ASPIRE: Adaptive Service Provider Infrastructure for VANETs

by

Agop Koulakezian

A thesis submitted in conformity with the requirements for the degree of Master of Applied Science (M.A.Sc.)
Graduate Department of Electrical and Computer Engineering
University of Toronto

Copyright © 2011 by Agop Koulakezian
Abstract

ASPIRE: Adaptive Service Provider Infrastructure for VANETs

Agop Koulakezian

Master of Applied Science (M.A.Sc.)

Graduate Department of Electrical and Computer Engineering

University of Toronto

2011

User desire for ubiquitous applications on-board a vehicle motivates the necessity for Network Mobility (NEMO) solutions for Vehicular Ad-Hoc Networks (VANETs). Due to the dynamic topology of VANETs, this approach incurs excessive infrastructure cost to maintain stable connectivity and support these applications. Our solution to this problem is focused on a novel NEMO-based Network Architecture where vehicles are the main network infrastructure. Within this Architecture, we present a Network Criticality-based clustering algorithm, which adapts to mobility changes to form stable self-organizing clusters of vehicles and dynamically builds on vehicle clusters to form more stable Mobile Networks. Simulation results show that the proposed method provides more stable clusters, lower handoffs and better connectivity compared to popular density-based vehicle clustering methods. In addition, they confirm the validity of the proposed Network Architecture. The proposed method is also robust to channel error and exhibits better performance when the heterogeneity of vehicles is exploited.
To my Loving Parents and Caring Brothers

Your motivation and support make everything possible
Acknowledgements

I would like to thank my Supervisor, Professor Alberto Leon-Garcia, for being an amazing guide and teacher for more than two years. In addition to listening to me for hundreds of hours and genuinely caring, he has provided me with constructive feedback and magnificent support. His belief in me and his valuable questions have helped me grow in confidence and become more independent in my research. I will forever be thankful for his supervision and contribution to my academic and personal life.

I would also like to thank my parents and cousin, Adour, for their continuous motivation and support and to my brothers who always gave their all so that I obtain the best education.

I am very grateful for Christine Shea who gave me a head start in NS2 by providing her implementation code for APROVE and Dr. Ali Tizghadam and Weiwei Li for their support in using the criticality metric.

I would also like to thank Amer Farroukh and Dr. Hadi Bannazadeh for their valuable ideas that accelerated acquiring new linux scripting skills, enabling me to run massive simulations in a short time.

In addition, I would like to thank Armin Ghayoori, Khashayar Khavari, Nadeem Abji and Houman Rastegarfar for their creative late-night research ideas and support for my thesis.

To my friends who constantly encouraged me, listened to me and helped me keep going, I thank you with all my heart.

Furthermore, I would like to say a special thank you to Elias Bou Harb, Saleem Khouri and Hillary Anne Armstrong, who have been the inspiration and rock I needed throughout my work on developing this thesis.

Lastly, I would like to thank my thesis defense committee for taking the time to review my work and providing me with valuable feedback and comments.
# Contents

1 Introduction to VANETs, NEMO and Clustering ........................................ 1
   1.1 Motivation .................................................................................. 1
   1.2 Vehicular Ad Hoc Networks ....................................................... 3
   1.3 Network Mobility ....................................................................... 5
   1.4 Clustering .................................................................................. 9
   1.5 Problem Statement ..................................................................... 9
   1.6 Objective .................................................................................. 11
   1.7 Contributions ............................................................................ 12
   1.8 Thesis Outline .......................................................................... 13

2 Background and Related Work ................................................................. 14
   2.1 VANET Network Architectures ................................................. 14
      2.1.1 Infrastructure-less Wireless Ad Hoc VANET Architecture .... 15
      2.1.2 Cellular Infrastructure-based VANET Architecture ............ 16
      2.1.3 RSU Infrastructure-based VANET Architecture ................. 17
      2.1.4 Brief Description of the ASPIRE Architecture ................. 18
   2.2 Survey of Clustering Algorithms ............................................... 20
      2.2.1 Lowest ID and MOBIC Clustering .................................... 20
      2.2.2 $k$-means Clustering ......................................................... 22
      2.2.3 Affinity Propagation Clustering ........................................ 23
2.2.4 Density-based and Mobility-based Clustering

2.2.5 Network Criticality-based Clustering

3 ASPIRE Network Architecture

3.1 Architecture Overview

3.1.1 Design Goals

3.1.2 Communication Scenarios

3.2 ASPIRE Architecture

3.2.1 ASPIRE VANET Model

3.2.2 ASPIRE Architecture Scenarios

3.2.3 Detailed ASPIRE Architecture Design

3.2.4 Protocol Layering in the ASPIRE Architecture

3.2.5 ASPIRE Physical Components

3.2.6 ASPIRE Architectural Components

3.3 ASPIRE Addressing and Routing

3.3.1 Assumptions and Basics

3.3.2 IP Packet Delivery in the ASPIRE Architecture

3.4 Chapter Summary

4 ASPIRE Clustering Algorithms

4.1 ASPIRE Clustering Requirements and Design

4.1.1 ASPIRE Clustering Requirements

4.1.2 Overview of the ASPIRE Clustering Design

4.2 Automatic Neighbour Relation Setup

4.2.1 ASPIRE Neighbour List

4.2.2 ASPIRE Messaging

4.2.3 Behavior of ASPIRE Messaging

4.3 Cluster Maintenance and Formation
4.3.1 Cluster Maintenance .............................................. 56
4.3.2 Cluster Formation ................................................. 60
4.3.3 Cluster Management ............................................... 63

5 ASPIRE Simulation Results ........................................... 65
  5.1 Simulation Setup .................................................... 66
    5.1.1 Traffic Simulator ............................................... 66
    5.1.2 Network Simulator ............................................. 66
  5.2 Performance Metrics and goals ................................... 67
  5.3 Mobility Performance with Vehicles ............................ 69
  5.4 Robustness Performance with Vehicles .......................... 75
  5.5 Mobility Performance with Vehicles and Trucks ............... 79
  5.6 Robustness Performance with Vehicles and Trucks ............ 84

6 Conclusion ..................................................................... 88
  6.1 Main Ideas ............................................................. 88
  6.2 Future Work ........................................................... 89

Bibliography .................................................................... 90
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>VANET network architecture alternatives</td>
<td>15</td>
</tr>
<tr>
<td>4.1</td>
<td>Neighbour entry for node $j$</td>
<td>51</td>
</tr>
</tbody>
</table>
# List of Figures

1.1 Intelligent Vehicle System Roadmap [1] ........................................ 2
1.3 Packet Encapsulation in NEMO [2] ................................................ 8

3.1 ASPIRE VANET Model ............................................................. 34
3.2 ASPIRE Network Architecture .................................................... 35
3.3 ASPIRE Protocol Stack ............................................................. 38
3.4 Truck and Vehicle Protocol Stacks .............................................. 39
3.5 Packet Delivery Entities ............................................................ 44
3.6 Packet Headers ........................................................................... 45

4.1 Clustering Algorithm State Diagram ........................................... 55
4.2 NEMO Overtaking ....................................................................... 58
4.3 Cluster Join ................................................................................ 61
4.4 NEMO Join ................................................................................ 62
4.5 Become Mobile Router .............................................................. 62
4.6 Cluster Management Algorithm Relationships ............................ 64

5.1 The average cluster size as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s. ............... 69
5.2 The average rate of clusterhead change as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and $Hel$ is swept over 1s and 2s. 

5.3 The average clusterhead duration as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and $Hel$ is swept over 1s and 2s. 

5.4 The average cluster member duration as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and $Hel$ is swept over 1s and 2s. 

5.5 The average mobile router duration as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and $Hel$ is swept over 1s and 2s. 

5.6 The average network connectivity as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and $Hel$ is swept over 1s and 2s. 

5.7 The average cluster size as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and $Hel$ is swept over 1s and 2s. 

5.8 The average rate of clusterhead change as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and $Hel$ is swept over 1s and 2s. 

5.9 The average clusterhead duration as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and $Hel$ is swept over 1s and 2s.
5.10 The average cluster member duration as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s. 78

5.11 The average mobile router duration as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s. 78

5.12 The average network connectivity as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s. 79

5.13 The average cluster size as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s. 80

5.14 The average rate of clusterhead change as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s. 81

5.15 The average clusterhead duration as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s. 82

5.16 The average cluster member duration as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s. 82
5.17 The average network connectivity as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s. .......................................................... 83

5.18 The average cluster size as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s. ........................................................................ 84

5.19 The average rate of clusterhead change as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s. ........................................................................ 85

5.20 The average clusterhead duration as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s. ........................................................................ 85

5.21 The average cluster member duration as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s. ........................................................................ 86

5.22 The average network connectivity as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s. ........................................................................ 87
Chapter 1

Introduction to VANETs, NEMO and Clustering

1.1 Motivation

Every person spends on average a couple of hours every day as a driver or a passenger in a vehicle in North America. In particular, Torontonians spend an alarming 80 minutes commuting to work on average every day [3]. The movement of these vehicles is spatially restricted and not hard to estimate, especially on highways, considering the fact that many of them maintain relatively constant speed and distance from a group of surrounding vehicles, and stay on a highway, such as the 401 highway, for at least 30 miles. Moreover, there are approximately 1.3 million trucks and 10.8 million vehicles registered in Ontario and a large portion of these trucks are actively operating on the highways [4]. For these reasons, research in Vehicular Ad Hoc Networks (VANETs) has attracted the attention of both the industry and academia. Vehicles are expected to have GPS and Wi-Fi-like capabilities, and to be able to communicate together to provide road safety, traffic management, and infotainment services for drivers and passengers. In [1], Hahn predicts the roadmap of Intelligent Vehicular systems as seen in Fig. 1.1. While the ini-
tial focus is on developing mechanisms that aid the driver, such as collision warning and automatic breaking, more mature technologies depend on vehicles co-operating with their neighbours by exchanging driver intentions and mobility parameters, ultimately aiming towards autonomous driving. However, recent research in Vehicular Networks has failed to take advantage of the mobility of vehicles that can be monitored and estimated and to support co-operation between them in order to provide services for passengers and drivers with a low infrastructure cost. The following story highlights the issues of this problem in further detail.

Almost eighteen months ago, I was a passenger in the minivan of my cousin Adour, heading to a picnic location, along with his wife, Rouzane, and their two sons, Krikor and Arad. A few hundred meters in front of us was another family in a car that we were following to the destination, and Rouzane was texting them, making sure we did not lose them on the 401 because of the tens of trucks within our view. While Adour was silencing everyone, trying to listen to the traffic report on the radio and to get informed about the accident ahead on the highway, his nine year-old son, Krikor, was asking me why he was not able to play games over wireless (with his friend in the other car) on his Nintendo DS-I as he was playing when they were at home. In addition, his six year-old brother Arad, was asking me about the location of the closest Swiss Chalet. As I did not have a data plan on my smartphone to set it up as an Access Point (AP) for Krikor
to play online (he would still not been able to play with his friend, but he might have been able to play with other people over the Internet), or to look for the closest Swiss Chalet location for Arad, I could not answer these questions. But the more difficult and interesting questions were:

1. Why did Rouzane’s text, destined to a car moving on the same route and less than a hundred meters away, have to go through costly fixed infrastructure?

2. Why did Adour have to seek the news about an accident on the road ahead, rather than receive it immediately from the vehicles between him and the accident?

3. Why was Krikor not able to play his favorite game online or with his friend in the other car?

4. Why would I have to lookup for information, such as the location of a restaurant, if someone else in the group of vehicles that had been moving with us for a while on the highway, might have already looked it up?

At this point, I realized that I had found an interesting, yet challenging problem for my thesis: How can we provide the services desired by these four passengers in the car together, and for a low cost? This thesis provides a solution for this problem, but it is not only about families heading to a picnic destination. The basis of this work lies on two future technological visions: Vehicular Ad Hoc NETworks (VANETs) and Mobile Networks (NEMOs), which are discussed in the following sections.

1.2 Vehicular Ad Hoc Networks

With the continuous development of micro-electronic and wireless communication technologies, vehicles are now becoming computers on wheels by being equipped with intelligent electronic devices called wireless On Board Units (OBUs). Each OBU includes
a computer processor, a Global Positioning System (GPS), sensing devices, storage devices and a wireless transceiver, providing ad hoc network connectivity for the vehicles. Equipped with OBU, vehicles can now communicate with other vehicles while moving on the roads or with fixed roadside infrastructures when passing by them. These communication methods are referred to as Vehicle-to-Vehicle (V2V) communication and Vehicle-to-Infrastructure (V2I) communication and basically form the Vehicular Ad Hoc Network (VANET). The fixed roadside infrastructures or Roadside Units (RSUs), are futuristic dedicated infrastructure elements, similar to base stations, which would be connected to the Internet Backbone through wired or wireless connections. These RSUs would most likely be located at critical points of the road, such as intersections or construction sites, and are intended for providing message dissemination, security and network stability for vehicular networks.

For this purpose, 75 MHz of spectrum in the 5.9 GHz band has been allocated in North America for the Dedicated Short Range Communications (DSRC) standard [5], a set of communication protocols and standards specifically designed for vehicular communications. This channel is a means through which real-time multimedia applications, including peer-to-peer (P2P) content provisioning, can be deployed in the future and various services can be provided. A similar band has been allocated in Europe and Japan.

Vehicular ad hoc Networks are one type of ad hoc network, but significantly differ from other wireless ad hoc networks, such as sensor networks, mobile ad hoc networks, etc. VANETs are infrastructure-based; thus, they can provide reliable communication services. However, the communication links in such networks have short connection times due to high mobility of vehicles. For example, two vehicles moving in the same direction and starting at the same position can only maintain a reliable link for 25 seconds between them if one of them is faster by 10m/s and the communication range is 250m. However, since the movement of vehicles is spatially restricted and the vehicles are
spatially dependent on each other in movement, link failure can be sometimes predicted and alternative multi-hop routes can help ensure communication between vehicles for longer periods.

Applications for VANETs include safety, traffic management, commercial ad dissemination, driver assistance, web surfing, voice, gaming and infotainment applications. A vehicle may contain multiple Application Units (AUs), which are built-in or portable communication devices running a variety of these applications. These applications have different delay, reliability, security, and bandwidth requirements. Due to the mere size that a VANET might grow into, with the millions of vehicles in each country, and due to the diversity of non-safety vehicular applications desired by various users, researchers have identified the necessity for a new mobility mechanism, defined as Network Mobility, to match these needs.

1.3 Network Mobility

Mobility has changed people’s lives, especially the way they communicate. As Internet access becomes more ubiquitous every day, user demand for mobility has exceeded single terminals, to include the notion of Network Mobility, where a group of users moving together change their point of attachment to the Internet. Building on concepts of Mobile IP (MIP) [6], NEtwork MObility (NEMO) [7] allows a mobile node, called the Mobile Router (MR), to act as a router for a set of nodes moving together as one entity. This group of nodes altogether is referred to as a Mobile Network or alternatively a NEtwork-that-MOves or NEMO. The NEMO is assigned to a particular network, called its Home Network, where it is located when it is not moving, as shown in Fig. 1.2. All the nodes of the NEMO have configured addresses belonging to one or more address blocks, called Mobile Network prefixes (MNPs), assigned to the home network. As the NEMO moves away from home, it maintains these configured addresses; thus, the packets addressed
to the nodes in the NEMO, called Mobile Network Nodes (MNNs) are still routed to its home network. Moreover, the MR acquires an address from the visited network, called the care-of Address (CoA), similar to MIP [2].

Mobile Network Nodes (MNNs) are regular MIP nodes in the NEMO and access the internet through a MR. Each NEMO typically has one MR, but may also have several MRs if needed. A Mobile Network Node can be: a Local Fixed Node (LFN), which is a fixed node belonging to the NEMO with no mobility support, a Local Mobile Node (LMN), which is a mobile node whose home link belongs to the NEMO, or a Visiting Mobile Node (VMN), which is a mobile node whose home link does not belong to the NEMO. A MNN has a Home Agent (HA) with a Home Address (HoA), similar to MIP, and does not get a new IP address as long it is part of the same NEMO. The MR performs handoff on behalf of these moving nodes, keeping them connected to the Internet and
reducing their handoff latency [2].

Fig. 1.2 shows how an IP datagram is exchanged between a MNN A and a node located in the Internet, called a Correspondent Node (CN). Since the datagram carries the IP address of MNN A, which belongs to the MNP of its NEMO, the datagram is routed to the Home Network of the NEMO. The HA of the MR located in the Home Network encapsulates this datagram into a new datagram sent to the CoA of the MR and sets the source address with its own IP address, as seen in Fig. 1.3. Note that both the CN and the MNN are unaware of the mobility of the NEMO because of this encapsulation. When the MR receives this datagram, it removes the outer IP header and delivers the original datagram to MNN A. The operation in the opposite direction is analogous, as seen in Fig. 1.3. In order to achieve this IP datagram exchange, a MR creates a bi-directional tunnel between itself and a NEMO-compliant HA, by sending a successful Binding Update to it, informing it of its current point of attachment, whenever it changes its CoA. Therefore, the MR sends one binding update on behalf of all the MNNs in its NEMO and the MNNs are not even aware of the changes in the routes of their datagrams.

Protocol extensions to MIP are used to enable support for NEMO. The extensions are backward-compatible with existing MIP functionality. A NEMO-compliant HA can operate as a MIP HA. Some of these extensions involve the dynamic home agent address discovery (DHAAD) mechanism from MIP [6] that allows a mobile node to discover the address of the HA on its home link. The MR sends Internet Control Message Protocol (ICMP) home agent address discovery requests to the anycast address of MIP HAs for the MNP. A new flag (R) is introduced in the DHAAD request message, indicating the desire to discover HAs that support MRs. This flag is added to the DHAAD reply message as well. On receiving the home agent address discovery reply message, the MR discovers the HAs operating on the home link and attempts home registration to each of the HAs until its registration is accepted.
A point-of-attachment, typically an Access Router (AR), sends out standard IPv6 Router Advertisement messages, including a MNP (see Fig. 1.2). When a MR moves into the communication range of a new AR, it uses the MNP to get its new IP address configuration and sends the binding update message to its HA, while MNNs keep their IP addresses and ongoing sessions. The HA sets up forwarding for all MNPs owned by the MR after receiving a Binding Update from the MR with the MR Flag (R) set; thus if a packet with a destination address belonging to the MNP is received, the HA tunnels the packet to the MR. This tunneling is done by using IP encapsulation (see Fig. 1.3).

A moving vehicle can be seen as a network that changes; as one entity, its attachment point to the Internet and its reachability in the infrastructure network [8]. Therefore, NEMO can be used to provide connectivity to the AUs in the vehicle, where an onboard unit (OBU) in the vehicle (Mobile Network) can act as a MR that performs the handoff on behalf of the AUs (MNNs) in the vehicle [8, 9, 10]. Therefore lately, there has been a lot of research in NEMO solutions for VANETs. On the other hand, another interesting research topic in VANETs has been the issue of clustering, discussed in the next section.
1.4 Clustering

The notion of clustering has existed for a long time. Clustering is defined as the method of separating data into groups of similar objects. In each group, called a cluster, group members share a similarity criterion with one another and are different from objects in other groups. The definition of the similarity criterion is application dependent and could be based on geographic location, desired content, available resources, reputation, or network contribution, among others. Generally, the main goal of clustering is to achieve simplicity in terms of data representation and allow co-operation. In wireless networks, clustering leads to efficient management of nodes in terms of addressing scheme, routing and load balancing.

Particularly, in the intense mobile environment of VANETs, where the connections of communication links are short lived, clustering of vehicles into groups of similar mobility helps reduce the relative mobility between communicating neighbouring nodes, consequently leading to intra-cluster stability. This simplifies the problem of managing a large number of nodes into a problem of managing a small number of clusters, where clusters manage themselves and rarely change. Moreover, it allows cluster members to co-operate, helping each other make better decisions, sharing available information and using scarce network resources, such as bandwidth, more efficiently. For this purpose, several researchers [11, 12, 13] have recently developed distributed clustering algorithms for VANETs.

1.5 Problem Statement

Due to the continuous increase of user demand for mobility and ubiquitous Internet access in the past few years, researchers have realized that non-safety applications are the marketing catalyst to push VANET research forward and make VANETs a technological reality. Moreover, since MNNs are regular MIP nodes and can run various types of
Internet (non-safety) applications on top of the vehicular network once connected, NEMO provides great flexibility for vehicular communication and has become a requirement for non-safety applications in vehicular communication standards [14]. Thus, much of the focus has been on NEMO solutions for VANETs.

Being designed for mobile networks having single-hop connectivity to a network infrastructure, such as trains, NEMO alone cannot provide connectivity over multi-hop, intermittent access to the network infrastructure, unless coupled with a VANET routing protocol [7, 8]. For this reason, recent vehicular network architectures depend on both Network Mobility for providing session continuity and reachability for a mobile network and on VANET routing for handling communication between vehicles and RSUs [9]. These designs require each vehicle to contain a NEMO-capable MR device (based on [7]) and RSUs to be installed at least every two miles to maintain network connectivity. However, the excessive cost of installing and maintaining such RSUs and the need to install NEMO-capable devices in each car cast a shadow on the feasibility, scalability and widespread deployment of VANETs. Therefore, there is a need for a new vehicular network architecture which reduces the dependency on dedicated fixed infrastructure, yet ensures high network connectivity and supports non-safety applications for vehicles.

Moreover, VANET-NEMO architectures have failed to take advantage of the estimable mobility, co-operation capability, and heterogeneity of vehicles. While vehicular clustering in the literature has only addressed the co-operation of vehicles among these properties, the solutions have been limited to infrastructure-less algorithms and have failed to identify ways to integrate with NEMO-based infrastructure required for global network reachability and providing non-safety applications. Therefore, there is also a need to bridge the gap between these two seemingly conflicting research paradigms, namely centralized NEMO architectures for VANETS and infrastructure-less distributed clustering algorithms, taking the advantages of both.
1.6 Objective

In this work, the objective is to design a novel NEMO-based Network Architecture for VANETs that does not depend on dedicated fixed infrastructure, yet ensures high network connectivity and provides non-safety services for vehicles on a highway. Therefore, we aim to rely more on vehicles as network infrastructure and develop clustering algorithms that support co-operation between them and exploit their estimable mobility patterns on a highway.

We have the view of a Service Provider (SP) who does not set up dedicated RSUs on a highway, but rather combines 3G/LTE cellular communication and clustering of vehicles with local communication between them for providing most of the network services. The 3G/LTE provides connectivity and global reachability to vehicle clusters. The SP also exploits the heterogeneity of vehicles by using trucks, which have higher processing, communication, and bandwidth capabilities than regular vehicles, to create moving infrastructure that improves the robustness of the vehicular network. If vehicles can take on many of the responsibilities of fixed infrastructure, we end up with a design that reduces major communications costs by exploiting elements that already exist on our roads. The key for the user is to remain connected and receive the services almost ubiquitously. Therefore, we aim to design a robust network with high network connectivity.

The desired network architecture requires that we design a robust and adaptive VANET clustering algorithm. The greater dependence on vehicles, and less dependence on fixed infrastructure makes it critical that the clustering algorithm performs accurate estimations for the mobility of vehicles in the network and forms clusters lasting for long periods. The clustering algorithm should also exploit the heterogeneity of vehicles in order to make vehicles with higher capabilities or better connectivity play more important roles as network infrastructure. In addition, the distributed clustering algorithm should take advantage of the richer centralized information from the network architecture to make intelligent clustering decisions. Using the clustering algorithms, our main aim is
for the network architecture achieve large cluster sizes and high network connectivity. We measure network connectivity by calculating the proportion of time a node is connected to a Mobile Router in the network; thus to the Internet. Comparing the performance of the architecture using various applications is out of the scope of this thesis.

1.7 Contributions

Our contributions can be summarized in the following points:

1. We have designed the Adaptive Service Provider InfrastructuRE (ASPIRE) for VANETs, the first network architecture for highway VANETs that reduces infrastructure costs by using vehicles as the main infrastructure in the network.

2. We have presented the view of a novel Service Provider, that adaptively builds on moving infrastructure, using the connectivity of a regular ISP only when the required information is not already cached in its adaptive network.

3. Building on the latest research in Network Mobility, we have considered vehicles, as being Mobile Network Nodes, rather than Mobile Routers in the network for the first time, and provided simulation results to verify the plausibility of this novel view.

4. We have developed a clustering algorithm that not only creates stable long-lasting clusters, but also leads to the distributed grouping of these clusters by forming adaptive Mobile Networks in the dynamic environment of VANETs.

5. We have built on a measure of network robustness from the field of network science to develop a clustering metric that outperforms the commonly used Density metric.

6. By combining the advantages of beaconing and predicting vehicle mobility, we have developed intelligent maintenance algorithms to achieve high cluster stability, high
network connectivity and robustness to channel error.

7. By exploiting the heterogeneity of vehicles and building on stable vehicles such as trucks as moving infrastructure, we have enhanced both the stability of the network with diverse mobility conditions and the robustness of the network under lossy channel conditions.

1.8 Thesis Outline

The rest of this thesis is organized as follows. In Chapter 2, we discuss the background and present a literature review of both VANET network architectures and clustering algorithms. In Chapter 3, we present the ASPIRE architecture, identifying the design goals, the communication scenarios addressed and explain how the proposed architecture fits the design requirements. In Chapter 4, we present the ASPIRE clustering algorithms, namely the beaconing, maintenance and cluster formation algorithms. Chapter 5 presents the simulation results of the performance tests using the NS-2 simulator. Finally, in Chapter 6, we provide concluding remarks and outline possible extensions to our work.
Chapter 2

Background and Related Work

This Chapter discusses the background and related work for both the ASPIRE network architecture and ASPIRE clustering algorithm.

2.1 VANET Network Architectures

In this section, we discuss several proposed network architectures presented in the literature for future vehicular networks for providing both safety and non-safety applications. Table 2.1 shows a summary of these designs, along with the type of infrastructure used by each, the academic papers that describe them and how they match the various architectural needs. This comparison allows us to identify the advantages and disadvantages of these designs, and select specific aspects from them in order to design the ASPIRE system, shown in the last row of Table 2.1. Throughout this section, we discuss each of these designs in detail and identify their advantages and shortcomings. We also briefly describe the ASPIRE design, highlighting how it is different from these designs.
Chapter 2. Background and Related Work

Table 2.1: VANET network architecture alternatives

<table>
<thead>
<tr>
<th>Type</th>
<th>Infrastructure</th>
<th>Paper</th>
<th>Low Cost</th>
<th>Low Handoff</th>
<th>Low Overhead</th>
<th>High Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wi-fi-based</td>
<td>None</td>
<td>[15]</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>3G-based</td>
<td>Existing</td>
<td>[16, 10]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSU-based</td>
<td>Existing and New</td>
<td>[9, 8]</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>ASPIRE</td>
<td>Existing</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

2.1.1 Infrastructure-less Wireless Ad Hoc VANET Architecture

The first design in Table 2.1 is the Infrastructure-less Wireless Ad Hoc VANET Architecture, or Wi-fi-based for short. In [15], Jetcheva et al propose such a system, which they call Ad Hoc City. The backbone network in this system is a mobile multi-hop network, composed of wireless devices mounted on mobile fleets such as city buses or delivery vehicles. To meet the requirements of this infrastructure-less design, Jetcheva et al have developed a routing protocol called C-DSR, which is an extension to the Dynamic Source Routing Protocol (DSR) [17].

Within this network architecture, base stations serve the mobile nodes within a topological cell and mobile nodes serve a varying set of personal mobile nodes (that do not assist in routing) within range. In order to find a route between a source and destination, C-DSR forwards the request from the source node to the source base station (through mobile nodes), which in turn forwards it to the destination base station and then to the destination node. This leads to a great overhead in setting up or changing routes and is a burden on the base stations, rather than the mobile nodes. The simulation results show that 10% of the nodes are more than six hops away from a base station, making
the path length between some of them longer than 12 wireless hops. Thus, in order to provide a good performance, the average overhead per node is extremely high, as visible in the results of [15].

The advantages of this architecture are the low cost of the design and its simplicity, along with the high wireless throughput provided by the abundance of mobile nodes. However, the design has high dependence on the density of mobile vehicles (that assist in routing), and on the resources of the base stations used. In addition, the movement of either a personal mobile node out of the range of its serving mobile node, or that of a mobile node out of the range of its serving base station, both lead to high handoff while re-establishing paths and restoring the sessions in the network. Moreover, the design has to deal with congestion and interference issues, and has a large overhead in setting up, maintaining and changing routes between node pairs.

2.1.2 Cellular Infrastructure-based VANET Architecture

The second design in Table 2.1 is the Cellular Infrastructure-based VANET Architecture, or 3G-based for short. In [16], Pack et al propose such a system, which they call the Mobile Hotspot system. The backbone network in this system is a set of base stations that provide Wireless Wide Area Network (WWAN) connectivity to mobile hotspots (vehicles), in terms of third generation (3G) cellular systems for example. Each vehicle is a NEMO with a MR (AP), which in turn provides wireless connectivity to the various MNNs within the vehicle (NEMO). Packets sent from a CN to a MNN are first routed to the base station serving the MR of the MNN, and subsequently routed to the MR and then to the MNN [16].

The advantages of this architecture are the simplicity of the design and the low handoff incurred with vehicle motion, due to the fact that the 3G connectivity is always available. Moreover, this design does not depend on the density of the vehicles, since all routing is done in one-hop between a base station and a vehicle. However, this leads to additional
overhead (extra encapsulation) and redundant messaging by vehicles to base stations for information that might be locally available in other neighbouring vehicles. This in turn results in unnecessary and excessive use of 3G connectivity, which is expensive and limited. Moreover, the fact that all messages have to be routed through the base stations (even if the source and destination are in different vehicles, a few meters away) decreases the effective throughput of the system and increases its cost.

In [10], Tsukada et al present a similar architecture, along with the additional feature of supporting multihop routing through vehicles, rather than only through base stations (vehicles do not have to be one hop away from a base station to be connected). This system is far more complex since it includes a Policy routing mechanism to choose between VANET (through vehicles) and NEMO (through base station) routes. Moreover, the system is still very costly due to its reliance on 3G base stations and the overhead of setting up, maintaining and changing routes between node pairs. Furthermore, this design requires each vehicle to contain a NEMO-capable MR device, which is expected to be a deterrent for many drivers. This consequently decreases the density of vehicles with MR devices on the road, which makes the policy routing mechanism virtually useless.

### 2.1.3 RSU Infrastructure-based VANET Architecture

The third design in Table 2.1 is the RSU Infrastructure-based VANET Architecture, or RSU-based for short. In [8, 9], Baldessari et al propose such a system, which includes dedicated fixed base stations, namely RSUs as infrastructure. Each vehicle is a NEMO (has a separate OBU acting as a MR) and includes two separate protocol stacks: one for VANET routing and another for NEMO routing. While Network Mobility is used for providing session continuity and reachability for the NEMOs, VANET routing is used for handling communication between vehicles and RSUs. The VANET routing protocol is coupled with a geographic routing mechanism, which helps the OBUs in the vehicles to potentially act as relays without the IP (NEMO) layer. Moreover, this
VANET sub-IP layer presents Ethernet-like characteristics so that standard mechanisms for datagram transport can be used [9]. This architecture is explained further in the GeoNet Architecture Design document [18].

The advantages of this architecture are the low handoff incurred with vehicle motion, due to the fact that connectivity to an RSU is always available. In addition, the idea of layering the different routing mechanisms is promising as it allows for immediate adaptation of data routes to topology changes [9]. Moreover, this design does not depend on the density of the vehicles, since all routing is done in one-hop between an RSU and a vehicle. However, just like the 3G-based design above, this design has additional overhead, uses redundant messaging by vehicles to RSUs for information that might be locally available in other neighbouring vehicles, and leads to unnecessary and excessive use of RSU connectivity, which is expensive and limited.

The idea of RSUs comes from VANET architectures for safety-applications, where high bandwidth is not a requirement. However, to guarantee providing services with various bandwidth requirements for non-safety applications, RSUs have to be installed every few miles on a highway, which may result in high congestion/interference around RSUs, increase costs significantly and even make them a single point of failure.

2.1.4 Brief Description of the ASPIRE Architecture

After analyzing the advantages and shortcomings of each of the designs mentioned above, we selected specific aspects of previous approaches to design the ASPIRE system. In this section, we identify the features taken from these designs, and how the problems associated with these features are addressed, leading to the ASPIRE design, which has low cost, low handoff, low overhead and high throughput.

ASPIRE adopts the idea of serving personal mobile nodes through other mobile nodes, which reduces cost and increases effective network throughput, and is used in the Wi-fi-based design. However, ASPIRE resolves the problems of handoff and overhead in this
design by limiting the path length between all source-destination pairs. This is made possible by building a stable hierarchical structure through clustering, in addition to keeping every node at most two hops away from a MR and maintaining a distance of at most four hops between any two consecutive MRs.

ASPIRE employs a routing method, similar to that of Policy routing used in the 3G-based design. However, by building a backbone network based on vehicles and viewing each vehicle as a MNN, rather than a MR, ASPIRE shifts the dependence from 3G/LTE base stations to free stable vehicles. Moreover, ASPIRE supports caching of information within the clusters and NEMOs, which reduces the unnecessary use of 3G/LTE connectivity.

ASPIRE adopts the idea of layering in the routing process, used in the RSU-based design, as it leads to fast adaptation to changes in the network topology. However, ASPIRE does not employ expensive RSUs, but rather relies on clustering in vehicles, along with multi-hop VANET routing and 3G/LTE connectivity for providing session continuity and reachability for the NEMOs. In addition, ASPIRE exploits the heterogeneity of vehicles and employs an adaptive clustering algorithm for building a stable vehicular network.

As a result, ASPIRE allows vehicles with regular MIP nodes to connect to its network, decreasing user cost and increasing the density of vehicles likely to join the VANET. Moreover, it reduces the cost of the SP by removing the excessive cost of installing and maintaining RSUs and using the connectivity of existing cellular bases stations instead. By supporting caching and local communication through clustering between vehicles, use of the 3G/LTE connectivity is reduced. Moreover, clustering enables selecting the best connected vehicles to act as moving infrastructure in the network, extending the connectivity of network nodes and improving the network robustness. The ASPIRE network architecture design is presented in detail in Chapter 3.
Chapter 2. Background and Related Work

2.2 Survey of Clustering Algorithms

In networks, clustering is the process of partitioning networks of nodes into organized groups. These groups, called clusters, are sub-networks in the overall network, thus form a hierarchical topology. Each node in a cluster is either identified as a Clusterhead (CH) or a Cluster Member (CM). On the one hand, a CH is an elected node that acts as the local controller for the cluster. It may have various responsibilities, ranging from routing, relaying of inter-cluster traffic from CMs, scheduling of intra-cluster traffic, and assigning channels for CMs. On the other hand, a CM is a regular node that belongs to a cluster. It does not necessarily participate in routing and is not involved in inter-cluster communication. Note that clustering can either be done by partitioning the network into groups and then choosing a CH for each, or by choosing nodes to serve as CHs and then all nodes that join to a certain CH would be part of the same cluster. Both of these methods are common ways of clustering.

Clustering can help reduce congestion, increase the stability of a dynamic network and reduce packet flooding. It can also reduce the relative mobility within a cluster, leading to more constant routes. Moreover, clustering can reduce the impact of node churn, namely the joining and leaving of nodes, which modifies the network topology. Since ASPIRE requires a robust and adaptive clustering algorithm, we have developed a novel VANET clustering algorithm based on the Network Criticality metric [19]. In this section, we survey various popular clustering and vehicular clustering algorithms in the literature and introduce the Network Criticality-based clustering method.

2.2.1 Lowest ID and MOBIC Clustering

A popular distributed clustering algorithm is the Lowest-ID algorithm [20]. In Lowest-ID, every node is assigned a unique ID and the node with the lowest ID, among its neighbours within a two-hop range, is elected as CH. The algorithm works in the following manner:
1. Nodes periodically broadcast their unique IDs, along with the ones of their neighbours.

2. Nodes that have the lowest IDs among their surrounding neighbours elect themselves as CHs.

3. Nodes that receive a broadcast message from a CH with an ID lower than its own ID identify the sender as their CH and become CMs.

4. When CHs become CMs (of other clusters), their original CMs restart the election process, removing them from possible CH options.

This clustering scheme is simple, but not suitable for scenarios where nodes are mobile due to the high frequency of re-clustering involved in maintaining the lowest-ID status of all CHs. An extension to Lowest-ID is MOBIC [21], which considers mobility to improve cluster stability. Assuming that the distance between two moving nodes is proportional to the power received from one to the other, MOBIC approximates the relative mobility of the nodes based on the ratios of the power received in two consecutive HELLO messages sent to each other. The aggregate relative mobility of a node is calculated by finding the variance of the relative mobility values between the node and all of its neighbours [21]. MOBIC works in the following manner:

1. Nodes begin in UNDECIDED state.

2. Nodes periodically broadcast their aggregate relative mobility (M) to their neighbours.

3. Among a set of nodes in UNDECIDED state, the node with the lowest M declares itself as CH and broadcasts its decision.

4. Nodes that receive a broadcast message from a CH identify the sender as their CH and become CMs.
5. Re-clustering is only needed when two CHs come within range of one another for more than the Cluster Contention Interval (CCI) time period.

A major disadvantage of MOBIC is the fact that an elected CH can remain as a CH for a long time after being elected, even though its clustering metric is now worse than that of its neighbours. This is because MOBIC fails to consider mobility during cluster maintenance, which makes a CH lose its CH status only when meeting another CH which has a higher M value. Thus, after running the algorithm for some time, the current CHs might be bad leaders, yet cannot be forced out of their roles. This problem is caused by the Fix-after-break approach that is used in MOBIC to reconstruct broken links and clusters, which may lead to harmful abrupt changes to the stability of the network. By monitoring the quality of links and the behavior of nodes during cluster maintenance as in [22], broken links can sometimes be predicted before they occur and taking appropriate actions accordingly can prevent harmful changes to network stability, as we shall see later.

### 2.2.2 k-means Clustering

Another popular clustering method is k-means clustering [23], which is used for data clustering. Its objective is to divide a set of data into clusters by minimizing the mean-squared error of relative intra-cluster distances. Analytically, it can be represented in the following way. Given a set of \( n \) data points, \( \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \), where each data point is a \( d \)-dimensional real vector, the aim of k-means is to partition the \( n \) points into \( k \) clusters, \( \{X_1, X_2, \ldots, X_k\} \), with \( k < n \) (and usually \( k << n \)), while minimizing the following sum:

\[
\sum_{i=1}^{k} \sum_{\mathbf{x}_j \in X_i} ||\mathbf{x}_j - \mu_i||^2
\]  

(2.1)

In (2.1), \( \mu_i \) is the \( d \)-dimensional CH vector of points in \( X_i \) and \( ||.|| \) is the Euclidean distance.
The \( k \)-means clustering algorithm works as follows:

1. Place \( K \) centres among the data points to be clustered
2. Assign each point to the nearest centre (partition into clusters)
3. Find the centroid of each cluster, which are now the new centres
4. Re-assign the data points to the nearest of the centres
5. Repeat 1-4 until the centres are stable

The \( k \)-means clustering method is very simple to implement, but does not always find the global optimum to the objective function. Since it is very sensitive to the initial selection of CHs, the algorithm has to be run several times with different initial CHs and the best result should be chosen. Therefore, this algorithm is not scalable due to the large number of necessary repetitions. Moreover, the \( k \)-means clustering method requires the number of clusters \( k \) to be set a priori to running the algorithm. Thus, the \( k \)-means clustering method cannot be adopted for mobile networks where the number of clusters would most likely be changing with the changes of the network topology, which is the case in VANETs.

2.2.3 Affinity Propagation Clustering

The Affinity Propagation Algorithm used for VANET clustering in [12] is a clustering method based on the idea of data clustering from \( k \)-means clustering and the idea of cluster contention from MOBIC. In affinity propagation, data points pass messages to one another, indicating the current affinity that one data point has for choosing another data point as its exemplar (CH). The similarity \( s(i,j) \) represents how well suited node \( j \) is to be the exemplar (CH) of node \( i \). This algorithm aims to maximize \( s(i,j) \) for every node \( i \) and its chosen exemplar \( j \). The affinity messages, used to calculate the similarity
values, include both *availability* $a(i,j)$ messages and *responsibility* $r(i,j)$ messages that indicate how well suited nodes are to become CHs and CMs for each other, respectively. Upon convergence, the CH of node (i) is:

$$CH_i = \arg \max_j \{a(i,j) + r(i,j)\} \tag{2.2}$$

The similarity function used in [12] is a combination of the Euclidean distance between node positions now and their respective (estimated) positions in the future. Therefore, every node should calculate both the responsibility and availability values for each of its neighbours based on the similarity function and periodically transfer all the information to every neighbour. Moreover, these values need to be damped with older values to avoid numerical oscillations that will prevent the algorithm from converging [12]. In the VANET environment, this corresponds to an attempt to create some kind of robustness metric, which adapts to the changes in the network topology while keeping *Memory* for all calculated values in the algorithm. This is done by making the new values equivalent to the damped weighted average of both the old and new values.

However, since Affinity Propagation is itself a *Memory-less* metric (might change completely between successive iterations), Affinity Propagation clustering may get negatively affected by the memory stored, which may lead to delayed responses to abrupt network topology changes. An example of this is when a CH $c$ has *good* affinity values as it had good links with its neighbours for some time, and abrupt changes to the network, such as a part of the cluster being disconnected, now make node $c$ a *bad* node in the network. In this case, Affinity Propagation might not be able to capture the changes in the network fast enough through its calculations, as they are still partly effected by the older *good* affinity values. Therefore, Affinity Propagation cannot provide high network stability when dealing with diverse vehicle mobility conditions, where connections are short-lived.
2.2.4 Density-based and Mobility-based Clustering

In [11], Kuklinski et al propose a clustering method based on density of connection graph, link quality and traffic conditions. The first step of this clustering method is Connectivity Level Estimation, which is simply measuring the number of active links to calculate the node density. The second step involves Link Quality Estimation by predicting future distances, as in affinity propagation clustering. The last step includes time averaging of subsequent received values from HELLO messages broadcast by the nodes in the network. Implementation steps of this algorithm are very similar to that of the affinity propagation algorithm. The problem of abrupt changes in clustering values is less severe compared to the affinity propagation algorithm, as the density clustering metric is less likely to change compared to the affinity propagation values. However, density-based clustering is still susceptible to abrupt and late clustering changes due to its dependence on time averaging of density values. This problem would arise when the communication links of a node with a large number of its neighbours are broken in a short period of time [11].

Gopalaswamy proposes a similar clustering method in [13], based on relative mobility rather than density. Its implementation is similar to that of MOBIC in that it uses an ORPHAN state, similar to the UNDECIDED state in MOBIC, along with CM and CH states. As an improvement to all the clustering algorithms mentioned above, it includes two-way message passing, where a CH accepts or denies a request to join a cluster. This is an important factor in using clustering mechanisms with infrastructure-based architectures where clusters might have size limits due to load-balancing between clusters. However in [13], all CMs are only one hop away from the CH, which limits the size of the cluster and does not exploit the capabilities of joining multi-hop VANET routing and IP-based routing.
2.2.5 Network Criticality-based Clustering

Network Criticality [19] is a metric that measures the robustness in a network by considering the effects of topology and load on various links (nodes) in the network. Consider the model of a network as an undirected weighted connected graph $G = (N, E, W)$, where $N$ is the set of nodes, $E$ is the set of graph links, and $W$ is the weight matrix of the graph. Assuming that time is discrete, we consider a finite-state irreducible Markov Chain with transition probabilities $p_{ij}$ of transitioning from state $i$ at time $t$ to state $j$ at time $t+1$. We can represent the Markov chain by a state transition diagram with states as nodes in a graph and edges corresponding to allowable transitions, and labels associated with the edges denoting the transition probabilities. We can also view the Markov chain as a random walk on the $n$-node graph with next-step transition probabilities $p_{ij}$, where $p_{ij} = 0$ if $j \notin A(i)$ and in the case that $j \in A(i)$, it is defined as:

$$p_{ij} = \frac{w_{ij}}{\sum_{k \in A(i)} w_{ik}} \quad (2.3)$$

where $A(i)$ is the set of adjacent nodes of $i$ and $w_{ik} \geq 0$ is the weight of link $(i,k)$ [19].

Let a random walk start from a source node $s$ and reach a destination node $d$ for the first time, after a random number of steps. The random-walk betweenness $b_{sk}(d)$ of a node (link) $k$ for source-destination pair $s-d$ is the expected number of times that the random walk passes through node $k$ in its journeys from source $s$ to destination $d$. The total betweenness of node $k$ is the sum of $b_{sk}(d)$ over all possible $s-d$ pairs [19].

Node Criticality of a node $i$ is defined as the random-walk betweenness of that node over its weight ($w_i$), which is itself calculated as the sum of the weights ($w_{ij}$) of its incident links [19]. Link criticality and betweenness are defined in a similar way. The criticality of a link $(i,j)$ ($\tau_{ij}$) is defined as the betweenness of the link ($b_{ij}$) over its weight ($w_{ij}$). A high value of $\tau_{ij}$ indicates that the high sensitivity of pair $(i,j)$ to changes in the topology of the network. Therefore, low values of $\tau_{ij}$ are desired. In terms of node criticality, a low ($\tau_i$) indicates that node $i$ is more robust to changes in the network topology than its
neighbouring nodes, thus it can act as a CH for its neighbours.

The weight metric required for network criticality calculation is application-dependent. Based on the needs of application, the weight metric used might be the capacity of the link, the effective capacity of the link (taking interference into account), the distance between the two endpoints of the link, etc. Since we are interested in using a mobility-dependent weight metric to account for the estimable mobility of vehicles in VANETs, two possible metrics would be the estimated future distance between the two endpoints of the link (after a constant time $T$) or the Link Expiration Time (LET) defined in [24]. Due to the simplicity of the LET metric compared to the future distance metric (which depends on a constant $T$), we choose the LET [24] as the weight of robustness for VANET clustering. This weight defines how long two nodes are expected to have a direct link with each other, assuming that they maintain their velocities, and the transmission range is $r$. More precisely, let every node $j$ have a position vector $(x_j, y_j)$ and a velocity vector $(v_{x_j}, v_{y_j})$. Therefore, given that a pair of nodes $(i,j)$ can communicate with one-hop (the distance between them is less than the transmission range), $\text{LET}_{ij}$ can be calculated as shown in (2.4), where $a = v_{x_i} - v_{x_j}$, $b = x_i - x_j$, $c = v_{y_i} - v_{y_j}$, $d = y_i - y_j$.

$$\text{LET}_{ij} = -\frac{(ab + cd)}{a^2 + c^2} + \sqrt{\left(\frac{(a^2 + c^2)r^2 - (ad - bc)^2}{a^2 + c^2}\right)}$$

(2.4)

Therefore, in order to use the criticality metric for clustering, nodes calculate their local node criticalities using LET as a weight (based on the criticalities of nodes in their surroundings) and send them to each other. Setting LET as a weight, we have that the longer the LET is for a link, the less likely it is that the link will break soon; thus that link is less critical than other ones. Nodes that have the lower ($\tau_i$) values (large number of such links) are more likely to be chosen as CHs.

Note that the criticality metric used here is the localized version of the criticality metric defined in [19], since the calculations are done by nodes independently, rather than by a centralized server. The centralized version uses the global view of the whole
network, accounting for all possible source-destination pairs in the network to calculate network criticality values. This technique is not useful for VANETs due to the size of the network, which could be made of tens of thousands of vehicles and would increase the run-time of the criticality calculation algorithm. Therefore, we resort to the localized version of the criticality algorithm, which can be calculated in a distributed manner by each node, based on its 2-hop neighbour view of the network (made of around a hundred nodes for common vehicle traffic patterns on a highway), and is a good approximation of the centralized criticality metric defined in [19].

Realizing the disadvantages of clustering algorithms mentioned above, we adopted Network Criticality as the robust clustering metric and developed the ASPIRE clustering algorithm that maximizes the advantages of these methods in the intense mobile environment of VANETs. ASPIRE clustering adopts the notion of using Cluster Contention Interval for CH neighbours from MOBIC. However, it does not suffer from the bad CH leaders problem in MOBIC as its adaptive maintenance algorithms help it to react efficiently to topology changes. ASPIRE clustering also adopts the idea of future distance prediction used in Affinity Propagation clustering and Density Based clustering. However, ASPIRE clustering leads to a stable clustering algorithm without the need for time averaging of subsequent values. Moreover, ASPIRE clustering employs the idea of two-way messaging as in Mobility-based clustering, in order to better interact with NEMO-based infrastructure and support load-balancing between clusters. However, Mobility-based clustering is inferior to other schemes since it does not capture the robustness of the network links by using metrics such as affinity, density, or criticality.

Due to its accurate measurement of network robustness, criticality is also used in this thesis to group clusters of vehicles into NEMOs, which has not been done by any clustering algorithm to date. Therefore, throughout our design of the ASPIRE clustering algorithm, our aim was to achieve stable clusters and stable Mobile Networks at the same time. The ASPIRE clustering algorithm is presented in Chapter 4.
Chapter 3

ASPIRE Network Architecture

3.1 Architecture Overview

The Adaptive Service Provider Infrastructure (ASPIRE) is a multitier NEMO wireless ad hoc network architecture for general purpose wide-area vehicular communication on highways. ASPIRE is based on the self-organizing network (SON) paradigm, and it views vehicles as the main contributing infrastructure in the network. ASPIRE mitigates the cost of dedicated fixed infrastructure by using existing wireless equipment, moving infrastructure and existing Cellular connectivity to provide extended network connectivity.

ASPIRE considers vehicles as being MNNs, rather than MRs in a NEMO, providing caching capabilities between vehicle clusters, reducing cost of deployment and mitigating the overhead of accessing the Internet for each vehicle request or topology change. Moreover, ASPIRE provides extended network connectivity by dynamically building on vehicle clusters to form more stable NEMOs, using trucks and ‘good-acting’ cars as MRs. By sharing information between these ‘moving infrastructure’ elements and predicting topology changes, ASPIRE ensures that NEMO elements move as a unit for long intervals (as required by [7]) and handover between NEMOs is minimized. This leads to efficient management of nodes in terms of addressing scheme, routing and load balancing between
NEMOs and clusters.

In this Chapter, we state the design goals of this architecture, describe the communication scenarios it supports, identify the details of its design and explain how ASPIRE addressing and routing are performed.

3.1.1 Design Goals

In this section, we state the design goals which led to the ASPIRE Architecture. They take into account the different communication modes based on user demand, different vehicular applications that need to be supported and various requirements for vast deployment. They help serve as guidelines and help understand various choices made during the design process of the network architecture. Some of these design goals originate from [18] and others have been added specifically for the ASPIRE architecture.

1. IPv6 support: The design should support both IPv6 networking and geographical VANET networking.

2. Communication Endpoints: The design should support V2V, V2I and communication between various endpoints in the internet.

3. Geographical broadcasting: The design should support transmitting data to a geographical position or zone.

4. Communication modes: The design should support self-organized ad hoc communication network with/without centralized infrastructure support (such as base stations).

5. Destination sets: The design should support point-to-point and point-to-multipoint communication.

6. Internet Connectivity: IP nodes inside vehicles should be able to communicate with IP nodes in the internet.
7. Seamless connectivity: The design should support ubiquitous connectivity, media diversity and disconnected access.

8. Mobility: A vehicle should be able to change its point of attachment and reconnect easily to the network.

9. Adaptability: Management policies can be dynamically changed, according to the application and the network environment.

10. Scalability: The design should work with a large number of vehicles, in sparse/dense conditions and go hand-in-hand with the Internet routing structure when possible.

11. Security and location privacy: Authentication, authorization, data confidentiality and location privacy modules can be easily included in the architecture, when required.

12. Cost and Reusability: The design should be able to use the connectivity of existing Base stations and inexpensive off-the-shelf wireless equipment, when possible.

13. Performance: The design should include or allow the addition of features such as priority, reliability, latency, efficiency (low overhead), fairness, and robustness.

14. Protocol layering: The design should follow the classical Internet Protocol approach in an end-to-end manner.

15. Adaptive Mobility Management: The design should use the predictive motion of various vehicles to maintain their connectivity.

### 3.1.2 Communication Scenarios

This section presents a few examples of communication scenarios that should be supported by the designed architecture. They are listed according to the communication mode and the destination range.
**Web browsing in vehicle:** The designed architecture that enables web browsing of a vehicle by accessing a webpage on the internet should support:

1. Endpoints: Vehicle originator and Internet destination or vice versa.
3. IPv6 signaling: IPv6 mobility management between vehicle and home agent.
4. IPv6 application: Webpage delivered to the vehicle from some well-known server in the Internet.

**Vehicle-to-Vehicle texting:** The designed architecture that enables texting between vehicles should support:

1. Endpoints: Two vehicles inside the vehicular network.
3. IPv6 signaling: IPv6 mobility management between vehicle and home agent, support of NEMO and VANET routing.
4. IPv6 application: Texting between 2 vehicles.

**Alarm broadcast to groups of vehicles:** The designed architecture that enables Weather/Accident/Alarm broadcast from Service Provider to various groups of vehicles in the network should support:

2. Destination range: Specified geographical area.
3. IPv6 signaling: MR management and support of vehicle mobility management within NEMOs.
4. IPv6 application: Weather/Accident/Alarm broadcast notification.
3.2 ASPIRE Architecture

This section describes design of the ASPIRE architecture, including the VANET model that it is based on, the scenarios it should operate in, along with the details and components of the design.

3.2.1 ASPIRE VANET Model

Fig. 3.1 illustrates the VANET model used in the ASPIRE architecture. On the one hand, the Fixed Service Provider (Fixed-SP) is an existing Internet access provider that controls base stations and offers internet access as a regular access provider through a GSM network or LTE network. On the other hand, the Adaptive Service Provider (Adaptive-SP) is a new type of SP that we are designing based on the SON paradigm. The Adaptive-SP manages moving Infrastructure (trucks and cars) in the vehicular network, along with Home Agents (HAs) and various servers in its fixed network. It offers its services to subscribers located in the vehicular network and uses the services of the Fixed-SP to access the Internet and reach its fixed network. A vehicle can access the Internet provided by the Fixed-SP through a base station or provided by the Adaptive-SP which deploys NEMO-based infrastructure.

We assume that vehicles and trucks both have Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication capabilities, namely that they possess GSM/LTE (V2I) and IEEE 802.11 (V2V) communication capabilities, with ranges of around 1000m and 250m respectively, but only turn on their GSM/LTE antennas when acting as a MR. The AUs within a vehicle are connected to the network through a NAT. This ensures that several AUs in one vehicle use one external IP address (the vehicle IP address) and removes the overhead of using Nested NEMOs (NEMO within a NEMO) [7].
Figure 3.1: ASPIRE VANET Model

3.2.2 ASPIRE Architecture Scenarios

The idea of the ASPIRE Architecture originates from the following four scenarios.

1. Scenario 1: The network consists of a set of vehicles that dynamically group themselves into clusters in the absence of fixed infrastructure.

2. Scenario 2: There is an additional fixed infrastructure in terms of a cellular base station. Unlike RSUs, these base stations are already deployed on the roads and have almost full coverage across and within cities.

3. Scenario 3: There are Trucks/Taxis in highways/cities, controlled by the SP, that have higher processing, security, and bandwidth capabilities than regular vehicles, and act as mobile infrastructure for the network, in the absence of base stations.
4. Scenario 4: There is additional infrastructure in terms of cellular base stations compared to Scenario 3. This is the most general scenario: It changes into Scenario 3 in the absence of base stations in a certain area, into Scenario 2 when a truck/taxi takes an alternate route off the highway, and into Scenario 1 in the absence of both base stations and trucks/taxis in a certain area.

### 3.2.3 Detailed ASPIRE Architecture Design

Fig. 3.2 illustrates the ASPIRE Network Architecture, based on the four scenarios above. The ASPIRE architecture consists of vehicles that adaptively group themselves into stable clusters of low relative mobility, taking roles of Cluster Member (CM) and Clusterhead (CH) in a cluster. Each cluster has a CH and clusters are grouped in Mobile Networks (NEMOs), each with a MR (Fig. 3.2). The ASPIRE architecture includes both a Fixed-SP and an Adaptive-SP as described in Section 3.2.1 and their fixed networks are shown here. The Fixed-SP Fixed Network (Fig. 3.2) is composed of HAs and several servers (ex.
SGSN, GGSN, ...) needed by the Fixed-SP. The Adaptive-SP Fixed Network (Fig. 3.2) is composed of HAs, Authentication servers, and other Management servers. On the one hand, the Adaptive-SP manages trucks and cars in the vehicular network, along with its servers in its fixed network. On the other hand, the Fixed-SP ensures session continuity for all vehicles through the seamless handover for the MRs using their HAs in its fixed network. In other words, the Adaptive-SP offers some of its services to subscribers located in the vehicular network using the Internet connectivity it attains from the Fixed-SP.

As shown in Fig. 3.2, a vehicle A can have two types of Correspondent Nodes (CNs), namely a fixed CN (CN_A) acting as a web server, or a mobile CN (vehicle K) sending a text message to Vehicle A for example. The path of a message from CN (CN_A) to vehicle A is shown in Fig. 3.2. Note that base station coverage areas are not assumed to overlap (Fig. 3.2). Note also that both trucks and vehicles can take the role of a MR, but trucks having a priority over cars, since we assume that they have higher processing, security, and bandwidth capabilities than regular vehicles, and we take into account that trucks usually maintain relatively equal velocities with each other and their motion can be well estimated.

The distinguishing feature of the ASPIRE architecture is the view of a NEMO as being formed of a number of vehicles, rather than being one vehicle. ASPIRE considers vehicles to be MNNs in a NEMO and to get connectivity to the internet through the MR, which might be a truck or another vehicle. This is based on the idea that groups of vehicles on a highway move as a unit for long intervals, thus can be considered together as a NEMO, where one vehicle, acting as the MR, sends binding updates on behalf of all the vehicles in the NEMO. This is also based on the idea of seeding in P2P networks, which is used when a user requests certain information (or a block of data) that has already been requested by other neighbouring peers. Therefore, ASPIRE provides caching capabilities between vehicle clusters and NEMOs, reducing the overhead and cost of accessing the Fixed-SP Network for each vehicle request or binding update due to a topology change.
Moreover, compared to RSU Architectures, where fixed infrastructure is used and the handover of vehicles between RSUs is very common, the handover cost of vehicles in ASPIRE between NEMOs is greatly reduced due to the fact that the MR is also moving with the vehicle and vehicles only choose MRs that they will be in contact with for a long time. Moreover, as long as a vehicle is within the same Mobile Network, it maintains its IP address and its sessions.

NEMO Route Optimization (RO) is made possible through V2I and multi-hop V2V communication. For example, two passengers in different vehicles (in different NEMOs) in the vehicular network might be texting to each other. Instead of accessing the internet (and HAs of MR and MNN) for each message, NEMO RO can help deliver the messages through V2V communication across the NEMOs. NEMO RO leads to less tunneling through multiple HAs, less IP encapsulation, and less load on the MR and HAs. All messages traversing through the MR-HA_{MR} tunnel going through the Fixed-SP network have encapsulated IP headers, hiding the addresses of the MNNs and their HAs. Unless a MNN connects directly through a base station, it will not be known in the Fixed-SP network and will not possess a HA there.

The designed ASPIRE architecture should not only operate in Scenario 4 described above, but should also operate in disconnected vehicular network situations (described in Scenarios 1-3 above). Therefore, we intend to maintain the operation of the VANET in the absence of Trucks, base stations or both, through the clustering algorithms in Chapter 4. Our aim is to have vehicles remaining in a NEMO for long intervals, maintaining their connectivity and session continuity. In addition, we aim to have load balancing between clusters using a low number of CHs. Hence, we target forming NEMOs with inter-NEMO stability, typically formed of a few clusters of vehicles, being a few hops away from the MR and having low relative velocities within the NEMOs. Thus, the algorithms presented in Chapter 4 aim to provide high network connectivity, low CH change rate and high cluster size, among others.
3.2.4 Protocol Layering in the ASPIRE Architecture

![ASPIRE Protocol Stack](image)

The ASPIRE Architecture combines the concepts of NEMO and multi-hop VANETs. Its focus is to support communication scenarios for non-safety vehicular applications, i.e. requiring both IPv6 networking and geographical VANET routing capabilities. Thus, the ASPIRE protocol stack, inspired from [18] and shown in Fig. 3.3, includes the following:

- The Application (Upper) layer is over TCP/IP.

- IPv6 is over the VANET Layer (for communicating with vehicles) or over the GSM or LTE layer (for communicating with base stations).

- The VANET Layer plays the role of the sub-IP layer for IPv6 as explained before.

- The VANET Layer identifier (VANET ID) plays the role of the sub-IP address in the vehicular domain, providing reachability across the vehicular network.

- The VANET Layer is running over 802.11 a/b/g/p and can run over other wireless technologies, if needed.
3.2.5 ASPIRE Physical Components

Based on the required architectural components, we define the network protocol stacks of a truck and a vehicle in Fig. 3.4-(a) and Fig. 3.4-(b), respectively. The VANET layer provides reachability in the vehicular network and simplifies forwarding. IP routing is rarely used within the vehicular network, and the VANET ID is the identity of a vehicle throughout the vehicular network (even across mobile networks). Fig. 3.4-(b) shows how various AUs can connect to the vehicular network through Ethernet/WLAN/Bluetooth connection with the OBU in the vehicle.

(a) Truck Protocol Stack
(b) Vehicle Protocol Stack

Figure 3.4: Truck and Vehicle Protocol Stacks

3.2.6 ASPIRE Architectural Components

The ASPIRE architectural components are defined as the Mobile Router, the Cluster-head, and the Cluster Member:

The Mobile Router: A stable moving infrastructure, typically controlled by the Adaptive-SP, acts as a Mobile Gateway for Adaptive-SP users around it. The MR accesses the internet and connects to the fixed network of the Adaptive-SP through the Fixed-SP (base stations). The MR is the filter between the VANET network and the internet and handles the encapsulation and decapsulation of IP packets for that purpose.
However, the link between the MR and Fixed-SP is of limited capacity and is expensive. Thus, the MR tries to handle most of its operations at the vehicular network level, and only uses that link for essential communication. For this purpose, the MR typically should have a higher capacity, larger power, and a longer range, compared to regular vehicles. Several MRs exist in the network, each providing connectivity to their corresponding NEMOs, based on [7]. For example, MRs forward packets to CNs somewhere in the internet on behalf of the MNNs in their NEMOs and forward packets from CNs in the internet to the MNNs. MRs are an integral part of the architecture, and significantly affect the security, performance and mobility management of the ASPIRE system.

The Clusterhead: A vehicle that possesses good links with its surrounding vehicles, makes decisions for allowing vehicles to join into its cluster, and tries to maintain the connectivity of its cluster members. The CH of a cluster of vehicles may change with time, based on changes in the topology, changes in relative velocities, or congestion in the network. The CH is responsible for its cluster members, in the way that a MR is responsible for its NEMO members (members of the NEMO), but a CH is not a mobile gateway to the internet and it does not possess better, capacity, power and range like some MRs (trucks). A CH is just like another vehicle, but with better physical links, and relative velocities with its surrounding vehicles that make up its cluster. Moreover, CH changes in a cluster are far more common than MR changes in a NEMO. A CH relies on a MR for internet connectivity, has more responsibilities than a regular vehicle and can sometimes be asked to play the role of a Mobile Router for a short time, if needed.

The Cluster Member: The Cluster Member (CM) is a vehicle that joins a cluster of vehicles after gaining approval from the corresponding CH. It forwards the packets of other CMs and maintains a list of its neighbours. It relies on the MR for internet connectivity and on the CH and other CMs in its cluster for forwarding its packets.

Typically, a truck is configured to function as a MR, while a regular vehicle is configured to function as a CH or a CM. In cases where there are no sufficient trucks, a regular
vehicle (typically a CH) can play the limited role of a MR for a short period of time, until connectivity with a truck is regained.

3.3 ASPIRE Addressing and Routing

In this section, we discuss the basics and assumptions of the ASPIRE architecture, in terms of addressing and routing. We also describe how IP packets are delivered within this architecture.

3.3.1 Assumptions and Basics

Every vehicle in the ASPIRE network has an OBU through which it connects to the network. Before a vehicle initiates a request to connect to the Adaptive-SP network, its OBU possesses a MAC Address, assigned to it from the vehicle manufacturer or the Adaptive-SP. Moreover, the Adaptive-SP Fixed Network includes a Registration Authority (RA) used for authenticating vehicles before they join the vehicular network. For this purpose, vehicles are preloaded with a public/private key pair, a 64-bit VANET layer ID, a corresponding public key certificate for that ID signed by the RA, and the IP address and public key of the RA in the Adaptive-SP Fixed Network. In addition, for location privacy purposes, a vehicle might be pre-loaded with multiple VANET layer IDs and their corresponding certificates. The MAC address and VANET layer ID are assumed to be unique at least within a geographic macro region, such as a continent.

The VANET Layer shown in Fig. 3.4 is common to all vehicles, and provides geographical routing and is able to transport and forward IPv6 datagrams without using the IPv6 layer. The VANET layer protocol provides routing of unicast and multicast packets, based on the geographical position of source, destination and intermediate nodes. One common and simple strategy for unicast geographic routing is greedy forwarding, which is composed of: Location service signaling to resolve the destination’s position and
routing of packets hop-by-hop, selecting the closest one-hop neighbour to the destination. Geocasting (used for IPv6 multicast packets) is an enhanced version of unicast greedy forwarding, where the destination is a geographical area, rather than a single geographic location. As shown in Fig. 3.4, the IPv6 layer is put on top of the VANET layer. The VANET layer provides Ethernet-like capabilities for standard datagram forwarding. The link seen by IPv6 layer includes nodes that are not directly reachable, but are portrayed as such by the VANET layer. In other words, the VANET layer presents to IPv6 a multicast link which includes a partition of the VANET, made by all nodes, within a certain geographical area, such as in a NEMO or a cluster. The VANET layer introduces some overhead, but it simplifies routing since the IP headers are not processed within the vehicular domain, except at the end points.

The HAs available in the Fixed-SP Fixed Network (see Fig. 3.2) are not NEMO-compatible (ones that support Network Mobility). Thus, the Adaptive-SP has to provide its own NEMO-compatible HAs for the MRs in the network; the HA for the MRs in the Fixed-SP Fixed Network are only used to provide reachability to the internet and thus to its NEMO-compatible HA in the Adaptive-SP Fixed Network. Note that HAs in each of the Fixed-SP and Adaptive-SP networks can be chosen to be geographically close to the MR when it first joins the corresponding network. Moreover, the Fixed-SP does not know anything about the packets of a regular vehicle (MNN) that are going through its network to reach the Adaptive-SP network, because they are tunneled through it and encapsulated by the IPs of the MR and its HA. Based on these assumptions and ASPIRE addressing basics, we describe next how packets are delivered in the ASPIRE Architecture.

3.3.2 IP Packet Delivery in the ASPIRE Architecture

IP packet delivery in the ASPIRE architecture involves the following entities:

1. IP originator: Runs an application and sends a packet with a suitable IPv6 header.
2. VANET source: Adds VANET and IEEE 802.11 headers with suitable address and identifier.

3. Choosing VANET neighbour: VANET neighbours are nodes that can communicate directly with one another using wireless connectivity. There is no transitivity in neighbour relationships as is the case for IP neighbours. Using the geographic routing protocol, the VANET source chooses the VANET neighbour, closest to the destination, and forwards the packet to it. This neighbour is designated by the destination MAC address in the IEEE 802.11 frame.

4. Choosing an IP next hop: The IP next hop is a destination from a VANET layer viewpoint. Before IP next hop, all intermediary nodes only check the packet’s VANET layer header, and ignore the IP header. Only when the packet reaches IP next hop, it consults the IP header to make a forwarding decision. Choosing an IP next hop includes the following:

   - If a destination is reachable through VANET forwarding (within a certain geographical distance), its IP next hop is the destination itself.
   - If not, the IP next hop is the MR.
   - The IPv6 header is encapsulated in the VANET header and IP next hop is designated by the destination VANET identifier in the VANET hear.

5. Destination: The node where the IP Packet is to be delivered to, designated by the destination IPv6 address in the IPv6 header

For a typical scenario, the packet delivery entities are shown in Fig. 3.5. IP packet delivery over VANET Layer involves the following steps:

1. The source generates an IPv6 packet with destination IPv6 address, encapsulates it with the VANET layer header with IP Next Hop’s VANET ID and encapsulates
with the 802.11 header with the VANET neighbour’s IEEE 802.11 MAC address as shown in Fig. 3.6.

2. For IP packet delivery, the source needs to find the VANET Layer ID of the IP next hop from a given destination IPv6 address and find the IEEE 802.11 MAC address from a given IP next hop’s VANET identifier.

3. The IP next hop is usually the MR, whether it is a Truck or a CH. Thus, the source should already know the MAC address of the IP next hop. However, this requires cooperation among the nodes in a mobile network, and help from the MR in providing inter-NEMO information. Next, the MR forwards the packet (after encapsulation) through the MR-HA_{MR} tunnel to the Home Agent of the source A HA_{A} and subsequently to the Correspondent Node (CN) of A.

4. The destination could be a CN in the (moving) vehicular network or a fixed/moving node anywhere in the Internet. Fig. 3.5 shows a fixed destination CN_{A}, acting as a
web server for vehicle A. Note that if the CN is in the VANET, no IP encapsulations or tunneling are required, since the IP datagram is forwarded through the vehicular network to the destination vehicle using its VANET Layer ID. The MR-HA<sub>MR</sub> tunnel would only be used in this case if the physical location of the vehicle is unknown.

5. Note that the packets delivered through the MR-HA<sub>MR</sub> tunnel use IP encapsulation as described in Fig. 1.3. Therefore, both the MR and HA<sub>MR</sub> are responsible for encapsulating the packets before sending them through this tunnel and decapsulating the packets transferred through this tunnel before delivering them to the destination.

![Packet Headers](image)

**Figure 3.6: Packet Headers**

### 3.4 Chapter Summary

In this Chapter, we presented the ASPIRE architecture design. After identifying the design goals starting from non-safety vehicular applications, we presented the ASPIRE VANET Model, where we defined a new type of Service Provider, namely the Adaptive Service Provider. Next, we presented the detailed design of the ASPIRE architecture, including its protocol layering, and architectural and physical components. We also discussed various addressing and routing issues, relevant to the architecture, and presented how IP Packets are delivered through this network.
Compared to various architectures presented in the literature, ASPIRE is the first VANET architecture to consider a group of vehicles, rather than a single vehicle, as a Mobile Network. Exploiting the advantages of Network Mobility and the heterogeneity of vehicles, ASPIRE mitigates the cost of dedicated fixed infrastructure by forming Mobile Networks of vehicles and using them as the main infrastructure in the network. Coupled with 3G/LTE connectivity, ASPIRE’s support for local communication through clustering further reduces costs, increases effective network throughput and increases the robustness of the network. The ASPIRE clustering algorithm is further discussed in the next Chapter.
Chapter 4

ASPIRE Clustering Algorithms

This Chapter proposes a new clustering algorithm for VANETs, based on network criticality [19]. Calculating network criticality, as presented in the background section, is a centralized, global technique to find the robustness of a communication network. Here, we adapt network criticality for clustering in VANETs by using local network criticality values and clustering in a distributed manner. The algorithms presented aim to provide high network connectivity, low CH change rate and high cluster size, among others. We first present the design requirements, along with an overview of the ASPIRE clustering algorithm. Next, we discuss the details of the design, including how ASPIRE nodes build neighbour relationships, maintain their architectural roles in the network and form various clusters and NEMOs. While presenting the algorithms in this Chapter, we avoid implementation issues related to addressing, routing, authentication, privacy and packet encapsulation.

4.1 ASPIRE Clustering Requirements and Design

This section identifies the design requirements for the ASPIRE clustering algorithm. In addition, it presents an overview of the design, including its major features.
4.1.1 ASPIRE Clustering Requirements

In this section, we state the design goals which led to the ASPIRE clustering algorithms. They take into account the different mobility scenarios of vehicles, channel conditions in the mobile environment of VANETs and different vehicular applications that need to be supported. They help serve as guidelines and help understand various choices made during the design process of the clustering algorithms.

1. Communication Endpoints: The design should reduce the relative mobility between communicating endpoints, consequently leading to intra-cluster stability.

2. Communication modes: The design should support self-organized ad hoc clustering with/without centralized infrastructure support (such as base stations).

3. Seamless connectivity: The design should support ubiquitous connectivity by maintaining high network connectivity.

4. Mobility: The design should be continuous and efficient in adapting to changes in the network topology, yet reduce unnecessary cluster head changes.

5. Traffic Adaptability: The design should operate in sparse/dense and should take advantage of the richer centralized information from the network architecture.

6. Robustness: The design should be robust to channel error and interference between nodes.

7. Scalability: The design should work with a large number of vehicles, and lead to large clusters and NEMOs of similar sizes to support load balancing.

8. Cost: The design should produce reduce cost by producing large clusters and NEMOs, reducing unnecessary messaging and limiting the required forwarding between communication endpoints.
9. Handover: The design should reduce the number of handoffs between NEMOs in order to maintain high network connectivity.

10. Management: The design should simplify management of the network by producing large clusters and NEMOs. It should also allow a SP to make changes to the clustering algorithm in a simple manner.

11. Performance: The design should support caching of information within clusters and NEMOs to reduce the use of 3G/LTE connectivity.

12. Heterogeneity of vehicles: The design should make vehicles with higher capabilities or better connectivity play more important roles as CHs or MRs.

13. Motion Estimation: The design should perform accurate estimations for the motion of various vehicles to maintain their belonging to clusters and NEMOs for long intervals.

Unlike various clustering algorithms discussed in the background, the ultimate aim of the ASPIRE clustering algorithm is to create large clusters and provide high network connectivity. In this thesis, we define network connectivity as the proportion of time a vehicle is connected to a MR; thus to the Internet, in the ASPIRE architecture. In contrast to regular clustering algorithms, we are willing to sacrifice low cluster head durations or high number of cluster head changes for example, if those changes result in better connectivity. However, we can see that intelligent cluster decision making using motion estimation and the ability of the criticality to capture the robustness of the network enable the ASPIRE clustering algorithms, not only to achieve high network connectivity and large clusters sizes, but also to minimize unnecessary cluster head changes and prolonging cluster head durations.
4.1.2 Overview of the ASPIRE Clustering Design

Nodes in the ASPIRE architecture use network criticality distributively to form clusters and NEMOs. Each node $i$ in the ASPIRE network periodically computes the link criticalities ($\tau_{ij}$) between itself and each of its one-hop neighbours. It then calculates its node criticality ($\tau_i$) as the sum of its link criticalities ($\tau_{ij}$) and transmits it to its neighbours. Each node keeps a neighbour list based on the messages it has received from the nodes in its surrounding. The entries of this list are updated and removed (if expired) to maintain the freshness of the list.

In addition to periodically removing old entries from its list, every node periodically checks whether it is still connected to the vehicular network (by checking if its CH or MR are available for example). If a vehicle has lost its connectivity to the network or needs to change its role in the ASPIRE architecture (CM, CH, or MR), it will run a corresponding clustering algorithm to find a new CH $j$ (with low ($\tau_j$) or to find a new MR that it can connect to for a long time, or to become a MR in the network. These algorithms are based on using criticality, as a measure of local connectivity, as well as estimating vehicle positions and mobility. Together, these algorithms ensure high cluster stability and high network connectivity in the network through forming clusters and NEMOs, where nodes are robustly connected and have low relative mobility to one another.

Note that criticality is calculated by each ASPIRE node in a distributed manner. While in normal network criticality calculation [19], a central computer has access to all criticality calculations for each node and makes global criticality calculations for the entire set, nodes in ASPIRE are forced to make independent criticality calculations and clustering decisions using only the local information. Moreover, ASPIRE gives a SP the ability to take advantage of the global view of the network and use resources more efficiently by load balancing between clusters and NEMOs. This is possible by dynamically changing the maximum cluster size and maximum NEMO size for each cluster and NEMO in the network, based on topology changes, congestion, and interference.
4.2 Automatic Neighbour Relation Setup

Nodes in the ASPIRE architecture broadcast messages to their neighbours periodically, forming relations with each other and taking roles of CM, CH and MR. This section discusses the information nodes store about their neighbours and the messages they exchange to keep this information updated.

4.2.1 ASPIRE Neighbour List

Every node $i$ maintains a neighbour list, $N_i$, which has a neighbour entry, $N_i^j$, for every neighbour $j$, 1 or 2 hops away. Each neighbour list entry, $N_i^j$, contains the fields shown in Table 4.1.

<table>
<thead>
<tr>
<th>ID$_j$</th>
<th>Unique ID of $j$ (VANET ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x, y)$_j$</td>
<td>Position vector of node $j$</td>
</tr>
<tr>
<td>(v$_x$, v$_y$)$_j$</td>
<td>Velocity vector of node $j$</td>
</tr>
<tr>
<td>CH$_j$</td>
<td>ID of $j$’s current cluster head</td>
</tr>
<tr>
<td>MR$_j$</td>
<td>ID of $j$’s current mobile router</td>
</tr>
<tr>
<td>CS$_j$</td>
<td>Current cluster size of node $j$</td>
</tr>
<tr>
<td>$\tau_j$</td>
<td>Criticality of $j$</td>
</tr>
<tr>
<td>texpire$_j$</td>
<td>Time that node $j$’s last message expires</td>
</tr>
</tbody>
</table>

Table 4.1: Neighbour entry for node $j$

This neighbour list is populated by the periodic messages sent to node $i$ from its neighbours. Although node $i$ will have its current transmitted messages at any time, it will not always have the current received messages from its neighbours, because of delay, message error, collision or asynchronous nodes. Therefore, $N_i$ contains all of the last received messages by node $i$. Note that $N_i$ also contains a self-entry, $N_i^i$, of node $i$, where current clustering state information is stored.
4.2.2 ASPIRE Messaging

The hello broadcast period in ASPIRE is defined as $T_H$, and is typically around 1sec. Every node $i$ in the network periodically (every $T_H$ sec) performs the following steps:

1. Independently compute the link criticalities, $\tau_{ij}$, between itself and each of its one-hop neighbours, using Link Expiration Time as the weight of robustness (as explained in section 2.2.5).

2. Calculate its own (node) criticality, $\tau_i$, as the sum of its link criticalities.

3. Update all values in $N_i^i$.

4. Broadcast a HELLO beacon: $<ID_i, (x, y)_i, (v_x, v_y)_i, \tau_i, CH_i, MR_i, CS_i>$.

After node $i$ receives a HELLO beacon from node $j$, it needs to perform the following steps:

1. If there is no entry $N_j^i$ (entry for node $j$ in $N_i$), then add a new entry $N_j^i$.

2. Else update $N_j^i$ with the new information.

3. Compute the time that node $j$’s last message expires ($t_{expire_j}$).

4. Add $t_{expire_j}$ to $N_i^j$.

5. Decrement TTL. If TTL>0, forward the beacon.

Updating the position and velocity vectors in $N_j^i$ is important for the accurate calculation of the current and future distance $FutureDist(i,j)$ between nodes $i$ and $j$. $FutureDist(i,j)$ is defined as the estimated distance between nodes $i$ and $j$ using $(x, y)$ and $(v_x, v_y)$ of each. This assumes that both nodes keep a constant absolute velocity for a $Future Prediction Period$, $T_F$, which can be tuned for different types of mobility. This period is set to 60s for the highway scenario in our simulations. This function is an improved version of the future distance function used in [12], which assumes that a vehicle
maintains constant velocity values $v_{x_j}$ and $v_{y_j}$, rather than a constant absolute velocity. This improvement is necessary, especially for accurate future distance calculations on curved roads, where $v_{x_j}$ and $v_{y_j}$ might be constantly varying.

Updating the remaining values in $N^i_j$ is important because they indicate the current role of node $j$ (whether it is a CM, CH, or MR) along with its connectivity level, which together show whether node $i$ should consider $j$ as a potential CH or MR. This information is even more important if node $j$ was node $i$’s CH or MR and it has changed its role or lost its MR for example. The way that node $i$ reacts to such cases is discussed in the maintenance algorithms in the next section.

Node $i$ makes clustering decisions based on its local view of the network (received criticality values). Based on the current role of a node (CM, CH, or MR), this clustering decision could be an attempt to join a cluster (sent to a CH), join a NEMO (sent to a MR), or become a MR (sent to Adaptive-SP through base station). Alternatively, it could be a response to a cluster join request (if CH), or to a NEMO join request (if MR). The messages for these requests (responses) contain the same entries as in HELLO beacons (in order to provide the most recent information for better decision making), in addition to flags specifying type of request (response). Decision making for clustering is discussed in the next section.

### 4.2.3 Behavior of ASPIRE Messaging

There are some important notes regarding the passing of messages for ASPIRE clustering. Although state information is kept, old criticality values are not needed for neighbouring nodes, and are immediately replaced once newer criticality values are received. Due to the nature of the criticality metric, which captures the robustness of the network, the ASPIRE clustering algorithm inherently has memory, as no time averaging of consecutive criticality values is needed. This property results in less frequent cluster changes and lower number of clusters. Moreover, message passing does not have to be synchronous.
Assuming no collisions or channel error and all nodes use the same value of $T_H$, the received messages and thus the entries in $N_i$ will be at most one $T_H$ old, no matter when each node broadcasts its messages during the $T_H$ interval. If channel error is taken into account, the entries in $N_i$ may become outdated, leading to performance degradation. The forwarding of messages to two-hop neighbours, used in ASPIRE, provides some redundancy and resilience that mitigate this problem. The behavior of ASPIRE with channel error is discussed in Chapter 5.

4.3 Cluster Maintenance and Formation

Fig. 4.1 shows the state diagram for the ASPIRE clustering algorithm. The five states shown in the figure represent the different states a vehicle can be in within the ASPIRE architecture. For trucks, the only possible state is the MR state. The five states for the ASPIRE clustering algorithm are the following:

1. **CM**: A vehicle is in this state when it has a CH, which is connected to a MR.

2. **CH** (connected to MR): A vehicle is in this state if it is a CH and is well connected to a MR.

3. **MR**: A vehicle is in this state when it is a MR and there are no trucks around it (its role as a MR is needed in the absence of trucks).

4. **Find Clusters**: A vehicle is in this state when it is a CM/CH and is trying to join a new cluster.

5. **Find MRs**: A vehicle is in this state when it is a CH, not connected to a MR.

This section describes the cluster maintenance and formation algorithms that lead to the transitions between the ASPIRE states. On the one hand, cluster maintenance algorithms deal with preserving the integrity of the ASPIRE Neighbour List and of the
Figure 4.1: Clustering Algorithm State Diagram
roles between nodes in the network. Thus, if changes in the topology of the network make a CH change its role for example, its CMs should be able to react to this role change by going from the CM state to the Find Clusters state. On the other hand, cluster formation algorithms (run when node is in state Find Clusters or Find MRs) deal with finding the best CHs or MRs to connect to in the network, along with trying to become a MR if no other MRs are in range.

In order to detect required state changes, each node periodically checks whether there are outdated entries in its neighbour list, whether its connectivity to its CH/MR is preserved, whether there are neighbouring CHs or MRs that could replace it, and whether its clustering requests have been approved. These steps help nodes to adapt to topology changes dynamically and form the basis of the ASPIRE clustering algorithms. In addition, this section discusses how ASPIRE clustering algorithms allow the Adaptive-SP to manage the clustering within the network and load balance between clusters and NEMOs, by dynamically changing the maximum cluster size and maximum NEMO size, of clusters and NEMOs, respectively.

### 4.3.1 Cluster Maintenance

Every $T_H$ seconds, node $i$ runs one of the PURGE algorithms shown in Algorithms 1, 2, or 3, depending on its current role of MR, CH, or CM, respectively. In order to keep information current in the ASPIRE Neighbour List and remove old neighbours, all nodes periodically check whether they have any expired neighbours that need to be purged (line 3 in each algorithm).

In order to minimize the number of MRs and maintain large NEMOs, each MR periodically checks whether there is another MR close to it, which would entail the possibility of joining these two NEMOs together. By choosing the best MR among the two MRs to be the MR of the new joined NEMO, we make sure that bad leading MRs are removed and do not stay in charge (of the NEMO) indefinitely. However, since we would
like to attain stable NEMOs and reduce handoffs between them, we have to make sure that joining and splitting NEMOs would be occurring only when necessary and when changes are expected to last for a long interval. In other words, if the possibility of joining two NEMOs arises, we need to make sure that joining them will not be followed by a split after a few seconds.

In the scenario shown in Fig. 4.2, a group of fast moving vehicles (NEMO₁) should be left to bypass a group of slow moving vehicles (NEMO₂) without joining the two NEMOs and having to split them again after the fast group is well ahead of the slow group of vehicles. This would help many vehicles avoid unnecessary handoffs between NEMOs;
thus maintaining their network connectivity for longer durations. The situation is similar when clusters are considered instead of NEMOs.

---

Figure 4.2: NEMO Overtaking

Thus, we use the FutureDist function to predict how long two MRs will be in close proximity to each other, as shown in line 5 of Algorithm 1. Moreover, we use a MR Timer (MRT\textsubscript{j}) to keep track of the number of times MR \textit{j} has been in close proximity with node \textit{i} (line 6), and only attempt to make changes once that timer reaches a limit, MRT-LIMIT, assumed to be 7 seconds (line 7). In that case, if the number of nodes in \textit{i}'s NEMO is less than a certain threshold, Thr\textsubscript{MR}, proportional to the maximum cluster size of a MR, MaxCS\textsubscript{CH}, which is set to 60, and if node \textit{j} has more NEMO Members and has enough room for node \textit{i}'s NEMO Members, then \textit{i} will cease its role as MR and the two NEMOs will join with node \textit{j} being the MR of the new NEMO. This makes node \textit{i} transition from the MR state to the Find Clusters state, as shown in Fig. 4.1.

A CH follows a similar PURGE procedure, shown in Algorithm 2. In this algorithm, CH \textit{i} checks whether its MR is present and is still a MR (lines 0, 5, and 6). If that is not the case, node \textit{i} transitions from the CH state to the Find MRs state, as shown in Fig. 4.1. Moreover, this algorithm uses a CH Timer (CHT\textsubscript{j}) similar to MRT\textsubscript{j} to count the number of times CH \textit{j} has been in close proximity with it. However, in order to choose the good CHs to remain and get rid of badly connected CHs, node \textit{i} checks whether \(\tau_j\)
0: Lost$_{\text{MR}}$ ← 1
1: for all $N_j^i \in N_i$ do
2:     if $t_{\text{expire}}_j < t$ then
3:         Purge $N_j^i$ from $N_i$
4:     end if
5:     if $\text{MR}_i = \text{ID}_j$ and $\text{MR}_j = \text{ID}_j$ then
6:         Lost$_{\text{MR}}$ ← 0
7:     end if
8:     if ($\tau_j < \tau_i$) and ($\text{CS}_i < \text{Thr}_{\text{CH}}$) and ($\text{CS}_i < \text{CS}_j$) and ($\text{CS}_i < (\text{MaxCS}_{\text{CH}} - \text{CS}_j)$) and ($\text{CH}_j = \text{ID}_j$) and ($\text{FutureDist}(i,j) < 450$) then
9:         $\text{CHT}_j = \text{CHT}_j + 1$
10:        if $\text{CHT}_j \geq \text{CHT-}\text{LIMIT}$ then
11:            $\text{CHT}_j = 0$
12:        end if
13:    end if
14: end for

Algorithm 2: PurgeCH (i,t)

$< \tau_i$ in line 8 (node $j$ has a better $\tau$). Furthermore, even with a smaller $\tau$, node $j$ would only be considered as a better CH, compared to node $i$, if they were connected to the same MR or node $i$ was not connected to a MR while node $j$ was (line 12). Only in that case would CH $i$ cease its role as CH, allowing the joining of the two clusters. This ensures that bad acting CHs will not remain indefinitely, and small clusters will merge together to improve load balancing, yet ensure low handoffs between NEMOs (changing a MR). Therefore, node $i$ transitions from the CH state (if $\text{MR}_i \neq -1$) or from the Find
MRs state (if MR\(i = -1\)) to the **Find Clusters** state, as shown in Fig. 4.1.

A CM follows a similar PURGE procedure, shown in Algorithm 3. In this algorithm, CM \(i\) checks whether its CH is present, is still a CH and has the same MR as before (lines 0, 5, and 6). If any of these conditions are false, then CM \(i\) transitions from the CM state to the **Find Clusters** state, as shown in Fig. 4.1.

0: \(\text{Lost}_C \leftarrow 1\)

1: **for all** \(N^j_i \in N_i\) **do**

2: \(\text{if } \text{texpire}_j < t \text{ then}\)

3: Purge \(N^j_i\) from \(N_i\)

4: **end if**

5: \(\text{if } \text{CH}_i = \text{ID}_j \text{ and } \text{CH}_j = \text{ID}_j \text{ and } \text{MR}_i = \text{MR}_j \text{ then}\)

6: \(\text{Lost}_C \leftarrow 0\)

7: **end if**

8: **end for**

**Algorithm 3: PurgeCM (i,t)**

### 4.3.2 Cluster Formation

As seen in Fig. 4.1, when a vehicle is trying to join the network (initial state), or has lost its CH, or has dropped its status as CH or MR, it goes into the **Find Clusters** state, where it will start looking for the best CH and join its cluster.

The situation is shown in Fig. 4.3, where vehicle \(E\) is trying to connect to the cluster of vehicle \(B\). The dashed arrows in the figure indicate the path of messages when authentication to join the cluster is required, in which case the CH (vehicle \(B\)) will verify the credentials of vehicle \(E\) through the Authentication server in the Adaptive-SP Fixed network. Therefore, a node \(i\), that has \(\text{Lost}_C\) set to 1 from any of the three maintenance algorithms above, it will run every \(T_H\) seconds the following algorithm:
1. Attempt to find CH with the lowest $\tau_j$ among CHs that have $MR_j = MR_i$.

2. If unsuccessful, attempt to find CH with the lowest $\tau_j$ among CHs that have any MR.

3. If unsuccessful, attempt to find CH with the lowest $\tau_j$ among remaining CHs in $N_i$.

4. If unsuccessful, set itself as CH of a new cluster aiming for other nodes to join it.

The third case is only unsuccessful if a node does not have any non-full clusters around it, or it has a better $\tau$ than all possible CHs. If node $i$ finds a possible CH (Pos-CH) $j$, whose cluster it would like to join (in any of the first three steps above), it sends $j$ a Cluster Join Request (CjoinReq) that gets forwarded by its neighbours (if needed) to reach node $j$. If successfully received, node $i$ gets an accept/reject decision for its request from node $j$ based on $CS_j$ and $FutureDist(i,j)$. If the CjoinReq has been accepted, node $i$ transitions from the **Find Clusters** state to the **CM** state.

In a similar (but simpler) fashion, CH $i$ looks for a new MR in case it has $Lost_{MR}=1$ and sends it a NEMO Join Request (NEMOjoinReq). This scenario is shown in Fig. 4.4, where vehicle $E$ is trying to connect to a NEMO. It represents a transition from the **Find MRs** state to the **CH** state if the NEMOjoinReq has been accepted.
Chapter 4. ASPIRE Clustering Algorithms

Figure 4.4: NEMO Join

Figure 4.5: Become Mobile Router
In case after several attempts (Mreq-timer ≥ 3) to join a NEMO, no possible MR is yet found (Pos-MR = -1) (this situation would be similar to a MR car or truck leaving the highway), a CH sends a Become MR Request (BecMRReq) to the Adaptive-SP through a possible base station (Pos-BS) and receives an accept/reject decision based on the existence of MRs close to it. This scenario is shown in Fig. 4.5, where vehicle E is trying to become a MR. It represents a transition from the Find MRs state to the MR state if the BecMRReq has been accepted.

### 4.3.3 Cluster Management

Fig. 4.6 shows the interaction between different ASPIRE clustering algorithms. In addition to making clustering decisions based on its local view of the network, each node uses richer information from the Adaptive-SP, which has a better view of the network. In addition to making NEMO join request decisions, a MR can load balance between the clusters in its NEMO and reduce the congestion within some of these clusters by controlling the max cluster size of the clusters within its NEMO. Moreover, such action could also be taken to reduce the bandwidth utilized by a certain CH to provide for the needs of its CMs. In a similar way, the Adaptive-SP can load balance between NEMOs and reduce the bandwidth used by some MRs through changing the max NEMO size for some of the MRs in its network, as seen in Fig. 4.6.

In the event that several users within the same NEMO use bandwidth-exhaustive applications (Voice), the Adaptive-SP can load balance between NEMOs based on bandwidth usage, rather than NEMO size. These cluster management techniques help the ASPIRE clustering algorithms to dynamically and intelligently react to changes in the network and provide the Adaptive-SP with the capability to maximize the resource efficiency of its network and provide users with application-dependent priorities.
Chapter 4. ASPIRE Clustering Algorithms

Figure 4.6: Cluster Management Algorithm Relationships
Chapter 5

ASPIRE Simulation Results

In this chapter, simulation results are presented for the ASPIRE clustering algorithm. The NS-2 simulator was used to validate ASPIRE clustering using Criticality against Density. The metrics used to measure clustering performance include: cluster size, CH changes, CH duration, CM duration, MR duration and network connectivity. The clustering performance is simulated for the ASPIRE clustering algorithm presented in Chapter 4.

This chapter first presents the simulation setup, including both the traffic and network simulators. This is followed by a description of the clustering metrics. Next, mobility simulations are performed to observe the clustering performance as a function of average velocity for the ASPIRE clustering algorithm with vehicles using both Criticality and Density. Moreover, the robustness of the algorithm is compared by plotting the clustering performance metrics as a function of channel error. Finally, the performance of the algorithm in terms of both mobility and robustness is simulated when taking into account the heterogeneity of vehicles, by using both vehicles and trucks in the simulations.
5.1 Simulation Setup

This section describes the simulation setup. Traffic scenarios were provided by the MOVE traffic simulator [25] and simulations were performed using the NS-2 simulator [26]. The traffic scenarios that are the outputs of MOVE supply the node mobility patterns as input tcl trace files to NS-2.

5.1.1 Traffic Simulator

Realistic traffic models for the highway VANET scenario were generated using the MOVE traffic simulator [25]. MOVE is based on the open-source traffic simulator, SUMO [27]. The MOVE tool outputs realistic NS2 traces, which were then used in the NS2 simulations. A rectangular looped 2-lane highway, including lane-changing, was chosen for the highway traffic scenario of the simulations. This loop is 30 Km long and 300m wide, and the two lanes travel around the loop in one direction. The rectangular loop was designed wider than the 250m broadcast range, so that messages from vehicles moving in opposite directions do not interfere with each other. All simulations were performed with 400 vehicles. In order to achieve a realistic highway scenario, the vehicles were given different maximum speeds. Random maximum speeds were assigned to the different vehicles by providing SUMO with a Gaussian distribution of input velocities. Five unique traces were generated for each of the average maximum speed groups of 11.1, 22.2, 33.3, and 44.4 m/s (40, 80, 120, and 160 km/h).

5.1.2 Network Simulator

The ASPIRE algorithm was implemented in NS2 [26]. The 802.11 MAC and the 914MHz Lucent WaveLAN DSSS network card with a radio range of 250m, were used in the NS2 simulations. The vehicles reach their maximum speed if possible and slow down at the turns and when blocked by slower-moving traffic (only 2-lanes are used), creating a
realistic pattern with both low and high density traffic. To ensure load balancing, CHs have a maximum cluster size of 15 and MRs have a maximum cluster size of 60 in the simulations, and cluster size of CMs is set to one. Note that regular nodes broadcast HELLO beacons with TTL=2, while MRs broadcast HELLO beacons with TTL=4 (for larger coverage) in the simulations. Each simulation ran for 1500s, however only the last 300s were used for performance metric calculations. This was to ensure that all vehicles had successfully entered the highway and reached their respective velocities. All of the simulation results were averaged over 5 different mobility scenarios.

5.2 Performance Metrics and goals

To evaluate the cluster stability, the validity of the ASPIRE Architecture and the overall performance of our algorithm, we use the following metrics:

1. Average Cluster Size: Large cluster sizes are important for efficient caching, management, and load balancing, where the CH is the central controller.

2. Average Rate of Clusterhead Change: This metric is useful since it takes into account both CH duration and the number of clusters formed.

3. Average Clusterhead Duration: This metric measures the average length of time a node remains a CH. Long CH durations are important for MAC schemes where the CH is the central controller and scheduler.

4. Average Cluster Member Duration: This metric measures the average length of time a node remains a member of a specific cluster. This metric is a good metric for judging the overall stability of clustering itself, but might not necessarily be a good indicator to the overall connectivity of a node.

5. Average Mobile Router Duration: This metric measures the average length of time a node remains in the role of a MR. Long MR durations are important for low
handovers between NEMOs and maintaining the overall connectivity of the network nodes.

6. Average Network Connectivity: This metric measures the percentage of time a node is connected to a MR, thus to the internet. It accounts for the time a node spends being a member of a NEMO (as a CM or CH) and does not include the time when a node is searching for a MR.

As no clustering algorithm leads to the creation of NEMOs in the literature, we evaluate the performance of Criticality ($\tau$) relative to the Density metric [11] within the realm of the ASPIRE algorithms. For Density simulations, each node $i$ broadcasts its Density (defined as the number of $N^j_i$ entries in $N_i$), instead of $\tau_i$ to its neighbours. Clustering and maintenance decisions are made using the inverse of Density, rather than $\tau$. Thus, nodes with high Density are more likely to be chosen as CHs. Most importantly, although vehicles are moving relative to their NEMOs, we show here that the ASPIRE algorithm, leads to high network connectivity (connectivity to a MR, thus the Internet). This is indifferent to the metric used, whether it is Criticality or Density; thus, it validates the ASPIRE Network Architecture presented in Chapter 3.

The next section presents the mobility simulations for the ASPIRE clustering algorithm with vehicles using both Criticality and Density and the subsequent section presents the robustness simulations for the ASPIRE clustering algorithm with vehicles using both Criticality and Density. The last two section of this Chapter present the performance results when trucks are included, in addition to vehicles in the traffic model. This sections evaluate both the mobility and robustness behavior for the ASPIRE clustering algorithm, using Criticality and Density.
5.3 Mobility Performance with Vehicles

In the first set of simulations, ASPIRE mobility performance is compared using both the Criticality and Density metrics. Using SUMO, highway scenarios are generated with a Gaussian distribution of velocities with means of 11, 22, 33, and 44 m/s for 400 vehicles (no trucks are used in these experiments). Separate simulations are made by changing $T_H$ (Hel in the figures) from 1 sec to 2 sec to self-assess the ASPIRE algorithms. The results for average cluster size, average CH changes, average CH duration, average CM duration, average MR duration and average network connectivity are displayed in Fig. 5.1, Fig. 5.2, Fig. 5.3, Fig. 5.4, Fig. 5.5, and Fig. 5.6 respectively.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{average_cluster_size.png}
\caption{The average cluster size as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and $Hel$ is swept over 1s and 2s.}
\end{figure}

The average cluster size is plotted in Fig. 5.1. Since cluster size is inversely proportional to the number of clusters, large average cluster size (low number of clusters) is desired. It is evident that Criticality leads to a significantly higher average cluster size (for both $Hel=2s$ and $Hel=1s$), compared to Density. This is due to the ability of Criticality values to capture the robustness of the network and help the ASPIRE algorithm...
adjust to topology changes efficiently, compared to Density values, which may change abruptly with topology changes in the network.

As the Hello broadcast period (Hel) increases, clustering decisions are made less often (for both Criticality and Density). This results in decreasing the tendency of breaking clusters or leaving clusters to join other (better) clusters, subsequently keeping large clusters, as observed in Fig. 5.1. In addition, as the mobility of the vehicles increases from 11.1 m/s to 44.4 m/s, vehicles tend to move outside of each other’s transmission range more often. Consequently, more links between vehicles are broken and clusters are split. Thus, increasing the mobility of vehicles from 11.1 m/s to 44.4 m/s leads to smaller clusters, as expected.

Figure 5.2: The average rate of clusterhead change as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s.

The average rate of clusterhead change is shown in Fig. 5.2. As the aim of the clustering algorithm is maintaining the stability of the network, a low rate of clusterhead change is desired. Although both Criticality and Density lead to a comparable average number of CH changes every second, as seen in Fig. 5.2 (and even less changes for 11.1
m/s and 22.2 m/s), Criticality still manages to produce larger clusters as seen before. We also see how an increase in Hel results in a decrease in the decision making rate of ASPIRE algorithms and the number of CH changes per second. Note that for the 11.1 m/s case, CH are more likely to be close to each other for long intervals, making it easier to reach the CH timer limit (CHT-LIMIT) that leads to merging clusters, as explained in Section 4.3.1. This describes the high rate of CH change for the 11.1 m/s case, which is the expense to be paid in order to achieve the large clusters shown in Fig. 5.1. In addition, increasing the mobility of vehicles from 22.2 m/s to 44.4 m/s results in more frequent CH changes, as explained before.

![Figure 5.3: The average clusterhead duration as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s.](image)

Fig. 5.3 and Fig. 5.4 show the average clusterhead and cluster member durations respectively. For a clustering algorithm in general, short clusterhead and cluster member durations are desired. However, since we are using a clustering algorithm to achieve both large clusters and high network connectivity at the same time, we are willing to tradeoff higher cluster member duration and higher cluster duration in exchange for larger
clusters, longer NEMO member durations and higher network connectivity. For example, merging two clusters leads to higher cluster size, which is desired, but also leads to lower cluster member and cluster head durations. Moreover, nodes that lose the connection with their MR, might make decisions to join other clusters within the same NEMO and have shorter CM/CH durations, in order to increase their member duration within the NEMO, subsequently increasing their network connectivity.

Fig. 5.3 and Fig. 5.4 both show a decrease in durations with the increase in mobility from 22.2 m/s to 44.4 m/s, for both Criticality and Density, as expected. The durations are smaller for the 11.1 m/sec cases due to high likelihood of merging clusters, which is an expense to be paid in order to achieve the large clusters shown in Fig. 5.1, as explained above. Furthermore, increasing Hel reduces the rate of clustering decisions, thus leads to an increase in the average CM and CH durations, as observed in Fig. 5.3 and Fig. 5.4.

The average MR duration is shown in Fig. 5.5. When vehicles take on the role of a MR (as no trucks exist in these simulations), it is desired that they remain with that
Chapter 5. ASPIRE Simulation Results

Figure 5.5: The average mobile router duration as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s.

role for long durations, in order to maintain the connectivity of all the members of their NEMO. When two MRs are neighbours for a long time, one of them (the worse one among the two) ceases its role as MR in order to allow the merging of the two NEMOs, as described in Section 4.3.1. As the mobility of vehicles increases, the likelihood of MRs meeting also increases, leading to lower MR durations, as shown in Fig. 5.5. Criticality and Density both lead to similar MR durations, but increasing Hel decreases the decision rate of MRs, thus increasing their durations.

Finally, the average network connectivity of nodes is shown in Fig. 5.6. Network connectivity is calculated as the percentage of time a node is a member of a NEMO (as a CM or CH), averaged over all 400 nodes. Note that network connectivity does not include the time when a node is searching for a MR (disconnection time). In other words, it calculates on average the percentage of time that a node is connected to some MR, thus to the Internet.

Fig. 5.6 shows that Criticality leads to higher average network connectivity than
Figure 5.6: The average network connectivity as a function of velocity for 400 Vehicles with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s.

Density for \(T_H=1s\). This is explained by the superiority of the Criticality metric in measuring the robustness of the network, as compared to the Density metric. While changes in the topology of the network might lead to abrupt changes in Density values and lead to unnecessary cluster merging or splitting, efficient tracking of these topology changes with smooth changes in the Criticality values avoids these unnecessary changes, leading to higher cluster stability (larger cluster sizes) and higher network connectivity.

The low network connectivity for 11.1 m/s and 22 m/s speeds, relative to 33 m/s speeds, is explained by the high likelihood of merging clusters or merging NEMOs, as explained before. Moreover, increasing the mobility of nodes makes the algorithm react less often to changes, thus not adapting fast enough to changes, leading to lower network connectivity, especially at higher speeds as seen in Fig. 5.6. Note that Fig. 5.2 measures the average rate of CH changes/sec for the total of 400 nodes. Therefore, the number of CH changes/sec/node is very minimal for both Criticality and Density metrics. In addition, using no fixed infrastructure and no trucks (as fixed MRs), the network
connectivity achieved is very high. Therefore, the ASPIRE Architecture, presented in Chapter 3, along with its clustering algorithms, presented in Chapter 4, are validated.

5.4 Robustness Performance with Vehicles

In this section, the robustness of the ASPIRE algorithms to channel error using both Criticality and Density is measured. We run the same simulations as before, and measure the same quantities, but varying the channel error, rather than the mobility speeds of vehicles. The channel error is produced with a uniform error model, where received packets are randomly dropped with a certain probability or Packet Error Rate (PER). Simulations were run with channel error varied over the following PER values: 0, 0.1, 0.2 and 0.3. The uniform error model overestimates channel error, especially when compared to the more realistic Nakagami model.

![Figure 5.7: The average cluster size as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s.](image)

We have chosen the speed of 33 m/sec for these simulations, as it is the most realistic
average speed on a highway worldwide. Moreover, messages are kept for longer periods (texpire is 5, instead of 3) to add redundancy for both Criticality and Density cases and no trucks are used. The performance results are displayed in Fig. 5.7, Fig. 5.8, Fig. 5.9, Fig. 5.10, Fig. 5.11, and Fig. 5.12. The performance of both Criticality and Density in all the Figures deteriorates as channel error is increased. However, Criticality maintains the same advantage of higher average cluster size over Density, even at high PERs, as seen in Fig. 5.7. Moreover, an increase in PER leads to a decrease in the average cluster size for both Criticality and Density cases. As the channel error increases, the messages broadcast between nodes are lost more frequently. Therefore, a CM that does not receive messages from its CH for texpire seconds tries to join other clusters, even though its CH might still be within its range. Similarly, a CH that does not receive messages from its MR for texpire seconds tries to join other NEMOs. Moreover, using outdated messages, not yet purged from the neighbour list of a node, might cause it to make it wrong clustering decisions, affecting the stability of the network. Therefore, an increase in channel error leads to unnecessary cluster and NEMO splits, thus decreasing the average cluster size (Fig. 5.7) and deteriorating cluster stability in general.

The performance of both Criticality and Density is similar in Fig. 5.8, which shows that high PERs lead to an increase in CH changes/sec, as expected. However, even at PER=0.3, ASPIRE clustering algorithms still have less than 0.005 CH changes/sec/node, showing robustness with a very reasonable cluster size (Fig. 5.7) under an extreme situation. Criticality and Density have almost similar results in Fig. 5.9 and Fig. 5.10 as before, with a decrease in the durations with the increase in PER, as expected.

In Fig. 5.11, Criticality and Density have almost similar results, with the MR durations increasing significantly with the increase in $Hel$, compared to Fig. 5.5. The likelihood of maintaining MRs as neighbours for the required MR timer limit (MRT-LIMIT) that leads to merging NEMOs, is halved when $Hel$ is doubled. This is made even less possible due to the random loss of packets in these simulations. Therefore, NEMOs are
Figure 5.8: The average rate of clusterhead change as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and $Hel$ is swept over 1s and 2s.

Figure 5.9: The average clusterhead duration as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and $Hel$ is swept over 1s and 2s.
Figure 5.10: The average cluster member duration as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and $Hel$ is swept over 1s and 2s.

Figure 5.11: The average mobile router duration as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and $Hel$ is swept over 1s and 2s.
Figure 5.12: The average network connectivity as a function of channel error for 400 Vehicles with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s.

Fig. 5.12 shows that Criticality leads to a higher average network connectivity, compared to Density, as before. In addition, for $T_H=1s$, both Criticality and Density lead to high average network connectivity (more than 90.5%) up to PER=0.2, which confirms again the robustness of the ASPIRE algorithms to channel error.

### 5.5 Mobility Performance with Vehicles and Trucks

In this section, we are interested in viewing the benefits of the ASPIRE architecture if the heterogeneity of vehicles is exploited. Therefore, we run the same set of mobility experiments as in Section 5.3, but add trucks, in addition to the vehicles in the simulations. The simulations performed in Section 5.3 gave rise to an average of 36 MRs in almost all cases. Therefore, in this section, we add 36 trucks onto the 30Km track used before and perform the same set of measurements versus mobility as in Section 5.3.
The trucks move with the mean velocities 11, 22, 33, and 44 m/s, and try to maintain equal distances between each other. While vehicles can decelerate at 5 m/s but can accelerate only at 0.8 m/s, we have allowed Trucks to be able to both decelerate and accelerate at 5 m/s, in order to try to maintain their mean velocities, after varying from it when faced with traffic or a curved road. This is desired because we want to analyze a situation where vehicles do not need to take the role of a MR due to the numerous number of trucks, as described in the real life example in Section 1.1. The performance results are displayed in Fig. 5.13, Fig. 5.14, Fig. 5.15, Fig. 5.16, and Fig. 5.17. Based on the simulations, no vehicles indeed took the role of MR because of the availability of enough trucks. Therefore, since trucks are the only MRs in these simulations and do not change their state, MR durations are equivalent to the time of the simulation; thus the average MR duration results are omitted.

Figure 5.13: The average cluster size as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s.

Compared to their counterparts in Section 1.1, all the results in this section are improved, for both Criticality and Density simulations. This is expected since having
trucks as MRs provides a high level of stability for the network and future mobility predictions are more accurate when truck motion is concerned. Therefore, vehicles will make better decisions to which NEMOs to join and clusters will not break as easily as before since trucks, unlike vehicles, maintain their roles as MRs. However, trucks are still as likely as vehicles to move out of vehicle transmission range due to their motion relative to different vehicles.

Figure 5.14: The average rate of clusterhead change as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s.

Fig. 5.13 shows a higher average cluster size when compared to Fig. 5.1, with Criticality maintaining larger clusters when compared to Density. This is due to the stability provided by the availability of trucks as MRs. As the CH are less likely to look for new MRs, as they will keep their connections with them for long periods, clusters remain intact for long periods and clusters within the same NEMO merge together, both leading to larger clusters. This also explains the slight decrease in the average rate of CH change, for both Criticality and Density, as shown Fig. 5.14, as compared to Fig. 5.2. The situation is similar in Fig. 5.15 and Fig. 5.16, where CH and CM durations increase
Chapter 5. ASPIRE Simulation Results

Figure 5.15: The average clusterhead duration as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s.

Figure 5.16: The average cluster member duration as a function of velocity for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Average maximum velocity is swept over 11, 22, 33, and 44 m/s, and Hel is swept over 1s and 2s.
significantly due to the stability at the NEMO level with Trucks.

Moreover, Fig. 5.17 shows a higher average network connectivity, when compared to Fig. 5.1. However, the network connectivity level at 11.1 m/s speeds is much higher than before for both Criticality and Density. This is due to the fact that NEMO merges, which are very likely at low speeds, are not possible when trucks are acting as MRs (since trucks maintain their roles as MRs). Therefore, the abrupt effects to the stability of the network due to NEMO merges are avoided when using trucks as MRs, leading to increased network connectivity, especially at lower speeds. Note that cluster merges are still possible here, but do not lead to abrupt effects to network stability as much as NEMO merges.

Therefore, ASPIRE algorithms can exploit the heterogeneity of vehicles to increase the stability of the network with various vehicle mobilites. In addition to creating larger clusters and requiring lower number of CH changes, which can improve load balancing between cluster, this optimization helps improve CH and CM durations significantly,
enabling resource efficient features; such as caching for fast information retrieval within clusters. This also leads to higher network connectivity, which entails lower disconnection times, as desired by users.

## 5.6 Robustness Performance with Vehicles and Trucks

In this section, the robustness of the ASPIRE algorithms to channel error, similar to Section 5.4 is measured, but using both vehicles and trucks as described in Section 5.5. Therefore, we are interested in examining the benefits attained on the robustness of the network by exploiting the heterogeneity of vehicles. The performance results are displayed in Fig. 5.18, Fig. 5.19, Fig. 5.20, Fig. 5.21, and Fig. 5.22.

![Figure 5.18: The average cluster size as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s.](image)

Compared to their counterparts in Section 5.4, all the results in this section are improved, for both Criticality and Density simulations. Due to the additional stability provided by trucks at the NEMO level of the ASPIRE architecture, vehicles are able to
Figure 5.19: The average rate of clusterhead change as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s.

Figure 5.20: The average clusterhead duration as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s.
Figure 5.21: The average cluster member duration as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and $Hel$ is swept over 1s and 2s.

make better clustering decisions, even when the wireless medium is lossy. After nodes go through intervals where they drop a lot of packets, the likelihood of keeping the same MR and the same CH is quite high, when using trucks, rather than vehicles, as MRs. Therefore, trucks mitigate the effect of packet loss to a certain degree, thus leading to higher cluster stability.

In Fig. 5.18, both Criticality and Density lead to higher average cluster size when compared to Fig. 5.7. Moreover, Criticality still results in larger clusters when compared to Density. Moreover, trucks decrease the average rate of CH change for both Criticality and Density, by more than 50% as shown Fig. 5.19, when compared to Fig. 5.8. The situation is similar in Fig. 5.20 and Fig. 5.21, where CH and CM durations increase more than 40 % due to the stability at the NEMO level with Trucks. Furthermore, trucks lead to increased network connectivity, as shown Fig. 5.22, when compared to Fig. 5.12.

Therefore, ASPIRE algorithms can exploit the heterogeneity of vehicles to increase the robustness of the network to channel error. This optimization leads to large clusters,
Figure 5.22: The average network connectivity as a function of channel error for 400 Vehicles and 36 Trucks with both Density and Criticality metrics. Probability of channel error is swept over 0, 0.1, 0.2, and 0.3, and Hel is swept over 1s and 2s.

infrequent CH changes, long CH and CM durations and high network connectivity, even under severely lossy conditions.
Chapter 6

Conclusion

6.1 Main Ideas

In the course of this work, we have looked into the problem of providing non-safety services for vehicles moving on a highway, while using almost no infrastructure. We presented a novel view of a Service Provider that builds an adaptive network based on local communication between vehicles and on existing 3G/LTE cellular communication, instead of dedicated fixed infrastructure. This view increases the effective throughput of the vehicular network by using the connectivity of a regular ISP only when the required information is not already cached in its adaptive network. Based on this view, we presented the Adaptive Service Provider InfrastructuRE (ASPIRE) for VANETs, a general multitier wireless ad hoc network architecture that exploits the mobility of vehicles and builds on stable vehicular clusters as network infrastructure. Using the concepts of Network Mobility, this architecture creates Mobile Networks made of groups of vehicles that move together; as one entity, along the highway. This allows vehicles with regular Mobile IP nodes to connect to the network, increases network connectivity and decreases the overhead of using the 3G/LTE cellular connectivity through caching in clusters.

To provide for the needs of the ASPIRE architecture, we also presented a novel
VANET clustering algorithm that enables nodes to adapt dynamically to mobility changes and regroup themselves in order to form stable, long-lasting clusters and Mobile Networks of vehicles. We have developed a localized version of the Network Criticality metric, which measures the robustness of a network, and used it for VANET clustering. In addition to grouping vehicles into clusters, Criticality was used to create stable Mobile Networks on top of these clusters and to reduce vehicle handoff between Mobile Networks. We have also developed maintenance algorithms based on mobility monitoring and estimation to make intelligent clustering decisions, in order to avoid unnecessary cluster head changes and keep the network nodes highly connected. Moreover, we have presented cluster management techniques that can provide the SP with the capability to maximize the resource efficiency of its network and provide users with application-dependent priorities.

The simulation results show that ASPIRE clustering algorithms using the Criticality metric lead to large clusters of vehicles, low average rate of CH change and high network connectivity, compared to the well-known Density metric. In addition, due to the efficient maintenance algorithms presented, ASPIRE also achieves high stability to various vehicle mobility conditions and high robustness to channel error. Furthermore, the high cluster stability and high network connectivity achieved, irrespective of the metric used, verify the validity of the ASPIRE architecture. Finally, we have shown how ASPIRE can exploit the heterogeneity of vehicles to improve cluster stability, provide higher connectivity and increase the system robustness to channel error.

6.2 Future Work

This work is far from being complete. The following are some ideas that could be worked on in the future.

- Comparing the performance of the ASPIRE architecture to that of other VANET architectures while running various types of applications
• Developing a greedy hop-by-hop routing protocol for the ASPIRE architecture, using link criticality as the graph weight

• Decreasing the overhead of messaging and depending more on cluster heads for Criticality calculation

• Comparing ASPIRE clustering with Density versus ASPIRE clustering with Criticality using Density as the graph weight

• Representing the ASPIRE network with Direct Acyclic Graphs (DAGs) and optimizing the broadcast of messages throughout the VANET

• Running a Publish/Subscribe Ad Dissemination system on top of the ASPIRE system, using MRs as Brokers

• Optimizing clustering decisions coupled with energy-saving decisions towards green transportation technologies

• Motivating driver behavior using P2P incentives and social networking ideas in order to improve transportation and enhance vehicular networking at the same time

• Using the effective capacity of the link (taking interference into account) as a weight metric for criticality calculation in order to optimize the resource allocation in the VANET
Bibliography


