where decisions are made to change directions. In addition, sensor nodes will be installed to

Figure 5.1: Network Topology

collect real-time information, and cloud computing elements will be used to meet the communication and computing requirement of the application. Figure 5.1 illustrates the main components.

1. ***Vehicles***: Vehicles can communicate with other vehicles and with infrastructure units (fog nodes). They are equipped with a Global Positioning System (GPS), which can tell the location of the vehicle. We can classify vehicles into two types. One is the static vehicles (parked vehicles) that are parked in a parking garage or the curbside of roads. The other is dynamic vehicles (mobile vehicles) that are moving.

2. ***Sensors***: Sensors are installed to monitor a variety of status. In particular, the parking application may have sensors to monitor whether a parking spot is free or not. In addition, modern vehicles are equipped with a lot of sensors that can also contribute information to the parking applications, such as location, speed, and direction.

3. ***RSUs***: Road Side Units (RSUs) are computing, storage and communication resources install on the road side for supporting modern transportation systems. They are an important component of VANET and will be a good candidate as the fog node in the fog cloud architecture. However, not all roads have RSUs installed. Vehicles and

RSUs communicate through WiFi IEEE802.11p/WAVE technology and the quality of the communication can be significantly better than vehicle to vehicle communication.

4. ***Dedicated Servers***: Dedicated servers may be installed in big parking garages for managing the parking space. They can also be incorporated into the fog cloud architecture.

5. ***Central Cloud***: Central cloud provide a central storage and processing service for the fog nodes connected to the architecture. It provide a global information necessary for coordination among the fog nodes.

5.4.2 Fog Cloud Architecture

We considered several architectural candidates for the smart parking application. One possibility is that we can depend on RSUs and dedicated servers on larger parking garage to handle the requests, without any help from the cloud architecture. There are four limitations. First, there are smaller parking lots without dedicated servers or not covered by RSUs. We will not be able to include those parking spaces into the application. Second, if the dedicated servers or the RSUs crashes, the information stored there will be lost. For example, the reservation information will not be able to be recovered and agreements with customers will not be able to be honored. Third, different parking lots will not be able to cooperate with each other, such as recommending alternative parking places if the current lot is full. And finally, deploying RSUs and dedicated servers can be costly.

The second possibility is that we can use a centralized cloud server to handle all the requests from users. There are two potential issues. First, the central cloud becomes the concentration point of all requests. It can take a longer time for a user to get the response than the case in which a request can be processed by a distributed and close-by server. Second, the state information collected by the sensors will be sent over to the cloud. This can be huge, depending on the scope of parking spaces the central cloud is handling.

Instead, we consider an architecture based on the fog cloud computing. We still have a central cloud service that has a large pool of processing and storage resources that can be allocated on demand to meet the ever-changing requirement of the system. In addition, we have fog nodes distributed in different parts of the system, located at the edge of cloud and close to the parking space they serve. They will handle the request for the spaces they are in charge. The central cloud server also acts as the backup service in case of some critical states at distributed fog nodes are lost. It also has the authority to manage the fog infrastructure by authorizing the setup and teardown of fog nodes. The fog nodes can be RSUs, dedicated servers, or even vehicles. Usually status information generated by the sensor nodes is sent to the fog nodes. Only aggregated information or critical states will be forwarded to the central cloud. For example, whether a particular parking spot is occupied is kept at the fog node, while the total number of free spaces may be forwarded from the fog node to the central cloud server.

For large parking garages, dedicated servers will be the first choice as the fog node, because their processing units are more powerful and storage space is larger. Similarly, when there is an RSU next to a parking lot, we will prefer using the RSU as a fog node to take care of processing requirements. These two kinds of fog nodes are easy to set up and we will not focus on them in our following discussion. However, for parking lots that do have dedicated servers available or are not covered by RSUs, or they are out of service due to hardware or software failure, we propose to utilize the capability of Vehicular Fog Computing (VFC) [45, 46, 27] to set up vehicle-based fog nodes to handle the requests for these lots. The VFC is especially useful for handling the cases such as curbside parking and small parking areas.

We have to address several issues. First, we need to select a vehicle as a fog node. Generally we prefer to select select a parked vehicle because it will stay in place for a longer period of time. Second, We need to enable vehicles and sensor devices to communicate with the selected fog node. The fog node maintains computing, storage and communication capabilities to handle users requests. Third, we need to connect the fog node with the central could so that it can be authorized to make decisions and be trusted by sensor nodes and other vehicles. Also we need to handle the transition to other vehicles once the current fog node needs to leave the location. The rest of this section will describe our approaches in detail.

5.4.2.1 Selecting Fog Nodes

Ultimately the central cloud has the authority to select which vehicle can be the fog node for a particular parking space. Information about potential candidates – parked vehicles – should be obtained by the central cloud to make the decision. If there are no existing fog node available, interested parked vehicles can send to the cloud directly. This may be done by vehicle to vehicle communications until the message reaches an RSU or other fog node handling a different parking area that has a channel to the central cloud. If there is an existing fog node in the area, it can collect the information about parked vehicle and send them to the central cloud so that a new fog can be selected once the current fog node needs to leave the area.

The attributes about parked vehicles that will be sent to the central cloud include:

1. Vehicle ID: the identification of the vehicle.
2. Current location: the x and y coordinates of the vehicle.
3. Storage size, computing and communication capacities: the processing power of the vehicle.
4. Parking duration: represents the time that the parked vehicle will stay there.
5. Willingness: a vehicle is willing to participate in forming a vehicular cloud. A vehicle sending the information usually has a high level of willingness to serve.

The central cloud maintains a data repository to gather vehicles and the fog cloud information. Based on the data collected regarding the vehicle's information, the central cloud uses some criteria to select the fog node as the initial setup for a parking space or as the replacement for the current fog node that is going to leave the area. First, the central cloud have a set of thresholds and only those nodes with metrics equal to or greater than the thresholds can become candidates. We give a list of criteria in the following.

1. Computing, Storage and Communication Capacities. Computing, storage and communication capacities vary from vehicle to vehicle. The central cloud specifies a threshold and filter out all those vehicles that do not meet the threshold requirement.

2. Duration of Being Active. Another factor to consider is how long the vehicle can provide service. The central cloud will balance the capacities and time being able to serve.

3. Behavior Rate. In addition to the vehicle's willingness to server, the central cloud also maintain information about the past behavior of vehicles. Vehicles with high ratings will be preferred. The central cloud performs an adaptive adjustment to rate the vehicle node's behavior based on its previous monitoring of the participating vehicles. Each time a vehicle participates as a fog node, the cloud monitors the fog nodes and rates them based on how long they serve and performance obtained by other vehicles.

5.4.2.2 Communications with Fog Nodes

After a static vehicle is selected the fog node, it will broadcast an *advertisement* message to inform other nodes in the same Geo-area, including sensor devices and other vehicles. The following naming convention is used for the advertisement message: "/advertisement/Fog_nodeID/current position/type/duration". In this format, "advertisement" indicates the type of message, which is a broadcast message; "Fog_nodeID" represents the identification of the selected fog node; "current position" indicates the current position of the fog node at the time; "type" indicates whether the fog node is an RSU, a dedicated server or a parked vehicles; and "duration" shows the duration in which the fog node will be active.

After receiving this message, other nodes will know information about the fog node, including Fog_nodeID, current position, type, and duration. The device (e.g., vehicle or sensor) can include the interest name of the advertisement message in its PIT table. Intermediate nodes in the NDN architecture also record it as a pending interest with the incoming interface. When the PIT entry is set up in the PIT table of a vehicle or a sensor node, it can push any sensed data messages or send any other requests regarding the smart parking applications to the fog node. The difference is that the device can periodically send updated state information, not just one message to meet the requirement of the interest packet. The device can identify the message as push-based status information as being associated with the fog node using its "Fog_nodeID". To support the push-based model, we need to modify the PIT table entry at intermediate nodes. Instead of removing the entry after one message meeting the interest is forwarded via the incoming interfaces, it will keep the entry for the duration specified by the during in the advertisement message sent by the fog node.

5.4.3 Finding Parking Lots with Available Parking Space

The first function we want to implement for the smart parking application is enable drivers to find parking lots with available parking space. We assume that drivers send a query to check parking space availability for a specific area. When these interest packets arrive in the target area, all those parking lots with available parking space will reply.

In order to realize the function, we assume that sensors (IoT devices) are installed to monitor the parking space in each parking lot. We assume that there are n number of parking spots denoted by $P = \{p_1, p_2, ..., p_n\}$. The real-time information of a parking spot can be collected by a parking sensor, and it will be pushed to the associated fog node. The fog node is responsible for collecting parking space information regarding occupancy and availability. It maintains a data structure, which is composed of a list of three-tuple: (parking Id, occupancy status, vehicleId).

1. parking Id (i): each parking spot is identified by a unique parking Id and is associated with the corresponding position (x_i, y_i) on the Euclidean plane.

2. occupancy status (OS): it indicates the status of the monitored parking space. It could be vacant or occupied, which can be represented in 5.1

$$OS(i) = \begin{cases} 1 & \text{if a vehicle is parked at parking spot } p_i \\ 0 & \text{if the parking spot } p_i \text{ is vacant} \end{cases} \tag{5.1}$$

3. vehicle Id: represents the vehicle's identification that occupies a parking space. This field is only used when the occupancy status value is 1; otherwise, this field will be empty.

To answer the request from the user whether there are available parking spaces, we can simply find the value of $\exists i : OS(i) = 1$. This is only true if we do not accept reservations. In case the fog node allows drivers to reserve parking space ahead of time, we will need a more complicated data structure, which will be described in the next section, to answer the question of whether there is a parking space available.

The naming format of the interest message for requesting whether there are available parking space can be: "application_type/Application_name/data name/timestamp/Geo-area". In this format, "Application_type" should be type A, indicating that it is an application interested in location-associated information. "Application_name" indicates it is a request for parking information, i.e., it is smart parking application; "Data name" represents that the request is specifically requesting for parking lots with available parking spaces. "Timestamp" specifies the time the driver is interested in the availability information. Normally it is the current time for requesting current availability. For the fog nodes that allow reservation, the query can also ask whether the parking lot has space available in some future time. This requires more complicated processing but can be accommodated with the reservation function discussed in the next section.

When a vehicle needs to look for a place to park, it can send a request as an interest packet for finding parking lots with available parking spaces. This request includes the geo-area information indicating the area the drive is interested in. This interest packet can be forwarded by other intermediate nodes using the algorithm proposed in the previous chapters to the desired target geographical area. The vehicles can use the V2V communication or take advantage of whatever infrastructure is available.

Upon receiving an interest message, an intermediate node will check its CS for matching content. If content with a matching timestamp is found, it will forward the corresponding data back to the vehicle making the request. Otherwise, it will check its PIT table for the same interest name. If the entry with the same interest name exists, the incoming interface of the interest will be added to the corresponding interfaces field in the PIT table. The interest entry in the PIT table will be removed if the entry timer expires.

When a fog node is in charge of a parking space in the target geo-area, it will process the interest packet and send back the data packet with related information. This data packet will be forwarded back to the vehicle making the request.

NDN fits perfectly with this function of the smart parking application because multiple drivers can send interest messages requesting the same content about available parking space. These requests will be aggregated along the way, except the fusion point will have multiple incoming interfaces, which will be used to forward data packets back to these drivers. All drivers can be satisfied with the same corresponding data packet from either the fog node or any caches at the intermediate nodes along the way.

5.4.4 Reserving a Parking Space

The other function of the smart parking application provide is to allow a driver to reserve a parking spot before his/her arrival from any location he/she currently is. The reservation will be based on the First Comes First Serve (FCFS) basis.

For the reservation application, we use the fog node to maintain the data related to parking spaces located at the same location and process the corresponding transactions locally. Also the fog node will use the central cloud as the backup service to maintain crucial states including existing user reservations. The central cloud also play a role of managing the setup of a new fog node and transition old states to it.

Multiple drivers may make reservations from the same parking lot with the same fog node. Each requester should be satisfied differently with the unique data content related to the request. NDN can satisfy multiple interest packets that carry the same name by fetching the original data from the producer or any intermediate node's corresponding data content. In this case, the data content will be the same for all the consumers. However, the case for reservation in the smart parking application is different in the sense that each requester should be satisfied differently, even though their interest packet may have the same name.

To deal with the issue, we need to make some modifications to NDN packets by extending the interest and data packet fields. The following fields are added to the interest packet: [my_id, my_arrival-time, my_departure-time, parkingSpace_location, my_position], where "my_ id" indicates the identification of the sender vehicle, "my_arrival-time" is the earliest possible arrival time, "my_departure-time" is the latest possible departure time, "parkingSpace_location" specifies the parking lot the driver wants to make the reservation, and "my_ position" indicates the sender vehicle position.

When an intermediate node receives the interest packet regarding the reservation, the interest name will be included in the PIT table if there is no matched data in the CS. The current NDN forwarding daemon has no concept about sending the same interest name and satisfying each consumer with different data packets. We introduce the concept of Identity of Vehicle Interfaces (IVI) by modifying the PIT table to include the IVI, instead of the incoming interface field. The IVI entries include the binding of the incoming interface to the vehicle ID of the requester and a version number. This binding can be presented as

<incoming interface, vehicleID, version>, where "version" is used to indicate the version number when a requester vehicle updates its route and sends a new version of the same interest. This binding helps to send the data packet to each requester vehicle separately. Once the intermediate node receives the data packet regarding a reservation request, the PIT will only satisfy the request with the same vehicleID. Only the IVI with the same vehicleID will be deleted. All other IVIs will remain there until the data packets matching their ID arrive. The PIT entry will be deleted if the data packet fails to return by the expiration time, or if the requests from all vehicles are satisfied.

When the reservation request arrives at the fog node responsible for a parking space, it extracts three tuple $< vid, r_a, r_d >$ from the request. vid is the id of the driver, r_a is the arrival time the driver expects to arrive to start parking the vehicle in the parking lot, and r_d is the expected departure time.

We assume that there are n parking spots in the parking space, denoted by $P = \{p_1, p_2, ..., p_n\}$, as shown in Figure 5.2. The system will allow drivers to reserve for a certain period of time (e.g., a week, a month, etc.). The y-axis represents the n parking spots in the parking area. The x-axis indicates the time. Any reservation (as marked) is an interval on the x-axis.

The data structure used to represent the reservations is shown in Figure 5.3. Any parking spot p_i consists of a list of reservations. Each reservation is represented by a node $< vid, a, d >$, where vid is the identification of the vehicle this spot is reserved for, a is the arrival time, and d is the departure time. $< a, d >$ represented an interval of the reservations. There is no overlap between all reservations for any given spot. To make the search algorithm for finding a spot that is available for the duration of a new request, we make sure that the list is sorted in increasing order based on the arrival time. If a spot has no reservation, the list is empty, and the pointer from that spot is NULL.

The system searches for a parking spot that is not reserved for the period of time $< r_a, r_d >$. If the system finds such an available interval, it reserves the parking spot for the

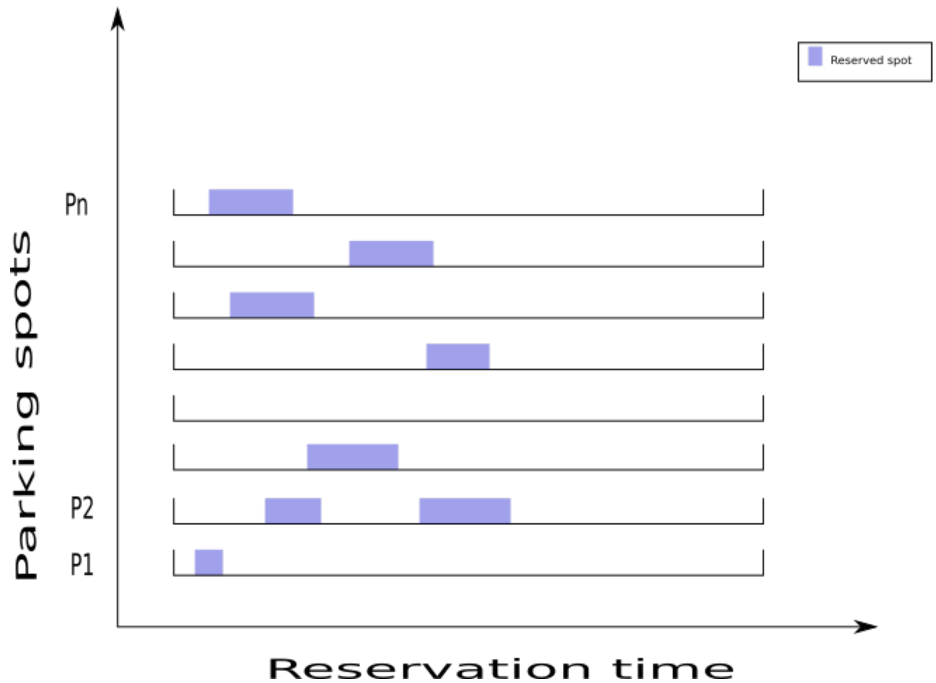

Figure 5.2: Schedules for the Reservation System

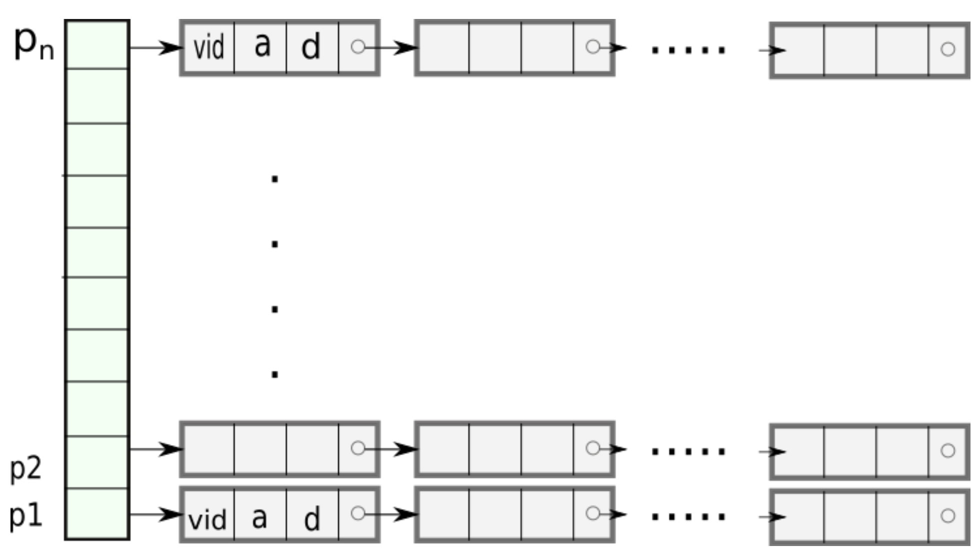

Figure 5.3: Data structure for the reservation application

requested duration, stores the reservation information and confirms the reservation. The reservation is in the format of $< vid, r_a, r_d, spot >$, where *spot* is the parking spot reserved for a vehicle identified by *vid*. If no such spot is found, the system will report failure. Figure 28 shows the detail of the algorithm for reserving a parking spot. It goes through all the parking spots. If it finds a spot that has not been reserved by any vehicle, it will reserve the time interval (lines 2 to 5). Otherwise, it searches through the ordered list of current reservations. If the departure time is less than the arrival time of the first reservation for a node, the reservation will be added to the beginning of the list (lines 6-12). Otherwise, if it can find in the list two reservations such that the requested arrival time is after the first reservation's departure time and the requested departure time is before the second reservation's arrival time, the requested reservation will be put in between (lines 13 to 25). If we still cannot find a parking spot that is available for the duration of the requested reservation after going through all the parking spots, it returns failure.

Drivers can also cancel a previous reservation by sending the tuple $< vid, spot, r_a, r_d >$ to the fog node in charge of the parking space. Figure 19 shows the algorithm for canceling a reservation. It first checks whether the parking spot number is valid or not (lines 1 to 3). If not, it will return with a failure. Otherwise, it searches the list of reservations for that spot (lines 4 to 18). If it finds a reservation that matches the user provided, it will delete the reservation from the list and return with success. If it cannot find the matching reservation, it will return with a failure (line 19).

5.5 Performance Evaluation

5.5.1 Simulation Setup

For the simulations, we used ndnSIM (Version 2.8) [39], which is an NS-3 based NDN simulator. We also used SUMO [40], a microscopic traffic simulation for urban mobility, to generate the traffic. We created a map of a part of the University Campus using OpenStreetMap Wizard [42]. The map size is 2000×2000 meters. We generated different

```
Data: the list of Parking Spots P[1..n], the id of the vehicle making the request
      vid, request arrival time r_a, request departure time r_d
for i = 1 to n do
    if P[i] == nil then
        nptr = new node(vid, r_a, r_d);
        P[i] = nptr;
        return(success, i);
    else
        cptr = P[i];
        if r_d < cptr.a then
            nptr = new node(vid, r_a, r_d);
            nptr.next = cptr;
            P[i] = nptr;
            return(success, i);
        else
            while ((cptr! = nil) and (r_a >= cptr.d)) do
                qptr = cptr.next;
                if ((qptr == nil)||(r_d <= qptr.a)) then
                    nptr = new node(vid, r_a, r_d);
                    nptr.next = qptr;
                    cptr.next = nptr;
                    return(success, i);
                else
                    cptr = qptr;
                end
            end
        end
    end
end
return(failure);
```

Figure 5.4: Algorithm for Making a Reservation

parking places distributed over the map. The vehicle speed limit is set to 20 km/h, and we set the transmission range of the signal of all vehicles to 150 meters. We utilized the ndnSIM parking applications as a starting point and implemented our smart parking application. The parameters for the experimental setup are shown in Table 5.1. We assume that at each parking area, some parking spaces are occupied, and others are not.

```
Data: the list of Parking Spots P[1..n], the id of the vehicle making the deletion
      request vid, the reservation spot, r_a, r_d, where spot is the parking spot
      reserved, r_a is arrival time, r_d is the departure time
if ((spot < 1)||(spot > n)) then
    return("Failure: Invalid spot number");
end
cptr = P[spot];
pptr = nil;
while (cptr) do
    if ((cptr.v == vid) and (cptr.a == r_a) and (cptr.d == r_d)) then
        if (pptr == nil) then
            P[spot] = cptr.next;
        else
            pptr.next = cptr.next;
        end
        return("Success: Reservation deleted");
    else
        pptr = cptr;
        cptr = cptr.next;
    end
end
return("Failure: Reservation not found");
```

Figure 5.5: Algorithm for Cancelling a Reservation

5.5.2 Simulation Results

We evaluated the performance of an NDN-based smart parking application. Our evaluation consists of two parts. One is the evaluation of the function for finding parking lots with available parking spaces. The other is the evaluation of the parking reservation function of the system.

Table 5.1: Simulation Step Parameters

Maximum transmission range	150 m
Maximum transmission range of RSU	400 m
Number of vehicles	20, 30, ..., 100, 150
Vehicle speed	20 km/h
Simulation duration	200 ms
MAC type	IEEE802.11

5.5.2.1 Finding the Availability of Parking Space

We compared our approach with the original NDN system. There are two main differences. One is how sensor nodes send the data to the fog node in charge of the parking space. In the original NDN, the sensor data will be sent to the fog node only if an interest message is broadcast from the fog node to the network. In our push-based model, after initial advertisement from the fog node, the sensor nodes can periodically push the sensed data to the fog node without the need to receive further interest packets.

The other difference is how the vehicles make requests to find whether there are parking spaces available. In the original NDN, vehicles do not know any information about the fog node (an RSU or a selected parked vehicle). They will broadcast an interest message to request the availability of parking spaces. In our approach, vehicles are making the request to a specific geographical area and only those fog nodes in the geographical area may respond to the availability request.

We evaluate the performance using two metrics, response time and push overhead.

1) Response time. It is defined as the time interval from the time when the vehicle makes the request by sending an interest packet to the time when it receives the response (data packet). The number of vehicles varies from 20 to 150.

Figure 5.6 shows the response time when the number of vehicles varies from 20 to 150. We observed that the response time decreases when the number of vehicles increases. One reason is the caching effect of NDN because, with more vehicles, we can have a higher hit ratio at the cache at the intermediate nodes. The response time of our approach is less than that of the original NDN. Specifically, it achieves a 16.74% to 20.7% decrease when the number of vehicles increases from 20 to 150.

2) Push-based Model Overhead: defined as the total number of periodic *advertisement* messages that are broadcast by the fog node over the total number of pushed data packets to the fog node. Figure 5.7 shows that the proposed push-based model achieves very low overhead, which decreases gradually as the number of vehicles increases. Establishing the

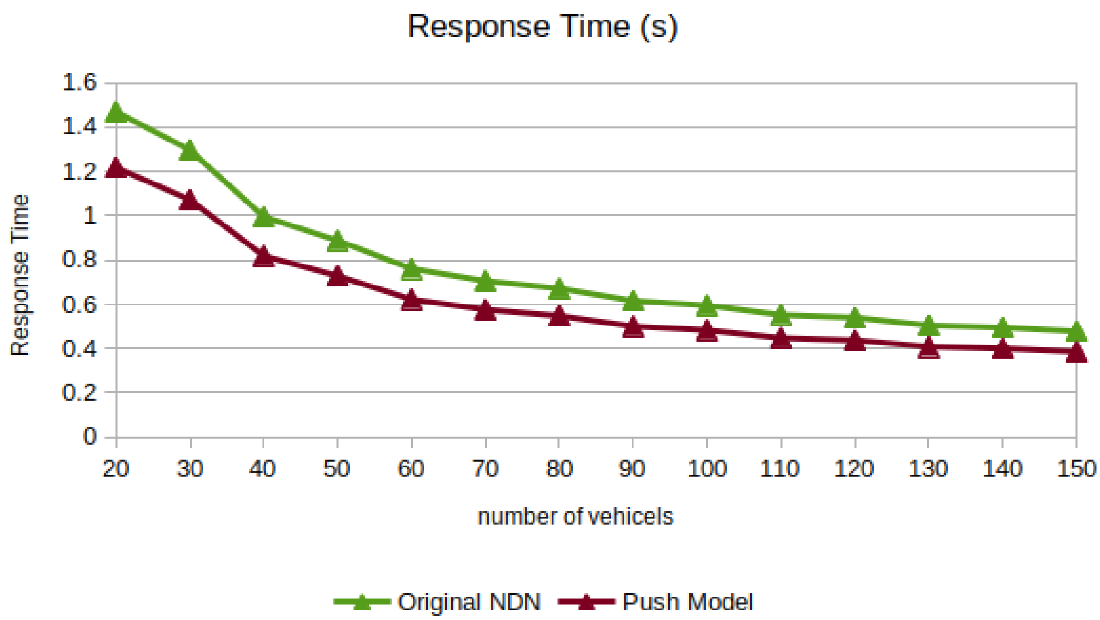

Figure 5.6: Response time for different numbers of vehicles

connection by *advertisement* message helps in generating low overhead because we only need a few interest messages to be sent over the network. The push-based model outperforms the original NDN because we set a communication channel between the fog node and sensors to allow sensors to send periodically status information to the fog node.

5.5.2.2 Reserving a Parking Space

We also evaluated the performance of the proposed reservation application by comparing it with a baseline approach, which allocates a parking space for a vehicle when it arrives at the parking lot without any prior reservation. The parameters for the experimental setup are shown in Table 5.2. Figure 5.8 illustrates the map generated by SUMO. Parking Areas are also identified on the map.

We evaluate the performance using two metrics, success rate, and the average time to park.

1) Success Rate to Find a Space. It is defined as the ratio of the number of vehicles that successfully find a vacant parking space over the total number of vehicles. Figure 5.9

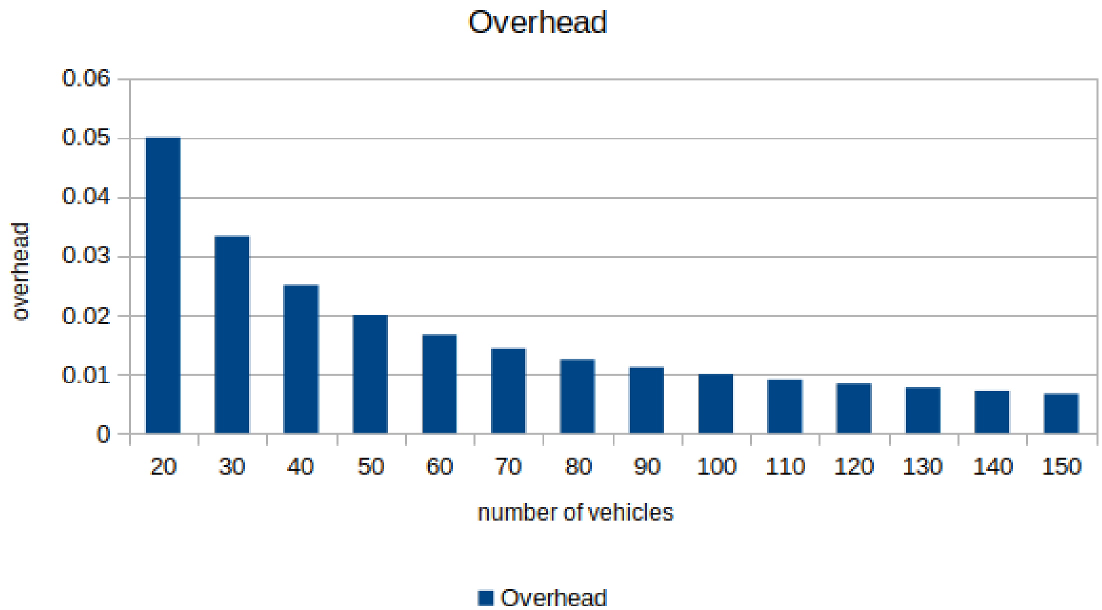

Figure 5.7: Overhead caused by the proposed Push-Model

Table 5.2: Experiment Setup for parking spaces

Parking slots	Total parking spaces
P1	25
P2	15
P3	30
P4	100
P5	60
P6	30

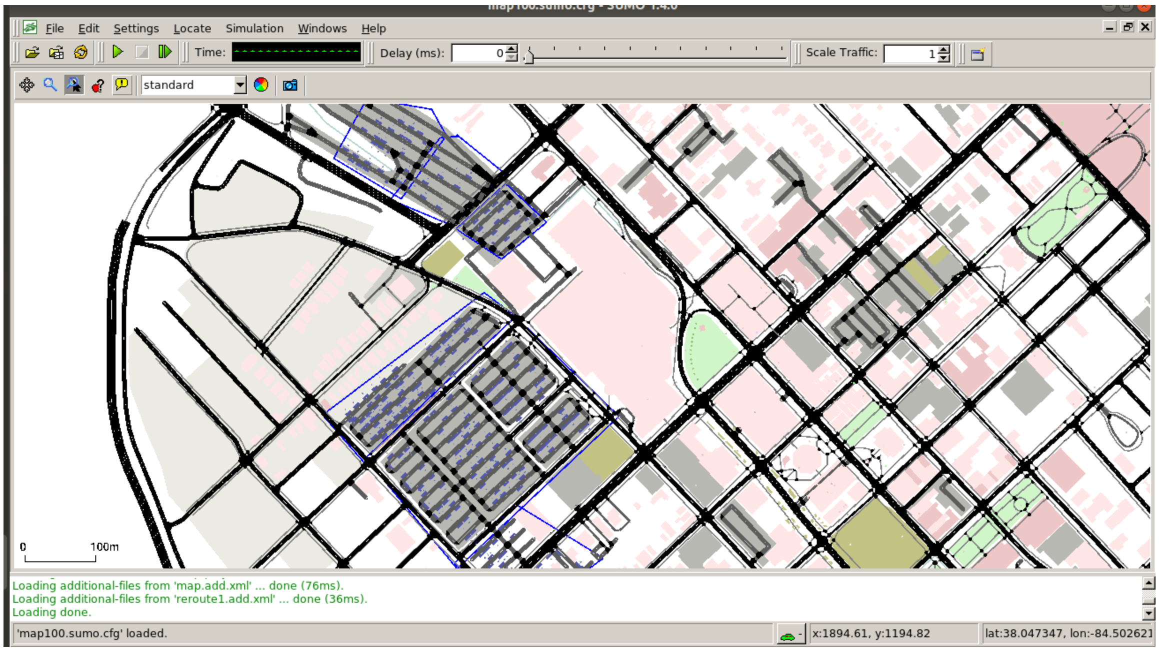

Figure 5.8: SUMO map for a smart parking application

illustrates that when the number of vehicles increases from 20 to 150, the success rate to find a space decreases. Our reservation system has a higher success rate compared with the baseline approach. In the baseline approach, the driver just drives to a parking lot, and if the parking lot is full, the task of finding space is considered failed even if the driver may go to other parking lots and find space later. When the number of vehicles is low, the difference between the baseline approach and our reservation system is small. However, the difference becomes more obvious when the number of vehicles increases. We observed the success rates of our approach are 9.53%, 26.98%, and 53.08% higher than the baseline approach when the numbers of vehicles are 70, 110, and 150, respectively. Our approach can do better because it allows drivers to reserve a parking place before the arrival time.

2) Average time needed to park a vehicle. It is defined as the average time required to find a vacant parking space for a vehicle. Figure 5.10 shows that the number of vehicles has an effect on the average time to park. As the number of vehicles increases, the average time needed to park a vehicle increases. From the figure, we can see that the average time to park for the reservation approach outperforms is smaller than that of the baseline

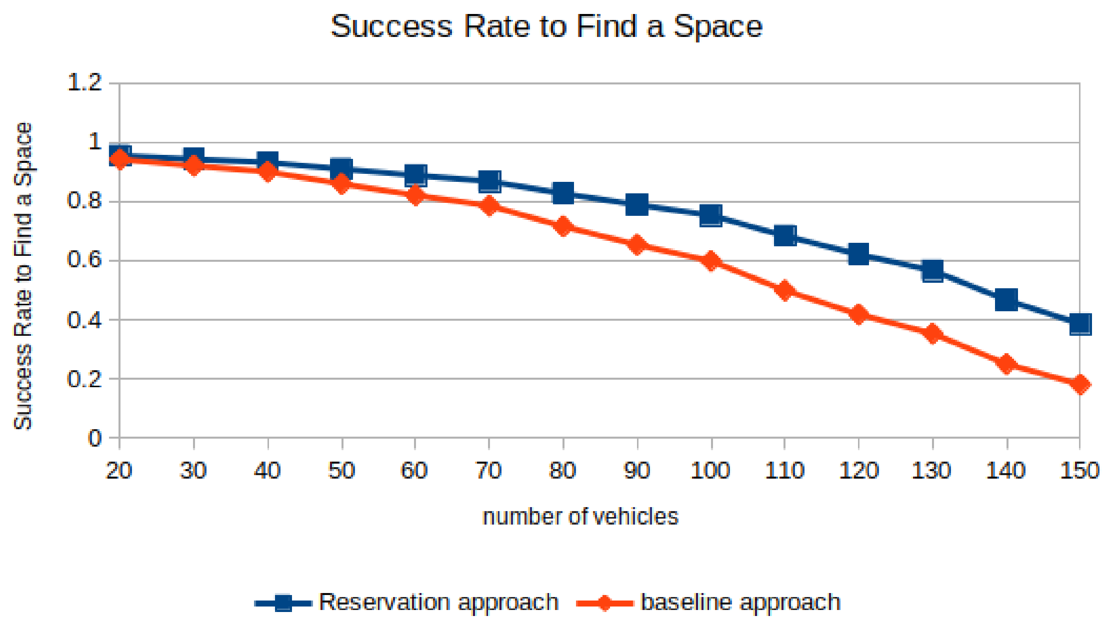

Figure 5.9: Successful Rate of Finding a Space

approach. More specifically, they are between 40.36% and 53.31% less than those of the baseline approach.

5.6 Summary

We categorized different types of vehicular named data networking applications and provided an NDN naming scheme for each type. We considered a priority method to provide better service for some time-sensitive applications. We developed an NDN-based framework for the smart parking application. A fog cloud architecture is adopted so that sensor data can be collected locally and requests can be handled by distributed fog nodes to reduce the load on a central cloud server. The cloud aspect allows us to dynamically set up a new fog node from static vehicle to provide service for those parking spaces without dedicated servers or RSUs. The smart parking application enables drivers to query about the parking lots that have available parking spaces and make a reservation for a parking space for an interval in the future.

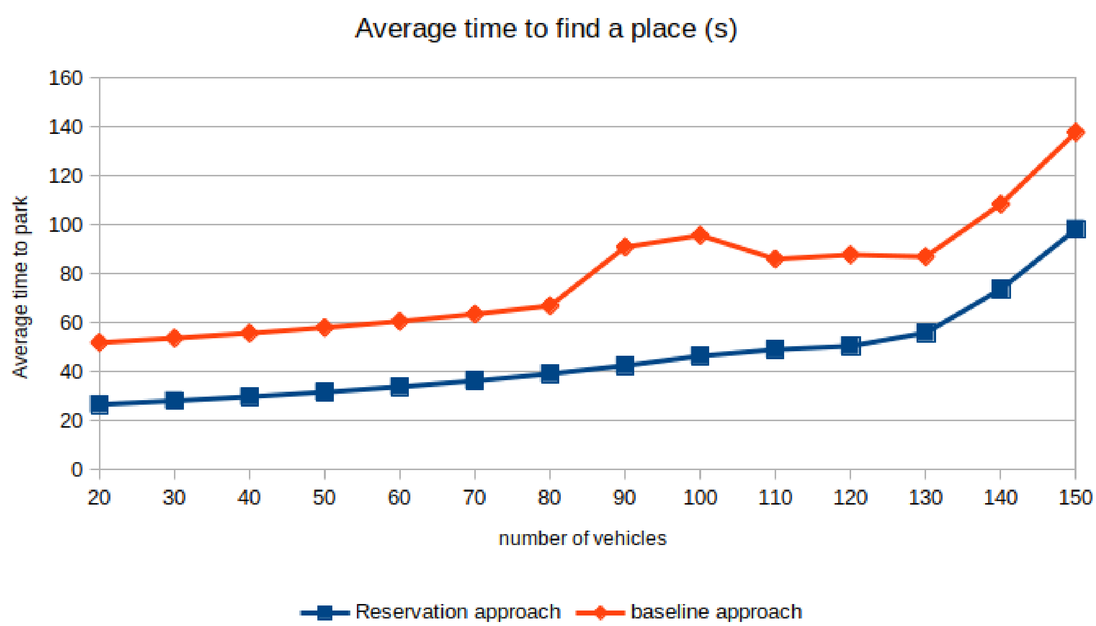

Figure 5.10: Average time needed to park

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this dissertation, we addressed the problem of routing and applications of vehicular named data networking. VANETs have some unique features, such as high mobility of vehicles, intermittent and short-lived connectivity between vehicles, and a wide variety of traffic patterns. These features make routing in VANETs a challenging issue. Traditional IP-based network does not fit well with the VANETs because an IP address is not only the identity of a node but also represents the node's attachment to the network. Therefore, it does not support mobility very well. We adopt NDN as the fundamental architecture for VANETs to take advantage of its focus on the content and its name rather than where the content comes from. This information-centric approach can better support mobility – an inherent feature in VANETs.

We addressed several issues coming with using NDN for VANETs and proposed novel solutions to these problems.

To address the flooding issue caused by broadcast of interest packets, we proposed a new hybrid geographical routing technique called GeoDTN-NDN, which combines greedy, perimeter, restricted greedy, and DTN modes. We described how the routing process should transition between different modes to achieve the ultimate goal of delivering the packets to the destination. Also unique is our approach to addressing the forwarding of data packets. To deal with the high mobility of vehicles in VANETs, we proposed to include geographical location of the consumer in data packets so that they can be forwarded to the consumer, even if the original intermediate nodes in the reverse path of interest packets have moved to different places. This enhanced the original NDN paradigm that depends on the reverse path for data packet forwarding.

To take advantage of the availability of road maps on vehicles, we proposed a progressive segment routing algorithm that calculates forwarding paths at a higher abstraction level based on the maps. We introduced a method to collect density information of the roads and designed the forwarding scheme to avoid less-traveled roads so that the NDN packets can be delivered to the destination without interruption. To accommodate the dynamic situation, we allowed intermediate nodes to re-calculate the forwarding path. The novelty of the progressive segment routing is that we defined a criterion to determine whether the routing is making progress so that we can guarantee that packets will be delivered to the destination, even if a temporary loop may be formed in the routing path.

We also explored the applications of NDN-based VANETs. By categorizing different types of VANET applications, we investigated the space of possible applications and different requirements. We designed an NDN naming scheme for these applications and proposed a priority model that can be used to give preference to those time-sensitive applications. We developed an NDN-based framework for the smart parking application that allows drivers to query the availability of parking space and make reservations for future parking needs. The novelty of our approach is that we combine the NDN routing with the fog computing architecture in the smart parking application, which facilitates the transmission of sensed data using a push-based model and allows dynamic setup of fog nodes for a wide variety of parking situations. Simulations demonstrated that our proposed approaches can improve the performance of routing and applications of vehicular named data networking.

6.2 Future Work

Routing and applications of vehicular named data networking is an interesting area worth further exploring. We will discuss two aspects which we are particularly interested in.

6.2.1 Security in Vehicular Named Data Networking

We intentionally left out the discussion of security and privacy issues in previous chapters. We think they are worth investigating as a separate topic. One of the unique features of NDN is that it provides data integrity by adding a digital signature to data packets. It ensures that the receiver of data packets can verify the signature and know that the data packets come from the claimed sender. This depends on the establishment of Public Key Infrastructure (PKI). The unresolved issue in NDN is that any node in NDN can make a request by sending an interest packet. There is no mechanism to verify the identity of the senders. In addition, no framework for setting up a secure channel in NDN has been established.

In addition, there are privacy issues associated with NDN. According to [47], NDN with VANETs faces several privacy challenges:

- Name privacy: NDN names are human-readable and hierarchical that are suitable for some applications such as safety messages, traffic information, and other infotainment services. However, naming in NDN is in plain text and can be exploited.

- Content privacy: Content in NDN can be cached in the network. In-network caching can benefit VANETs, such as reducing traffic overhead when exchanging traffic information between vehicles. However, caching content allows any consumer to retrieve the content and identify the corresponding name.

- Cache privacy: A malicious node can infer its neighbors' cache hits from time information.

- Signature privacy: While integrity can be verified based on the digital signature, sensitive information can be revealed about the signer.

We are interested in studying these security and privacy issues and how we can deploy security mechanisms into routing algorithms and applications to improve the security of the system.

6.2.2 NDN-based Traffic Management Application

Related to the smart parking application, we are interested in applying vehicular named data networking to develop a traffic management application. Traffic is a matter that can affect people's lives, especially in crowded places. It can cause delays, put people under stress, and affect driving safety. It is also bad for the environment due to more fuel consumption. A traffic management system can monitor traffic conditions, detect traffic congestion, and provide route planning and suggestions. Providing real-time monitoring is challenging because it requires installation of equipment and communication infrastructure. Based on the information collected, we can predict the potential congestion and provide suggestions to drivers. We will discuss two functions in some detail.

1. Traffic forecast and predictions. Based on the monitored data, we can predict the congestion level of each road segment in real time. The congestion degree changes over time because of the the mobility nature of VANET. The degree of road congestion when suggesting an alternative path can be different from that when a vehicle arrives at those roads. So we have to consider the time factor. We need a real-time monitoring system that provides a continuous stream of relevant data, which helps the management system predict future trends.

2. Suggesting alternative paths. Suggesting alternative paths is especially helpful when an accident occurs on a specific road segment. It causes congestion of this road segment and other relevant roads. The traffic management system can suggest an

alternative path to the vehicles that are affected. A simple suggestion can make all vehicles that have the same destination to go the same path to the destination. This may cause the congestion on the suggested path. A more sophisticated suggestion can try to load balance among multiple alternative paths, even for those vehicles for the same destination. Another factor that needs to be considered is that drivers may not necessarily follow the exact same road as suggested. A probabilistic model about how drivers follow suggestions may further help make better suggestions.

In VANETs, one goal is to achieve a balanced distribution of traffic flow in the entire topology so that the overall traffic flow can be improved and congestion can be avoided. We plan to explore how a traffic management system with real-time monitoring information can balance the traffic flows among road segments. We believe vehicular named data networking is a promising architecture for supporting the traffic management system.

Bibliography

[1] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '09, pages 1–12, New York, NY, USA, 2009. ACM.

[2] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(7):26–36, July 2012.

[3] Yingdi Yu, Alexander Afanasyev, David Clark, kc claffy, Van Jacobson, and Lixia Zhang. Schematizing trust in named data networking. In *Proceedings of the 2nd ACM Conference on Information-Centric Networking*, ACM-ICN '15, page 177–186, New York, NY, USA, 2015. Association for Computing Machinery.

[4] Giulio Grassi, Davide Pesavento, Lucas Wang, Giovanni Pau, Rama Vuyyuru, Ryuji Wakikawa, and Lixia Zhang. Vehicular inter-networking via named data. *CoRR*, abs/1310.5980, 2013.

[5] I. Stojmenovic. Position-based routing in ad hoc networks. *IEEE Communications Magazine*, 40(7):128–134, July 2002.

[6] Christian Lochert, Martin Mauve, Holger Fü, and Hannes Hartenstein. Geographic routing in city scenarios. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(1):69–72, January 2005.

[7] Brad Karp and H. T. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, pages 243–254, New York, NY, USA, 2000. ACM.

[8] Kevin C. Lee, Pei-Chun Cheng, Jui-Ting Weng, Lung-Chih Tung, and Mario Gerla. Vclcr: A practical geographic routing protocol in urban scenarios. 2008.

[9] Z. Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *IEEE Communications Surveys Tutorials*, 8(1):24–37, First 2006.

[10] M. Nakamura, T. Kitani, W. Sun, N. Shibata, K. Yasumoto, and M. Ito. A method for improving data delivery efficiency in delay tolerant vanet with scheduled routes of

cars. In *2010 7th IEEE Consumer Communications and Networking Conference*, pages 1–5, Jan 2010.

[11] X. Yang, L. Liu, N. H. Vaidya, and F. Zhao. A vehicle-to-vehicle communication protocol for cooperative collision warning. In *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004.*, pages 114–123, Aug 2004.

[12] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, pages 1–11, April 2006.

[13] Moneeb Gohar, Naveed Khan, Awais Ahmad, Muhammad Najam-Ul-Islam, Shahzad Sarwar, and Seok-Joo Koh. Cluster-based device mobility management in named data networking for vehicular networks. *Mobile Information Systems*, 2018, 2018.

[14] W. Huang, T. Song, Y. Yang, and Y. Zhang. Cluster-based selective cooperative caching strategy in vehicular named data networking. In *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, pages 7–12, Aug 2018.

[15] C. Bian, T. Zhao, X. Li, and W. Yan. Boosting named data networking for efficient packet forwarding in urban vanet scenarios. In *The 21st IEEE International Workshop on Local and Metropolitan Area Networks*, pages 1–6, April 2015.

[16] Yu-Ting Yu, Yuanjie Li, Xingyu Ma, Wentao Shang, M. Y. Sanadidi, and M. Gerla. Scalable opportunistic vanet content routing with encounter information. In *2013 21st IEEE International Conference on Network Protocols (ICNP)*, pages 1–6, Oct 2013.

[17] H. Maryam, A. Wahid, and M. A. Shah. Mitigating broadcast storm in interest/data packet forwarding in vehicular content centric networking. In *2017 International Conference on Communication Technologies (ComTech)*, pages 162–167, April 2017.

[18] G. Grassi, D. Pesavento, G. Pau, L. Zhang, and S. Fdida. Navigo: Interest forwarding by geolocations in vehicular named data networking. In *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–10, June 2015.

[19] Z. Yan, S. Zeadally, and Y. Park. A novel vehicular information network architecture based on named data networking (ndn). *IEEE Internet of Things Journal*, 1(6):525–532, 2014.

[20] A. Kumar and N. Hariharan. Enhanced mobility based content centric routing in rpl for low power lossy networks in internet of vehicles. In *2020 3rd International Conference on Intelligent Autonomous Systems (ICoIAS)*, pages 88–92, 2020.

[21] Jérôme François, Thibault Cholez, and Thomas Engel. Ccn traffic optimization for iot. In *2013 Fourth International Conference on the Network of the Future (NoF)*, pages 1–5, 2013.

[22] M. Amadeo, C. Campolo, and A. Molinaro. Internet of things via named data networking: The support of push traffic. In *2014 International Conference and Workshop on the Network of the Future (NOF)*, pages 1–5, 2014.

[23] Marica Amadeo, Claudia Campolo, Antonella Molinaro, and Nathalie Mitton. Named data networking: A natural design for data collection in wireless sensor networks. In *2013 IFIP Wireless Days (WD)*, 2013 IFIP Wireless Days (WD), pages 1–6. IEEE, 2013.

[24] H. A. Khattak, S. U. Islam, I. U. Din, and M. Guizani. Integrating fog computing with vanets: A consumer perspective. *IEEE Communications Standards Magazine*, 3(1):19–25, 2019.

[25] Redowan Mahmud, Ramamohanarao Kotagiri, and Rajkumar Buyya. *Fog Computing: A Taxonomy, Survey and Future Directions*, pages 103–130. Springer Singapore, Singapore, 2018.

[26] W. Balzano and F. Vitale. Dig-park: A smart parking availability searching method using v2v/v2i and dgp-class problem. In *2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 698–703, 2017.

[27] Xueshi Hou, Yong Li, Min Chen, Di Wu, Depeng Jin, and Sheng Chen. Vehicular fog computing: A viewpoint of vehicles as the infrastructures. *IEEE transactions on vehicular technology*, 65(6):3860–3873, 2016.

[28] Y. Xiao and Chao Zhu. Vehicular fog computing: Vision and challenges. In *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 6–9, 2017.

[29] Jyoti Grover, Ashish Jain, Sunita Singhal, and Anju Yadav. Real-time vanet applications using fog computing. In Arun K. Somani, Sumit Srivastava, Ankit Mundra, and Sanyog Rawat, editors, *Proceedings of First International Conference on Smart System, Innovations and Computing*, pages 683–691, Singapore, 2018. Springer Singapore.

[30] Chaogang Tang, Xianglin Wei, Chunsheng Zhu, Wei Chen, and Joel J. P. C Rodrigues. Towards smart parking based on fog computing. *IEEE access*, 6:70172–70185, 2018.

[31] Yi Zhang, Chih-Yu Wang, and Hung-Yu Wei. Parking reservation auction for parked vehicle assistance in vehicular fog computing. *IEEE transactions on vehicular technology*, 68(4):3126–3139, 2019.

[32] Trista Lin, Herve Rivano, and Frederic Le Mouel. A survey of smart parking solutions. *IEEE transactions on intelligent transportation systems*, 18(12):3229–3253, 2017.

[33] Khaoula Hassoune, Wafaa Dachry, Fouad Moutaouakkil, and Hicham Medromi. Smart parking systems: A survey. In *2016 11th International Conference on Intelligent Systems: Theories and Applications (SITA)*, 2016 11th International Conference on Intelligent Systems: Theories and Applications (SITA), pages 1–6. IEEE, 2016.

[34] IEEE Electrical Insulation Society Staff Corporate Author. Smart parking solutions for urban areas. In *2013 IEEE 14th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2013 IEEE 14th International

Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), pages 1–6, [Place of publication not identified], 2013. IEEE.

[35] Faheem, S.A. Mahmud, G.M. Khan, M. Rahman, and H. Zafar. A survey of intelligent car parking system. *Journal of Applied Research and Technology*, 11(5):714 – 726, 2013.

[36] Thierry Delot, Nicolas Cenerario, Sergio Ilarri, and Sylvain Lecomte. A cooperative reservation protocol for parking spaces in vehicular ad hoc networks. 01 2009.

[37] N Doulamis, E Protopapadakis, and L Lambrinos. Improving service quality for parking lot users using intelligent parking reservation policies. pages 1392–1397. IEEE, 2013.

[38] ElMouatezbillah Karbab, Djamel Djenouri, Sahar Boulkaboul, and Antoine Bagula. Car park management with networked wireless sensors and active rfid. In *Electro/Information Technology (EIT), 2015 IEEE International Conference on*, 2015 IEEE International Conference on Electro/Information Technology (EIT), pages 373–378. IEEE, 2015.

[39] I. Moiseenko S. Mastorakis, A. Afanasyev and L. Zhang. ndnSIM 2: An updated NDN simulator for NS-3, ndn, technical report ndn-0028, revision 2, 2016, 2016.

[40] Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, 5(3&4):128–138, December 2012.

[41] James Bernsen and D. Manivannan. River: A reliable inter-vehicular routing protocol for vehicular ad hoc networks. *Computer Networks*, 56:3795–3807, 11 2012.

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

[43] D. D. Clark, D D Clark, and D L Tennenhouse. Architectural considerations for a new generation of protocols. *Computer communication review.*, 20(4):200–208, 1990.

[44] Kayhan Z. Ghafoor. Video streaming over vehicular ad hoc networks: A comparative study and future perspectives. *ARO-The Scientific Journal of Koya University*, 4(2):25–36, 2016.

[45] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, page 13–16, New York, NY, USA, 2012. Association for Computing Machinery.

[46] Ivan Stojmenovic, Sheng Wen, Xinyi Huang, and Hao Luan. An overview of fog computing and its security issues. *Concurrency and Computation: Practice and Experience*, 28(10):2991–3005, 2016.

[47] Rihab Boussada, Balkis Hamdane, Mohamed Elhoucine Elhdhili, Soumaya Argoubi, and Leïla Azouz Saïdane. A secure and privacy-preserving solution for iot over ndn applied to e-health. *2018 14th International Wireless Communications Mobile Computing Conference (IWCMC)*, pages 817–822, 2018.

VITA

Education

UNIVERSITY OF KENTUCKY, Collage of Engineering, Master's Degree of Computer Science, 2013, Lexington, USA

UNIVERSITY OF DAMMAM, Collage of Education, Bachelor of Science and Education in Computer Science, 2006, Jubail, KSA

Professional Positions Held

Yanbu University Collage: Yanbu, Saudi Arabia - Lecturer, Computer science, 2014-Present

Riyadh Alhoda School: Jubail, Saudi Arabia - Instructor, Computer science, 2008-2009

Aljyal Institute of training: Jubail, Saudi Arabia - Instructor, Computer science, 2006-2007

Bassma Aldahlan