A Content Dissemination Framework for Vehicular Networking

Ilias Leontiadis

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of the

University College London.

Department of Computer Science

2009

I, Ilias Leontiadis, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Vehicular Networks are a peculiar class of wireless mobile networks in which vehicles are equipped with radio interfaces and are, therefore, able to communicate with fixed infrastructure (if available) or other vehicles.

Content dissemination has a potential number of applications in vehicular networking, including advertising, traffic warnings, parking notifications and emergency announcements. This thesis addresses two possible dissemination strategies: i) *Push-based* that is aiming to proactively deliver information to a group of vehicles based on their interests and the level of matching content, and ii) *Pull-based* that is allowing vehicles to explicitly request custom information.

Our dissemination framework is taking into consideration very specific information only available in vehicular networks: the geographical data produced by the *navigation system*. With its aid, a vehicle's mobility patterns become predictable. This information is exploited to efficiently deliver the content where it is needed. Furthermore, we use the navigation system to automatically filter information which might be relevant to the vehicles.

Our framework has been designed and implemented in .NET C# and Microsoft MapPoint. It was tested using a small number of vehicles in the area of Cambridge, UK. Moreover, to prove the correctness of our protocols, we further evaluated it in a large-scale network simulation over a number of realistic vehicular trace-based scenarios.

Finally, we built a test-case application aiming to prove that vehicles can gain from such a framework. In this application every vehicle collects and disseminates road traffic information. Vehicles that receive this information can individually evaluate the traffic conditions and take an alternative route, if needed. To evaluate this approach, we collaborated with UCLA's Network Research Lab (NRL), to build a simulator that combines network and dynamic mobility emulation simultaneously. When our dissemination framework is used, the drivers can considerably reduce their trip-times.

Acknowledgements

A PhD is usually considered as a lonely and reclusive endeavour. However, I did not experience any of these feelings thanks to the support of numerous people. They helped me not to view my research as "work" but rather as an enjoyable and remarkable experience that I will always remember. Without their invaluable help this PhD thesis would not have been possible. Thus, my deepest gratitude goes to everyone that supported me over the last few years.

First of all, I would like to thank my supervisor Dr. Cecilia Mascolo for inspiring me with her passion and energy for research, for being always there for me, guiding me throughout this long trip. I have been very fortunate to have a supervisor that is able to both give me the freedom to pursue the PhD at my own pace and to provide guidance so that I am always on the correct track. This thesis is the result of our long discussions, numerous e-mail conversations, continuous inspiration and her constructive criticism. She has devoted so much time to me, and for this am indebted to her.

Special thanks to Dr. Paolo Costa for our thought-provoking chats. His experience and help on Content-Based-Routing were vital as they helped to steer this work towards the correct direction. I am also very grateful to Professor Mario Gerla, Professor David Rosenblum and Dr. Stephen Hailes for their ideas and support during the first years of my PhD.

Many thanks to all the people of the seventh floor for being there for me. This PhD was a very sociable experience. I will always remember our endless conversations at our desks, in the corridors and common room. I will never forget our geeky discussions in the nearby pubs that made people around us to stare in surprise. Many thanks to all new friends that I made during my stay in London, Los Angeles and Cambridge. Special thanks to my flatmate, Michalis, for putting up with my boring PhD conversations during the last years and to Liam McNamara for his suggestions and feedback. Finally, warm thanks to all my friends in Greece who I have missed all this time, but who are

always there for me, when I ever visit.

This thesis is dedicated to my parents who have supported me during all these long years of studying. As I am an only child I can understand that it was really tough for them to see me only a couple of times a year. I would like to thank them for understanding my choices and supporting me with their love and patience.

Contents

List of Figures

List of Tables

1

Introduction

During the last years there has been an increasing research interest in wireless networks due to the large variety of possible applications and the challenges of this communication paradigm. Wireless networks can be classified into two major categories: *infrastructure-based* wireless networks and *ad-hoc* wireless networks.

Infrastructure-based wireless networks usually consist of some fixed nodes connected to a wired backbone. These nodes are often called *infostations*, *basestations* or *access points*. They provide communication between the wireless nodes and they may act as gateways to other (possibly fixed) networks. Typical examples of this kind of communication are GSM networks, WiMax and wireless networks inside buildings.

On the other hand, wireless *ad-hoc networks* are characterised by the lack of such infrastructure. These networks are composed of nodes which communicate directly with each other by means of a (usually) short-range wireless medium (e.g., WiFi, Bluetooth, ZigBee, etc.). These connections form an arbitrary network topology. In order to com-

municate with distant nodes, a *multi-hop* (store-and-forward) communication paradigm is used, meaning that a message can be delivered through a number of intermediate forwarding nodes (hops). In essence, every node acts as a host and as a router at the same time.

Hybrid networks constitute another category, which combines elements from both infrastructure-based and ad-hoc mobile networks. In hybrid networks fixed infrastructure is used, where it is available, and ad-hoc communication, where it is not. Apart from communication in areas where there is no infrastructure, ad-hoc connectivity is often used to exchange local information (e.g., file sharing between people in the same room) as this is faster and shows lower latency than long-range infrastructured communication.

Thanks to the advantage of electronics, wireless devices can now be small enough to be carried by humans or vehicles. In fact, an increasing number of *pervasive* devices are now part of everyday life: mobile phones, PDAs, laptops, navigation systems, etc. In *Mobile Ad-hoc Networks (MANET)* nodes are free to move from one location to another using any kind of mobility pattern at any speed (e.g., walking, driving and flying).

In fixed networks link failures are usually exceptional, whereas in MANETs they are very common. These disconnections are, in most cases, unpredictable and lead to *intermittently connected mobile ad-hoc networks*. In fact, in most mobile networks the fundamental assumption of an existing path between the communication parties is not valid and, hence, any synchronous communication paradigm is likely to show poor performance.

Therefore, the new research area of *Delay Tolerant Networks (DTN)* has emerged where an *asynchronous* communication model is employed. These networks are also referred to as *Opportunistic Networks (ON)*. DTN protocols [Fall 03, Jain 04] provide communication in such performance-challenged environments, where continuous end-to-end connectivity cannot be assumed, by employing a *store-and-forward* message switching: fragments of a message (or the whole message) are forwarded from host to host and stored until the message reaches the destination. Data exchange occurs during *opportunistic contacts* of the hosts. The mobility patterns of the hosts and the selection of the next message carrier determine whether the messages will eventually be delivered

to their final destination.

To disseminate information to multiple hosts in the network, a basic approach such as *epidemic dissemination* [Vahdat 00] or gossip-based methods [Haas 06] can be used. In this range of protocols each host keeps a message buffer and when, as a result of movement, it comes into contact with other hosts it forwards copies of messages that they do not possess, making them new carriers. Eventually, the message will be delivered to all hosts in the network, provided that the mobility patterns allow this. Epidemic routing is a simple but effective approach, since it does not rely on information about which hosts require a message, it is just spreading the information everywhere in the network. This maximises the delivery ratio and the speed at which the information is spread, however, it also causes a considerable communication overhead. Many attempts to optimise epidemic dissemination have been made. For example, in [Xiangchuan 01] the authors employ a utility function to limit the spreading of epidemic messages and to ensure faster delivery using fewer resources.

Furthermore, the research community has devised a large variety of delay tolerant routing protocols [Shah 03, Zhao 04, Pentland 04, Lindgren 03, Musolesi 05]. These protocols exploit different mechanisms to route a message to the destination, such as statistics of previous encounters, social characteristics, or even precise mobility schedules to find the best carriers to forward messages. Finally, there have been several attempts to constrain the dissemination inside specific geographical areas with the aid of GPS receivers or other location services. This kind of communication is usually referred to as *GeoCast* [Mohapatra 04, Mauve 01] and it will be also mentioned later on in this thesis.

1.1 Vehicular Networks

Vehicular Networks are mobile networks in which vehicles are equipped with radio interfaces and are, therefore, able to communicate with fixed *infrastructure* (if existing) or other vehicles in an opportunistic way (*Vehicle-to-Vehicle (V2V) communication*).

These type of networks may have a large variety of interesting applications: First of all, for road safety, in order to provide warnings when a vehicle is approaching a red-light, to warn when a leading vehicle suddenly brakes, to co-ordinate lane merging in highways

or even to form platoons of vehicles. Secondly, applications that disseminate various information like free parking spots, traffic warnings/conditions, or information about fuel prices, local landmarks and bus times. Additionally, vehicles can be regarded as distributed sensors that collect any type of observations and report them to local base-stations (e.g., average speeds, potholes, temperature, pollution). Finally, entertainment applications can also be implemented, for instance file sharing, advertisements, voice communication with nearby vehicles or even distributed gaming.

While cellular networks can be used to offer these services, this solution can also create a number of issues. First of all, the service providers in each country impose different rules and restrictions as to what kind of data can be exchanged through their network or even what type of applications can access it, making it impossible for vehicular applications to be deployed globally (e.g., at least a per country agreement will be required). Additionally, the cost of cellular data communication is restrictively high, as it can reach a few pence per KB. Even expensive "unlimited" plans are usually capped to a few hundred megabytes per month, making large-scale communication between vehicles unfeasible. Furthermore, although 3G connections can support up to 128 Kbits/sec inside a moving vehicle, the bandwidth is shared between all users inside the cell. Even today, when 3G is not widely used, the network is swamped by traffic, resulting in very low throughput in densely populated areas [Luna 09].

Broadcast based solutions (e.g., FM RDS/TMC communication) are also very limited. First of all, the major drawback is that they do not support bi-directional communication. FM-based transmissions provide very low bit-rates and this is why they are mainly used to broadcast important information updated every 10-15 minutes. Furthermore, they provide limited coverage and require an area licence and expensive infrastructure deployment.

On the other hand, the use of *WiFi* does not require any kind of licence or any infrastructure deployment (although road-side infostations and WiFi hotspots can be used to maximise coverage and provide communication to/from the Internet). But, most importantly, local wireless communication between vehicles is becoming a reality: there is increasing interest and support for vehicular networks [Herrtwich 05, Resendes 08], led mainly by the interest to maximise road safety (i.e., to avoid vehicle collisions). First of all, the 802.11 Working Group of the IEEE is developing 802.11p or *Dedicated*

Short Range Communications (DSRC) standard [IEEE 09]. This new standard defines enhancements to 802.11a required to support Intelligent Transportation Systems (ITS) applications that support data exchange between high-speed vehicles (up to 120km/h in each direction) and between the vehicles and the roadside infrastructure. The new protocol supports high data rates (up to 27Mbps) in a relatively short range (1km). Furthermore, governments around the world actively support vehicle-to-vehicle connectivity by licensing a large (and expensive) part of the wireless spectrum for this use (5.9 GHz in USA and 5.8GHz in Europe and Japan). Additionally, major car companies like Toyota and Nissan have already developed wireless collision warning systems [Robinson 06, Nissan Mot. 09] based on DSRC. These systems alert drivers to the presence of vehicles moving too fast towards an intersection or red-light, school zones, etc. Toyota's system also generates warnings when leading vehicles in range brake suddenly (by transmitting the position, speed, direction and the break status of the leading vehicle). It is worth mentioning that Nissan plans to test its system using 20,000 vehicles in Kanagawa (a prefecture south of Tokyo) [Nissan Mot. 09]. Finally, even modern navigation systems [TomTom 09, Dash 09] include wireless interfaces (mainly to download updates when the device is taken to the owner's house). All these are indications that in the near future short-range wireless connectivity between vehicles will be available making the deployment of *V2V applications* even more feasible.

Apart from these one-hop examples, vehicles can form larger networks where infostations and vehicles can co-operate to route and disseminate information: Infostations are fixed access points that are potentially connected to the Internet. They may act as dissemination points, from where information coming from the backbone network flows towards the vehicles and vice versa. Vehicles can inter-network with each other, disseminating messages even further, practically extending the range of the infostations.

Vehicular networks may be considered as intermittently connected (delay-tolerant) networks with some unique characteristics:

- *Unique mobility patterns:* These networks are highly mobile resulting in a constantly changing network topology. Approaches that try to maintain location/route information (e.g., [Perkins 94, Jacquet 01]) cannot be applied, since this will induce significant overhead. However, the mobility patterns are more predictable than in other mobile networks, as vehicles are travelling on roads

(where the topology is already known) and because drivers follow specific mobility trends (for example, buses have predefined routes, vehicles tend to drive on major roads to reach remote areas, drivers usually select some routes with higher probabilities).

- *Communicating parties are not always aware of each other:* Information in vehicular networks may not be targeting specific vehicles (i.e., IP routing may not be applicable), but it is rather aiming to deliver information to large groups of vehicles or even just to some specific areas. Examples of this kind of information are traffic information, fuel prices and parking spots availability.

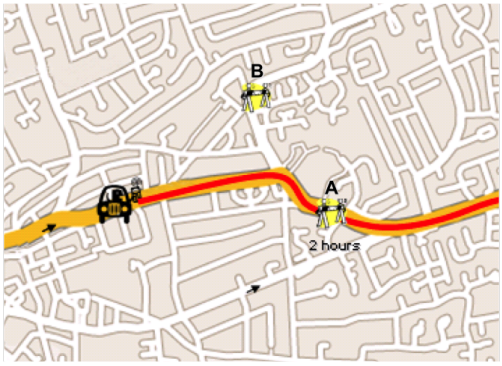
- *Information relevance:* Similarly, information should not be disseminated everywhere or to all vehicles. It is relevant only to some *areas/vehicles* based on the road-topology and the nature of the information. For example, information about an accident on a highway is not required by vehicles travelling in the opposite lane or vehicles that are beyond the accident location. The content of the information in conjunction with the mobility patterns of the vehicles and the road topology can play a vital role in determining whether the information is required or not. In the same way, vehicles usually request to download information about their current location, route or destination.

- *Large network:* Vehicular networks may consist of thousands or millions of vehicles. Any kind of approach should be scalable so as to support this volume of communication.

- *Hardware constraints:* In traditional mobile ad-hoc networks hardware constraints like power, storage and CPU are taken into account. In vehicular networks no such constraints exist, as vehicles are large enough to accommodate appropriate hardware like computers, antennas, and sensors, and these components can be powered by the vehicle's generators and batteries.

- *Varying density and penetration:* The network is so dynamic that considering just one scenario is not an option. Vehicle density can vary greatly in different locations (e.g., city, rural, highway scenarios) or even in the same location but at a different day/time-of-the-day (e.g., Sunday night vs. Monday rush hour). This calls for flexible protocols that can handle both high density and partitioned networks.

(a) TomTom's GUI. You can see the suggested route and the estimated time of arrival (ETA).



(b) When the vehicle's navigation route is known (highlighted line), we can evaluate whether the information about a location is relevant.

Figure 1.1: Navigation System

- *Availability of information:* In vehicular networks we can assume that vehicles may have access to location services (such as GPS), may know their speed and direction, be aware of the road topology (e.g., have a map database) or even access navigation information.

All these points should be considered by any routing and dissemination protocol that is targeting vehicular applications.

1.2 Navigation System

Nowadays, more and more vehicles are equipped with satellite navigation systems (NS) that are typically composed of i) a GPS receiver to identify the vehicle's location, ii) maps to navigate the driver to a specific address, point of interest or location, iii) the appropriate hardware/software to aid the driver to navigate to her destination.

These systems provide turn-to-turn navigation assistance to the driver until the vehicle reaches the destination. The driver may select her destination and preferences, and the navigation system calculates a *suggested route* from the current position of the vehicle to the final destination. An example is given in Figure 1.1(a).

To calculate the suggested route, the map of the navigation system contains statistical and historical information about speed limits, average speed, etc, and it employs a

shortest-path algorithm on the road network (e.g., Dijkstra [Sedgewick 84] on weighted graphs). Furthermore, NS provides information on the Estimated Time of Arrival (ETA) or the *Estimated Time Required (ETR)* for the vehicle until it reaches its destination.

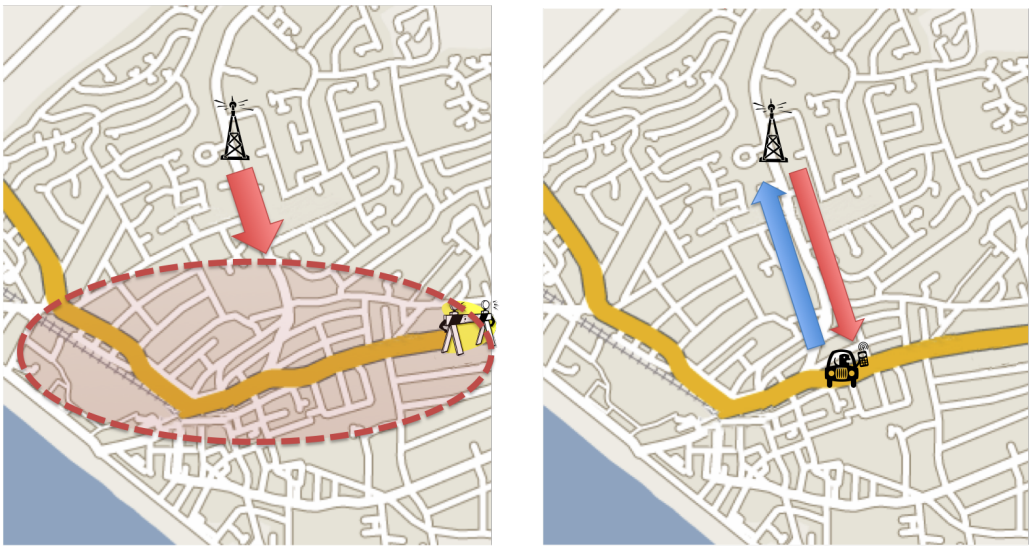
The latest NS devices support all kinds of features to make the navigation even more effective. First of all, recent devices include WiFi and allow the update of the map database and the uploading of traffic statistics, when they are in the range of the driver's WiFi network. Furthermore, some of the devices allow coarse-grained traffic updates, usually delivered by FM RDS or TMC radios, in order to help drivers to avoid traffic jams in major highway segments. Finally, some of the devices try to make the navigation assistance easier by providing a 3D representation of the road networks. It is obvious that these systems are becoming more and more sophisticated and, at the same time, affordable and, thus, they will play a key role in the design of future vehicular applications.

Apart from navigation assistance to the driver, the navigation system provides valuable information such as the *suggested route*. This information *makes the mobility patterns of the vehicles more predictable* and may be used to efficiently select the most appropriate vehicles to forward and spread messages into specific geographical locations.

Secondly, the suggested routes, in conjunction with the map database (i.e., the road topology and the available points of interest), can help us *evaluate whether the disseminated information is relevant* to a vehicle or not. For example, in Figure 1.1(b), as the vehicle's route is known (highlighted line), we can assume that the vehicle is more likely to be interested in receiving information about location A rather than information about location B. We will later examine how this information may be exploited to automatically push and pull content.

1.3 Research Problem and Thesis Contribution

This thesis focuses on the problem of building the appropriate architecture and protocols to *disseminate information* in hybrid vehicular networks. We aim to design *robust* dissemination mechanisms (i.e., improve delivery ratio) without causing much communication overhead. In our approach communication delay is considered as a secondary priority as this framework is designed to address delay-tolerant applications.



(a) Push-based Dissemination: Popular information is disseminated to all interested vehicles inside the greyed area.

(b) Pull-based Dissemination: Vehicles request custom information from a nearby infostation. The reply is then routed back.

Figure 1.2: Push v.s. Pull

First of all, we argue that it is possible to disseminate large volumes of information by using vehicle-to-vehicle and vehicle-to-infostation communication with the aid of a key tool: the navigation system. Secondly, we assert that two kinds of communication paradigms are required:

- *Push-based dissemination:* This allows applications to *Publish* information to multiple vehicles at the same time (i.e., information that concerns many vehicles like traffic information, parking spots, etc.). An example is given in Figure 1.2(a) where an infostation is pushing information about road works. The dissemination should be constrained within areas where there are vehicles interested in receiving this information. We believe that a content-based routing approach can be used that in order to deliver the information only to the affected vehicles. Finally, we will examine whether we can use the navigation system to make sure that the dissemination will be time-stable (will not fade away).

- *Pull-based dissemination:* This allows vehicles to *pull custom information* from nearby infostations. The V2V connectivity may be used to expand the range of any available infrastructure and maximise the available bandwidth. An example is given in Figure 1.2(b) where a vehicle sends out a request to the nearest known infostation and later receives a reply containing the requested data. Furthermore,

vehicles can just subscribe to receive relevant information and later, when there are matching publications, receive the required information. This kind of model is appropriate for user-specific (unpopular) data like landmark pictures, map updates, music downloads, etc.

Therefore, in this thesis we will show that both these models are required, in order to provide support for a wide range of information dissemination applications.

Furthermore, we will study whether this dissemination framework can actually help drivers to make informed decisions, by building a test-case where vehicles collect and disseminate road-traffic information in order to minimise their trip-times (i.e., by avoiding congested areas).

The contribution of this thesis is the following:

1. Examine how network protocols can use the navigation system i) to exploit the known vehicles' mobility patterns and ii) to evaluate whether a vehicle is affected by the information's content.
2. Build a novel geographical routing protocol that will be the stepping-stone to our dissemination techniques.
3. Devise a push-based dissemination protocol to disseminate and maintain popular data inside certain areas. We will also examine how we can use the content-based routing model to spread the information to areas where there are vehicles that need it.
4. Design a pull-based dissemination protocol that allows vehicles to request custom information from nearby infostations and delivers the information back efficiently, despite the fact that the vehicles are constantly moving.
5. Develop a framework architecture that incorporates all these protocols.
6. Evaluate the performance and feasibility of our approach through implementation of a working prototype and testing, using a small number of vehicles. Furthermore, a large-scale simulation-based evaluation to prove the validity of our protocols.
7. Investigate whether drivers can benefit from such a dissemination. In our test case we let vehicles collect and disseminate traffic information. We will then examine

whether such a distributed dissemination system can help the drivers to make smart re-routing decisions, in order to minimise their trip times.

In this thesis we did not look into security and privacy concerns. In vehicular networks the disseminated information is important, as it can affect driving decisions. Furthermore, if vehicle-to-vehicle communication is used to help vehicles forming platoons and performing co-operative braking (e.g., [Nissan Mot. 09]), then the disseminated information can actually raise safety issues (e.g., cause accidents). Furthermore, drivers can raise privacy concerns and may be unwilling to share information about their routes, position, etc. Security, trust and privacy mechanisms could be built over our framework to make sure that such a framework can be widely used.

These problems are addressed by various projects that are orthogonal to our approach. For example, in [Sampiget. 05, Gerlach 06, Gerlach 07b] the problem of improving privacy in location-aware applications is examined. Various measure can be taken to improve user's privacy: for example, vehicle can have randomly chosen IDs that frequently change. Furthermore, their final destination can be hidden or it can be slightly inaccurate. Numerous trust mechanisms were also devised [Serna 08, Gerlach 07a, Wang 07b] to improve cooperation and quality of the disseminated information, by building a secure recommendation system where vehicles are ranked, based on how accurate their information was in the past. Finally, security mechanisms [Papadim. 08, Haas 09] can also be enforced to ensure safety and privacy. For example, authentication mechanisms may help to identify malicious users that spread misleading information. Encryption protocols may also be used to hide private information from the drivers, etc.

1.4 Thesis Outline

The thesis is organised as follows.

- Chapter 2: In this chapter we describe a geographical routing protocol that is a key element of the protocols presented in the rest of this thesis as it allows us to i) push information to specific geographical regions, ii) make sure that the dissemination is persistent throughout the dissemination time, iii) pull information from nearby infostations and iv) route information to the backbone.

- Chapter 3: In this chapter we outline our push-based dissemination architecture and algorithms. This allows us to disseminate information that concerns many vehicles (spread out common information). Our approach i) aims to achieve persistency (disseminate the information for a long period of time) ii) focus on the dissemination only to affected vehicles/drivers.

- Chapter 4: In pull-based dissemination the vehicle can request customised information from the nearest known infostation or for a specific location. This communication enables the vehicles to pull personalised information by exploiting nearby infostations without having to be in direct contact with them.

- Chapter 5: In this chapter we describe our prototype implementation using Microsoft .NET and Microsoft MapPoint (as the navigation system). We also illustrate our small-scale experiments using a few real vehicles.

- Chapter 6: To prove the validity of our protocols, a large number of vehicles is needed. In this chapter we present our simulation implementation using the Omnet++ event based network simulator. To make the simulation as realistic as possible, we used mobility traces that are based on real maps for different scenarios.

- Chapter 7: In this chapter we use a simple case study to evaluate whether such a dissemination system may actually help the drivers. Our goal is to evaluate whether such a dissemination can help (or not) drivers to improve their trip times.

- Chapter 8: Finally, this chapter concludes this thesis and provides some further discussion.

You can find additional information in some of our papers. First of all, details about our geographic routing protocol can be found in [Leontiadis 07b]. Our push-based approach is further analysed in [Leontiadis 07c, Leontiadis 09b] whereas in [Leontiadis 10a, Leontiadis 10b] we present our pull-based mechanisms. In [Leontiadis 09a] we demonstrate our real implementation and our field testing whereas in [Leontiadis 07a] we present our framework architecture. Finally, in [Leontiadis 09c] you can find details about our case study concerning the impact of an ad-hoc dissemination on the drivers' trip times.

2

Geographic Opportunistic Routing

In this chapter we present *GeOpps* a Geographic Opportunistic routing protocol for vehicular networks. This is a key component of our framework for the following reasons:

- First of all, in vehicular networks information is often relevant to specific geographic regions (e.g., the area of an accident). Additionally, vehicles may route information to the location of known gateways so as to reach other networks (e.g., route information to the nearest known infostation). A routing protocol will allow this kind of multi-hop communication between locations.
- Furthermore, a geographic routing protocol is required to push out information to specific areas. This may happen in two steps: i) route the information inside the intended area and ii) disseminate it around (More details in Chapter 3).
- Moreover, as we will see in Chapter 3, GeOpps will ensure that the dissemination will be kept alive for a period of time (i.e., while it is relevant) in cases where no infrastructure is available.

- Finally, vehicles may pull information from the nearest infostation using geographic routing. We will later examine (in Chapter 4) how we can use concepts inspired by GeOpps to perform these tasks.

2.1 Motivation

Vehicles can be considered as mobile sensors that gather all kinds of information (e.g., traffic condition, potholes, images, pollution). This is quite a realistic assumption and other systems build on it (e.g., CarTel [Bychkovsky 06]). Afterwards, vehicles can dispatch this information to a central location for processing through the nearest known infostation. And since constant connectivity between vehicles and infostations cannot be assumed (especially in remote areas), other vehicles need to act as data mules, carrying information from the vehicle to the infostation and vice versa. A geographic routing protocol that is designed especially for vehicular networks is, therefore, required to perform this task.

Similarly, a central decision point (e.g., a highway agency) can generate traffic warnings concerning specific road segments and suggest alternative routes to vehicles that are approaching them. To warn the drivers, the traffic management centre has to dispatch this information to the vehicles in these areas. Initially, warnings are sent to the nearest infostation and, from here, they need to be routed to the affected road segments using the vehicular network. Upon reaching the area, local message dissemination techniques (like constrained flooding or localised epidemic) can be employed to spread the information to nearby vehicles. Consequently, a vehicular routing protocol can be utilised to route the information *from* and *back* to the vehicles from an infostation.

However, designing a routing protocol for vehicular networks is not a trivial task. There are many issues that prohibit the re-use of traditional methods. For example, due to the high mobility, the network topology is constantly changing. This means that static routes cannot be maintained due to the high message overhead required to frequently update them and because the discovered paths rapidly become obsolete. Therefore, proactive routing protocols like DSDV [Perkins 94], OLSR [Jacquet 01], STAR [Garcia-Aceves 99], WRP [Murthy 95] and TBRPF [Ogier 04] are unpractical.

Reactive protocols are more suitable, as in most cases they do not require any kind

of routing table maintenance: these protocols calculate a route only when a message needs to be routed and are, thus, more suitable for highly mobile scenarios.

Furthermore, since vehicular networks are often intermittently connected, routing protocols should be robust to face link failures and no end-to-end connectivity. A *delay-tolerant* approach is far more suitable.

Finally, as we described in our scenario above, in vehicular networks information is often relevant to geographic areas rather than to specific hosts. Therefore, a *geographic routing* protocol seems more appropriate for such kinds of communication.

To address all these properties we designed a novel geographic, delay-tolerant routing algorithm that exploits the availability of information from the navigation system in order to opportunistically route data to a geographic location. We take advantage of the vehicles' NS suggested routes to select vehicles that are likely to deliver the information closer to its final destination. This protocol will become a key element of the dissemination mechanisms that we will present in the following chapters.

2.2 GeOpps

As we briefly described before, when the driver selects her destination, the navigation system calculates a *suggested route*, starting from the vehicle's current position. Apart from supplying the route (as a list of road segments), the navigation system provides information about the Estimated Time Required (ETR) for the vehicle to reach each intermediate intersection: for each intersection a tuple <*intersection id, location, ETR*> is calculated.

This information is extremely important because it can be used to predict the vehicles' mobility patterns. As we will examine in detail later, knowing a vehicle's mobility schedule can be used to evaluate whether it is a strong candidate to deliver a message to a geographic location.

Our approach largely relies on forecast routes available in the navigation systems. This implicitly assumes that users are co-operative and willing to insert their destination. One might argue, however, that this assumption is only partly verified in practice as users tend to avoid using navigation systems for known routes (e.g., when going to

Figure 2.1: Calculating the Nearest Point (NP) to message's Destination (D).

their work places). Nevertheless, we expect the drivers will have an incentive to insert their destination in the navigation system as this will automatically allow them to use the extra services provided by the ad-hoc 'communication which are of utmost importance even for daily routes (e.g., as we will see in Chapter 3 the drivers will automatically start receiving news about their route, critical information concerning traffic congestion, warnings about accidents, etc.). Furthermore, systems such as Predestination [Krumm 07], that can automatically detect the drivers' destination based on general driving trends and historical data for each driver, can be used to supplement or substitute the navigation system information. Finally, vehicles that do not have navigation information can still use GPS information as a fail-safe mechanism (more details about this in section 2.2.4).

Lets assume that we have a message m for a certain geographic location D (e.g., D can be the geographic location of a known infostation). GeOpps aims to deliver the message from the current location to D by selecting vehicles that are likely to carry the message as close as possible to D. An example is shown in Figure 2.1. Briefly, our routing protocol selects the next message carrier with the following mechanism:

- Vehicles periodically broadcast a 1-hop advertisement that contains their suggested navigation routes. This information is taken directly from their satellite navigation system. For example, when a vehicle a advertises its route, it informs the 1-hop neighbours about its current position and about the fact that it is cur-

rently intending to drive towards a specific geographic destination using a route R. If no such information is available they just advertise their current GPS position.

- When a's advertisement is received by a neighbour b, then it uses the advertised information to evaluate if a will be a good carrier for the message m: more specifically, it examines how close (in terms of driving time) a will be driving to m's destination D. To formalise this evaluation we define a utility function that expresses the *minimum estimated time* that m would need to be delivered if a becomes the next carrier. The value of this utility depends on a's route R, the destination of the message D and the road topology (the map) (More details in Section 2.2.2).

- Finally, b compares a's utility with its own: the vehicle that has the best utility is given the message. In our case *smaller values* are considered better (as this means that the host can deliver the message sooner to its final destination).

We will now provide more details about the two steps involved in calculating whether a vehicle is appropriate to deliver a message to D: i) the calculation of the nearest point and ii) the calculation of the utility value.

2.2.1 Calculation of the Nearest Point

Let us assume that a vehicle has calculated a suggested route R to its destination. When this vehicle is given a data packet for a location D, it can calculate the *nearest point*[1] NP on its route, compared to D. In other words, it calculates the point on its driving path that will be the nearest to the destination of the packet. Figure 2.1 illustrates an example: The dotted line is the minimum driving distance between the route of the vehicle and D.

To calculate this point, for each intersection I on the vehicle's suggested route R the navigation calculates the driving time between I and D. The intersection that minimises this time is selected. In Figure 2.3 (Part B) you can find the pseudo-code for this operation whereas in Figure 2.2 you can find the definitions required.

To find the nearest point NP, the actual driving-distance (dashed line in our example) or the Euclidian (straight-line) distance from NP to D may be used (Fig-

[1] in terms of driving time

Variables

- self: node's own id.
- $\mathcal{M}$: the message buffer. It contains the messages to be routed to D.
- $(\bar{x}, \bar{y})$: neighbour's current position.
- $\mathcal{R}$: node's route, expressed as a ordered list $\langle (I_1, t_1), (I_2, t_2), \ldots, (I_r, t_r) \rangle$ of intersection points I_i and (expected) arrival time t_i.
- $\mathcal{DT}_{\mathcal{I}}$: driving time required between Intersection I and the message's. destination D. This calculation is performed on the map by the navigation system.
- $\mathcal{D}_{\mathcal{I}}$: Euclidean distance between Intersection I and the message's destination D.
- α: weight putting more emphasis on distance.

Messages

- CONTROL$< ID, R, x, y >$: the route advertisement of a node n. It contains:

  ID: n's ID.

  R: the expected route of n (taken by the navigation system).

  x, y: n's current position.
- DATA$< ID, Type, TTL, D, Content >$: this is a data packet stored in $\mathcal{M}$. When a better carrier is found it is forwarded n. It contains:

  ID: a unique ID for this message.

  $Type$: the type of the message. In this section $Type$ = Route(route to location).

  TTL: the time to live. When this expires the message is discarded.

  D: the final geographic destination of the message.

  $Content$: the payload of the message (the data).

Functions

- **send**$(msg) \rightarrow n$: send a unicast message to a neighbour n. This is used to send a DATA message.
- **broadcast** (msg): broadcast message to all 1-hop neighbours. This is used to send a CONTROL message (an advertisement to all our neighbours).
- **computeNP**(R, x, y): find the point (NP) on route R which is closest to the location (x, y)
- **computeETR**$(A(x_a, y_a), B(x_b, y_b))$: compute the Estimated Time Required (ETR) to opportunistically move a message from location $A(x_a, y_a)$ to location $B(x_b, y_b)$. To estimate this value we use Dijkstra on the map. We consider the map as a weighted graph (weight = time required between intersections, usually based on speed limits) and we run Dijkstra's algorithm to find the shortest path and the estimated time required.

Figure 2.2: Pseudo-code definitions for the GeOpps algorithm.

ure 2.4, Part D). The first technique is more CPU intensive because it requires running a weighted shortest path algorithm (already implemented in the NS) from every intersection along the suggested route of the vehicle. The second method is less precise (because it assumes that smaller straight-line distances result in shorter driving times, which is usually the case), but it is much faster.

Part A. 1-hop Route Advertisement: Invoked periodically or when the vehicle's route changes

NeighbourAdvertise()

1: $x, y \leftarrow$ **GPS.GetCurrentPosition**
2: $R \leftarrow$ **NavSys.GetRoute**
3: $CONTROL \leftarrow <\text{self}, R, x, y>$
4: **broadcast** (CONTROL)

Part B. Calculate Nearest Point: Invoked by ReceiveAdvertisement()

computeNP($R, <\bar{x}, \bar{y}>, D$**)**

1: $NP \leftarrow <\bar{x}, \bar{y}>$
2: $MinTime \leftarrow$ **computeETR**(NP, D)
3: **for all** $I \in R$ **do**
4: $\quad DT_I =$ **computeETR**(I, D)
5: $\quad$ **if** $DT_I < MinTime$ **then**
6: $\quad\quad MinTime \leftarrow DT_I$
7: $\quad\quad NP \leftarrow I$
8: **return** NP

Part C. Receive 1-hop CONTROL: Invoked when a route advertisement is received by a neighbour

Receive $CONTROL < n, \bar{R}, \bar{x}, \bar{y} >$

1: $x, y \leftarrow$ **GPS.GetCurrentPosition**
2: $R \leftarrow$ **NavSys.GetRoute**
3: **for all** Messages $m \in \mathcal{M}$ where $TYPE = Route$ **do**
4: $\quad D \leftarrow m.D$
5: $\quad NP_{\text{self}} \leftarrow$ **computeNP**$(R, <x, y>, D)$
6: $\quad U_{\text{self}} \leftarrow \alpha \cdot$ **computeETR**$(<x, y>, NP_{\text{self}}) +$ **computeETR**(NP_{self}, D)
7: $\quad NP_n \leftarrow$ **computeNP**$(\bar{R}, <\bar{x}, \bar{y}>, D)$
8: $\quad U_n \leftarrow \alpha \cdot$ **computeETR**$(<\bar{x}, \bar{y}>, NP_n) +$ **computeETR**(NP_n, D)
9: $\quad$ **if** $U_n < U_{\text{self}}$ **then**
10: $\quad\quad$ **send**$(m) \rightarrow n$
11: $\quad\quad$ remove m from $\mathcal{M}$

Figure 2.3: Pseudo-code of the GeOpps algorithm.

Part D. Faster method to calculate Nearest Point: Invoked by ReceiveAdvertisement()

computeNP_Simple($R, <\bar{x}, \bar{y}>, D$**)**

1: $NP \leftarrow <\bar{x}, \bar{y}>$
2: $MinDistance \leftarrow$ **EuclideanDistance**(NP, D)
3: **for all** $I \in R$ **do**
4: $D_I =$ **EuclideanDistance**(I, D)
5: **if** $D_I < MinDistance$ **then**
6: $MinDistance \leftarrow D_I$
7: $NP \leftarrow I$
8: **return** NP

Part E. Faster method to handle received advertisement: Invoked when a route advertisement is received by a neighbour

ReceiveAdvertisment_Simple$(CONTROL < n, \bar{R}, \bar{x}, \bar{y} >)$

1: $x, y \leftarrow$ **GPS.GetCurrentPosition**
2: $R \leftarrow$ **NavSys.GetRoute**
3: **for all** Messages $m \in \mathcal{M}$ where $TYPE = Route$ **do**
4: $D \leftarrow m.D$
5: $NP_{\text{self}} \leftarrow$ **computeNP_Simple**$(R, <x, y>, D)$
6: $U_{\text{self}} \leftarrow \alpha \cdot$ **computeETR**$(<x, y>, NP_{\text{self}}) + \frac{\textbf{EuclideanDistance}(NP_{\text{self}}, D)}{AverageSpeed}$
7: $NP_n \leftarrow$ **computeNP_Simple**$(\bar{R}, <\bar{x}, \bar{y}>, D)$
8: $U_n \leftarrow \alpha \cdot$ **computeETR**$(<\bar{x}, \bar{y}>, NP_n) + \frac{\textbf{EuclideanDistance}(NP_n, D)}{AverageSpeed}$
9: **if** $U_n < U_{\text{self}}$ **then**
10: **send**$(m) \rightarrow n$
11: remove m from $\mathcal{M}$

Figure 2.4: Pseudo-code of the GeOpps algorithm (faster method).

2.2.2 Utility Function

When a vehicle encounters one or more others (contacts), the NS has to evaluate if it should keep the message m or forward it to one of the neighbours. To make this assessment a *utility function* U_v is computed for each candidate vehicle v. The utility function U_v represents an estimation of the *minimum time* that a message would require to reach its destination D if we let v carry it until its nearest point NP. Therefore,

Fail-Safe Method. Invoked when the vehicle and all known neighbours will not drive closer to D.

FailSafe$(CONTROL < n, \bar{R}, \bar{x}, \bar{y} >, m, D)$

1: $MyPos \leftarrow$ **GPS.GetCurrentPosition**
2: $NeighbourPos \leftarrow < \bar{x}, \bar{y} >$
3: $MyDistance \leftarrow$ **EuclideanDistance**$(MyPos, m.D)$
4: $NeighbourDistance \leftarrow$ **EuclideanDistance**$(NeighbourPos, m.D)$

5: **if** $NeighbourDistance < MyDistance$ **then**
6: **send**$(m) \rightarrow n$

Figure 2.5: Pseudo-code of the GeOpps fail-safe algorithm. Note that instead of Euclidean distance we can also use the driving time to D (if it is available)

smaller values are considered better as this intuitively means that this vehicle's route can help to deliver m faster to its final destination.

More formally, after calculating the nearest point, the NS may use the map to calculate the estimated time required (ETR) for the vehicle v to drive to NP. Similarly, it can also calculate the *estimated time that a vehicle would need to drive from NP to m's destination D* (i.e., within the range of the final receiver). The sum of these two values is an indication of how much time is required for m to be delivered if vehicle v carries it until NP (see Figure 2.1 for a graphical representation). Therefore:

$$U_v(m) = \alpha\text{ETR to NP} + \text{ETR from NP to D}$$

Clearly this value mainly depends on how close NP is compared to D (how close the evaluated vehicle v will drive to D). However, other factors like the road topology and the vehicle's speed also matter (as the utility is taking into account estimated driving times between locations).

For example in Figure 2.6, the utility value for vehicle b will be lower than the value of a because the time required to drive from P_1 to NP_a and then to D is higher than the time required to drive from P_1 to NP_b and then to D.

This calculation assumes that when v arrives at NP there will be another vehicle that can carry the message to its final destination D. Consequently, U_v represents an

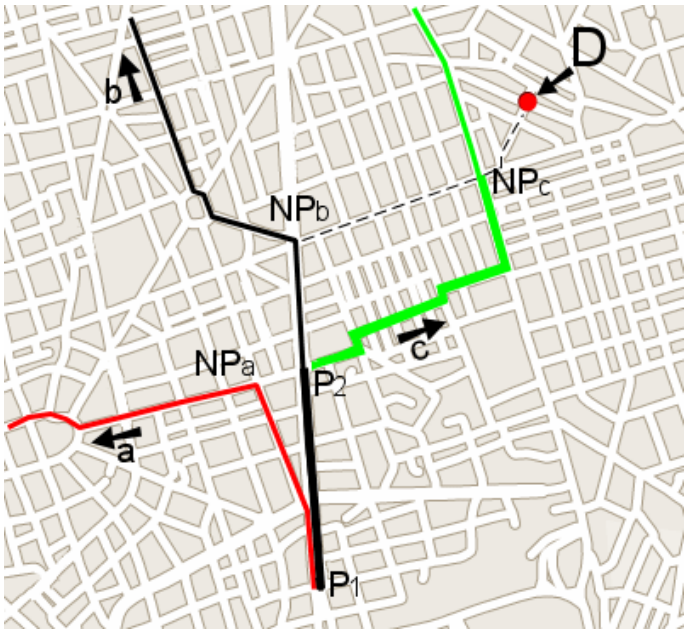


Figure 2.6: Example of comparison of different routes from a message's Destination (D).

under-estimate of the time required to deliver the message. For this reason, this utility represents the *Minimum Estimated Time of Delivery* (METD) for this message.

The parameter α (values $0 < \alpha < 1$) is used to further favour vehicles that actually deliver the message closer, no matter how slowly they are driving to NP. For example, if $\alpha = 0$ the only thing that matters for the utility is how close to D is NP and not how fast is the route that v has selected to drive towards NP.

In Figure 2.3 (Part C) the pseudo-code for the utility calculation is given. When a CONTROL message is received by a neighbour n, the vehicle evaluates whether each message m in its buffer M should be forwarded (line 3). Then it calculates and compares its own utility to n's (lines 5-11).

Finally, if the Euclidian distance is used (to reduce processing time), we can further simplify this utility function by using the straight-line distance between NP and D (Figure 2.4 Part E):

$$U_v = \alpha\text{ETR to NP} + \frac{\text{Distance Bet. NP and D}}{\text{Average Speed}}$$

This method is considerably faster than calculating a route from NP to D on the map. However, it is less accurate as it only provides an approximation: it only considers the Euclidian distance between NP and D, whereas before we calculated the

estimated driving time using the map. The *Average Speed* can be considered as a weight (similar to α) that puts emphasis on distance (i.e., select a vehicle that is going closest to the destination no matter how much time it takes) or delay (i.e., select a vehicle that might not be going that close, but it is getting to NP faster). In fact, the parameter α is not required any more (as the average speed can act as a weight) but it is kept for compatibility between the two methods (i.e., so that the navigation system can dynamically switch between algorithms based on its load).

2.2.3 Routing Algorithm

The main step of the algorithm is to keep looking for vehicles that can potentially deliver the message earlier (i.e., vehicles that further minimise the utility). Somehow then, information about the routes of the various cars needs to be exchanged among the vehicles. The algorithm follows these steps:

1. Vehicles periodically broadcast their suggested routes (Figure 2.3 Part A).
2. 1-hop neighbours, calculate the Minimum Estimated Time Of Delivery ($METD$) for the messages that they currently keep (Figure 2.3.C or Figure 2.4 Part E, lines 1-8).
3. The current carrier either keeps the message (if it has the lowest $METD$) or forwards it to the neighbour with the lowest value. (Figure 2.3 Part C or Figure 2.4 Part E, lines 9-11)
4. This process is repeated until the message arrives at its destination or it expires.

For example, in Figure 2.6, at point P_1 a vehicle polls vehicles a and b for their $METD$ values. These vehicles calculate the nearest point that they will get to D (NP_a and NP_b). Vehicle b becomes the next packet carrier. As b travels to its destination, it keeps looking for other vehicles that have even lower $METD$ values. At point P_2, it encounters vehicle c that is going even closer to the destination and, therefore, it forwards the packet to c. Notice that the packet never reached NP_b.

An interesting side-effect that we noticed is that when a large group of vehicles have the same NP (e.g., a part of their route that contains NP is the same) then the packet is forwarded to the leading neighbour because it reports smaller $METD$. Therefore, we

noticed that when the density is high, the packets travel much faster than the mobility flow of vehicles.

2.2.4 Special Cases

This protocol exploits navigation information (e.g., suggested route, ETR) to opportunistically select a neighbour that is estimated to get closer and faster to the destination of the packet. However, there are some assumptions concerning the accuracy of this information.

We have to consider what happens in cases where the drivers do not follow the suggested route. When a driver deviates from the route, its navigation system automatically recalculates an alternative route and ETR. At every contact, the NS always uses the latest utility estimation and thus, this includes any deviation. There is also the case where a vehicle is ignoring the calculated route. Most of the existing navigation systems will automatically cancel a route if the driver misses a number of turns or deviates completely from the suggested path. This behaviour can also be detected by observing a sequence of missed turns: the solution used here is to just ignore the navigation information for this vehicle (just use the GPS). Furthermore, the NS can constantly evaluate the driver's behaviour in order to predict how likely he/she is to follow the suggested route.

Additionally, we should also consider vehicles that stop/pause their trip. If the driver switches off the engine, the system will forward all the messages to any neighbouring vehicle. In case the vehicle stops for a long time without switching off the engine (in our implementation more than two minutes) the NS forwards all the messages to any neighbour.

Finally, notice that although we would prefer most of the vehicles to programme their itinerary in advance (or doing that automatically using a system such as Predestination [Krumm 07]), GeOpps does not require all the vehicles to have calculated routes (e.g. it does not require all the drivers to indicate their destination). Source vehicles may begin routing the packets using a greedy algorithm until the packet contacts a vehicle that has a calculated route that can get them closer than the current position. Similarly, if the vehicle arrives at the nearest point (NP) and no better carrier is found we need to forward the message as the vehicle will now start to drive away

from the message's destination. To solve this issue we use the greedy algorithm as a fail-safe method: the message will be routed to any vehicle closer to the destination until at least one neighbour is found with navigation information that leads closer to the message's destination than the current position. Effectively, we can only exploit navigation information when this is available. Figure 2.5 provides the pseudocode for this calculation.

2.3 Related Work

A number of existing geographic routing protocols are available [Mohapatra 04, Mauve 01]. One widely used protocol is *Greedy Forwarding*: In greedy packet forwarding when an intermediate node receives a packet, it forwards it to a neighbour that is nearest to the geographic location of the recipient. However, greedy packet forwarding does not guarantee delivery: there are cases where there is a path between the sender and the destination where greedy routing fails to deliver the message because the message reaches a local minimum. There are many algorithms proposed to solve that problem: 2-hop greedy routing, alternate greedy method, disjoint routing [Bose 01, Giordano 01]. The most well-known greedy algorithm with guaranteed delivery is *GPSR* [Karp 00]. GPSR uses greedy packet forwarding until it reaches a local maximum. When this happens it switches to recovery mode where it uses a planar graph and the *right hand rule* to forward the message until it reaches a node that can use greedy forwarding. GPSR faces some problems when there are obstacles and node localisation errors. These can introduce the risk that the planar sub-graph used by GPSR's perimeter mode may not be connected.

However, these protocols have not been specifically designed for vehicular networks and are not suitable for a number of reasons [Fubler 03]; in these networks, the topology is constantly changing but in a somewhat predictable way (e.g., cars move on roads). Furthermore, vehicles tend to move in clusters towards a specific direction, creating networks that might be not always connected (i.e., there is no end-to-end connectivity). Therefore, a *geographic* but also *delay tolerant* approach is needed.

There are a number of existing delay tolerant routing protocols [Shah 03, Zhao 04, Pentland 04]. These protocols exploit different mechanisms to route a message to the destination such as statistics of previous encounters or collocation probabilities. How-

ever, none of these approaches apply directly to vehicular networks: GeOpps takes advantages of the map and mobility patterns given by the navigation system to accurately predict vehicles' mobility patterns. Moreover, projects that employ vehicular DTN routing protocols like CarTel [Bychkovsky 06], DieselNet [Burgess 06], Drive Through Internet [Ott 05], FleetNet [Zhao 04] are available. These systems however do not consider geographic routing as they are just aiming to deliver a message to a certain vehicle ID. Geographic DTN protocols like GeOpps are more appropriate for vehicular networks as information is more often related to geographic locations than to specific vehicles.

Although DTN protocols have higher delivery ratio, when a more rapid communication model is required, a greedier approach may be used to minimise delay in exchange for higher communication overhead or lower robustness. In [Skordylis 08], the authors present an algorithm that uses traffic statistics to decide whether to use DTN forwarding or more aggressive routing algorithms, to meet a given delay threshold with the lowest possible communication overhead. We believe that such an approach can be used together with GeOpps.

Moreover, there are some attempts to design geographic routing protocols for vehicular networks. For example, Move [Lebrun 05] uses the relative direction of the vehicles (angle) to determine if the vehicle can potentially carry the information to the destination. Vehicular Greedy [Zhao 04, Fubler 03] tries to improve GPSR in order to work in vehicular networks. In [Lochert 05b] the authors use maps to greedily route a message from intersection to intersection until it reaches its destination. However, with respect to these works, GeOpps offers significant performance benefits because it exploits information from the navigation system to efficiently route packets (more information can be found in Chapter 6).

Finally, in VADD [Zhao 06], the authors designed an algorithm that, at each intersection, estimates the correct direction in which a packet should be forwarded using a combination of map and GPS data. More specifically, at each intersection the vehicle evaluates which direction is the optimal for the data to flow in order to reach its destination. This is based on historical information (e.g., density, average speed of vehicles, etc.) and the road topology. In essence, at each intersection VADD selects the next road segment where the packet should be forwarded in order to minimise delivery delay.

Afterwards, it selects a vehicle on the selected road segment, or a vehicle that is going towards it (if available) as the next message carrier. GeOpps, on the other hand, is not evaluating to which direction the message should be sent, but it is evaluating where the existing neighbours will be driving so as to identify carriers that will be driving closer to the destination, thus providing more robust choices (it only evaluates the paths of existing neighbours). Although VADD is a good example of using the map to route information in vehicular networks, its evaluation is left as future work.

2.4 Conclusions

In this chapter we have illustrated GeOpps, an opportunistic geographic routing algorithm. The main contribution of this protocol is that it exploits the information that is available in modern vehicles to efficiently select appropriate packet carriers. More specifically, navigation information (i.e., the suggested route), together with the map database, are used to find carriers that are likely to deliver a message to its destination. We designed a utility function that expresses how appropriate a vehicle is to perform this task. Finally, we designed a two-phase protocol where vehicles advertise their navigation information and receive from their neighbours the messages that they are likely to deliver.

In the following two chapters we examine how we can use GeOpps to disseminate information. In Chapter 5 we will evaluate the performance of this method in a real implementation and in Chapter 6 we will evaluate the protocol's performance in a large scale simulation.

3

Push-Based Information Dissemination

In push-based dissemination popular information is broadcast to the network so that multiple hosts can receive it. This communication paradigm allows information to be published in a specific geographic area (e.g., publish an accident warning to all vehicles on a highway). Interested vehicles just receive it when they happen to be within the communication range of a host that already holds a copy (i.e., no explicit request is made to some central server). Examples of this kind of information are traffic updates, available parking spots and warnings (e.g., accidents, road closures).

There are many possible ways in which information can be disseminated in such a setting:

- *Flood-based:* This is the simplest communication paradigm where every host re-broadcasts any received message. There is no message buffer and, thus, only nodes that are within the same connected partition receive the message. This results in quickly spreading the information to every host of the network (assuming that

the network is not partitioned at the time of the dissemination). However, this also leads to a significant communication overhead.

- *Epidemic dissemination:* Epidemic Routing protocols such as the one introduced by A. Vahdat and D. Becker [Vahdat 00], employ *gossip* to deliver the message to all hosts. In these protocols messages are stored in a buffer until their expiration time is reached or until the buffer is full. When two hosts get within communication range, they exchange the messages that they have not previously received. These protocols are specialised for partially disconnected networks: Epidemic routing can eventually deliver a message even if there is no connected path between the source and destination at any given point in time (provided that mobility patterns allow this).

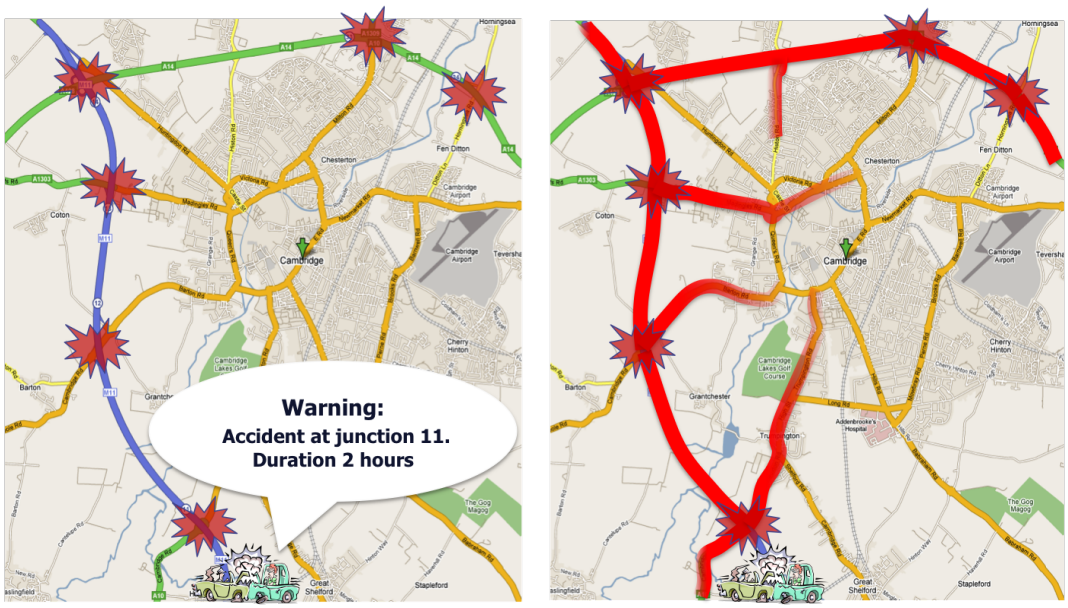
- *Geographic dissemination:* In location based broadcast/multicast, a node sends a message to all the nodes within a geographical area. This kind of communication is referred to as *GeoCast* [Mohapatra 04, Mauve 01]. There are two main approaches on how to broadcast a message to all the nodes in a region: flood-based and route-based.

  In *flood-based geocast*, a *forwarding zone* is defined between the sender and the destination area. A controlled flood is performed in the forwarding zone in order to deliver the message: only nodes in that area will forward the message to their neighbours [Ko 98, Liao 00, Camp 03].

  In *route-based geocast*, the message is first routed to any node inside the geocast region (e.g., using GPSR [Karp 00]). When the message arrives inside, a localised flooding is performed to deliver the message to all the nodes that are currently inside. Examples are GeoTora [Ko 00] and GeoNode [Imielinski 99].

- *Content-based routing:* Content-based routing (CBR) differs from classical dissemination paradigms as messages are routed based on their *content* rather than their destination address or a geographic location. This form of implicit, multi-point communication fosters a high degree of decoupling, since the communicating parties are not necessarily aware of each other, and can therefore change dynamically without affecting the rest of the system.

In the remainder of this chapter we will motivate our push-based mechanisms, analyse our system architecture and provide details about our dissemination protocol.



(a) Example of an event and the dissemination areas. The indicated areas are the ones where vehicles should be notified about the accident.

(b) Distribution of the subscribers. Most of the vehicles that are interested will be driving on road segments that can lead to the accident (the indicated areas).

Figure 3.1: Our scenario.

3.1 Motivation and Scenario

In vehicular networks, information about events like traffic warnings, accidents and road closures require a dissemination scheme that takes into account some unique requirements:

- Information usually concerns a specific *Point Of Interest* (POI). This can be the location of an accident, a parking spot, diversion etc. An example is given in Figure 3.1(a) where POI is "Junction 11". The POI of the publication offers significant information that should be taken into account in order to spread it appropriately. For example, an accident on a highway should be spread to all the vehicles approaching, miles before the accident. However, a similar publication about an accident on a *small* urban alley should not be disseminated more than a few meters away. Clearly, our dissemination protocol should take into account the fact that *information is relevant to specific geographic areas* and focus on these areas.

- Only vehicles that could be affected by the information should receive it. For

instance, an accident warning should reach only those vehicles that might be delayed by it. This, of course, depends on the location of the accident (the POI), the vehicle's destination (the suggested route) and the road topology: as you see in the example of Figure 3.1(b) only the vehicles on the highway, or near the ramps of the highway leading to the accident, are those that are likely to need the publication. As we move away from these areas the interest about this publication fades away. Therefore, we believe that a *content-based* approach is required to determine whether the vehicle is affected or not by the publication's content.

- Information should be *spatio-temporal* (location and time) persistent. In vehicular networks information is valid for a time period. For instance, drivers should be notified about an accident until it clears (2 hours in our example in Figure 3.1(a)). Due to mobility and varying density we need mechanisms to ensure that the information will be persistently propagated while it remains relevant so that new drivers in the area will be notified.

Our aim is to design an appropriate dissemination scheme that takes into account these requirements (i) location/geographic aware, ii) content aware, iii) spatio-temporal persistence. To address these requirement we are presenting a protocol for *persistent content based dissemination in vehicular networks*. This protocol enables applications to:

- *Publish* messages to geographical locations by first sending them to the relevant areas either directly (if infrastructure is available) or using GeOpps (Chapter 2).
- *Store* master-copies of the publication at *key locations* inside the dissemination area using a combination of infostations (if any) and vehicles. We call these locations *homeZones*. These master-copies, called *replicas*, are maintained throughout the dissemination time to ensure that they will not fade away. For example, in Figure 3.1(a), we would like copies of the publication about the accident to be maintained near the highway entrance ramps for 2 hours. When no infrastructure is available, we will examine how we can keep these replicas near their homeZones.
- *Deliver* the messages to *subscribers* (i.e., vehicles that are affected by the message) when they are driving through the homeZones. Additionally, further *spread the*

dissemination in areas where there is a high concentration of subscribers. An example is shown in Figure 3.1(b) where the dissemination is spread on the entire highway and the main roads leading from the city to the highway, as these are the locations where most of the vehicles, that are interested in this accident, can be found.

To address these requirements, we use the navigation system together with a content-based routing communication paradigm. As we discussed in Chapter 2, the navigation system can be used to efficiently select carriers to forward publications to the affected areas.

Furthermore, *the suggested routes can be used to infer interests* in order to automatically filter only information relevant to the driver. For example, the NS can automatically subscribe to receive traffic warnings that affect the suggested route, to receive fuel prices from nearby fuel stations when the vehicle is running out of petrol, or to receive free parking notifications concerning the vehicle's destination (*automatic subscriptions*). However, a user may also be allowed to insert specific subscription interests, which are not automatically calculated, e.g., information on nearby restaurants or hotels (*custom subscriptions*).

Clearly, the content of the notification together with the navigation information and the driver's interests can be used to route the information to the vehicles that need it.

3.2 Notification Dissemination Architecture

In this section, we examine how we can use the content-based communication paradigm to disseminate information in vehicular networks. Before getting into more details we will clarify our terminology:

POI - POI (Point Of Interest) refers to the area that the publication concerns (e.g., the location of an accident).

Persistence Area - The area where the message persistence is enforced (time-stable notification).

Figure 3.2: Information should be disseminated in the persistence area throughout the dissemination time. Black dots are the locations of the homeZones. In each homeZone a notification replica will be maintained. Interested vehicles driving through these locations should be notified. The information can further spread opportunistically outside the persistence area.

Implicit Subscriptions - Subscriptions based on the navigation information of the vehicle.

Custom Subscription - Driver/application specific subscriptions.

Publication Message - A copy of the message which should be disseminated only to subscribers.

Replica - A *master copy* of the publication. Replicas are maintained in key locations in the persistency area to inform incoming subscribers.

HomeZone - The geographic location where a replica should be kept.

Later in this chapter we will provide more details about these terms.

3.2.1 Primitives

Our goal is to design appropriate primitives that will aid the application developers to take advantage of the vehicle's navigation information to push out information. From a publisher's point of view, the information should be time and location persistent. Therefore, the publisher needs to specify in which areas the publication should be persistently disseminated. In this *Persistence area* our framework will create and maintain

a number of *Replicas* in order to keep the dissemination alive. Furthermore, the publisher defines the time interval during which the publication is relevant. Therefore, our publish primitive is:

`publish(message, persistenceArea,` T_{start}`,` T_{end}`)`

where:

- `message` is the body of the publication to be delivered. The message content will be combined with the navigation information to determine if a vehicle is interested in this publication. More information about the topic will be given below;
- `persistenceArea` indicates the area(s) in which the message should be *persistently disseminated*. This can be any definition of area (e.g., circle with centre-radius, list of street segments, etc). Notice that this process can be automated. We can automatically identify the areas where the information is relevant based on historical data and the road topology. As we will examine later, the information will spread even further from this area, based on the message content and the current mobility patterns. Furthermore, popular publications (based on how many vehicles are interested in the content of the disseminated information) will spread further.
- T_{start} is the time at which the publication begins to be valid; it can be the current time or any time in the future;
- T_{end} is the time at which it ceases to be valid;

For example: the primitive which should be invoked by the publishing application is:

```
publish(Type = Warning:Road works| PointOfInterest = A51:Junction 12, 2km around
                                Junction 11, 4pm, 6pm)
```

In this example, drivers that intend to use *Junction 12* of motorway A51 will be informed that there are road works, when driving close to *Junction 11* between 4pm and 6pm.

The message contains the type of the publication (Warning) and the sub-type (Road works). It also contains the *Point of Interest (POI)* of the publication (e.g., the location of the road works, A51:Junction 12). This content is used to determine whether a vehicle that receives the publication is affected or not (i.e., if the vehicle will drive through Junction B).

We would like the drivers to be able to register interests about certain events concerning their current location, future route, final destination, etc. These interests are called *subscriptions* and the vehicles interested in an event are called *subscribers*. Subscriptions can be invoked automatically from the application (by the navigation system). We envisage that a driver only needs to define some general policies and their navigation system can generate all the appropriate subscriptions automatically and be able to match them.

For instance, the navigation system of a vehicle could subscribe to receive warnings about road segments in its calculated route:

```
subscribe(Type=Warning|PointOfInterest=MyRoute)
```

As before, these keywords contain the type of the publication (any warning) and the area of interest (vehicle's route). The vehicle will later use its navigation information (if available) to determine if it is interested or not.

Finally, the application can unsubscribe its interest as follows:

```
unsubscribe(subscription ID)
```

3.2.2 Content Matching

The message content is crucial as it will be used to identify which of the vehicles are actually interested in and, thus, influence the content-based dissemination. Similarly, subscriptions provide guidelines (keywords) about what the application is interested in receiving (e.g., *receive warnings about accidents on the vehicle's route*). However, our system should be able to evaluate whether a publication matches a subscribed interest by combining subscriptions with navigation information. More specifically, the vehicle has to respond as *1) Interested* or *2) Not Interested*.

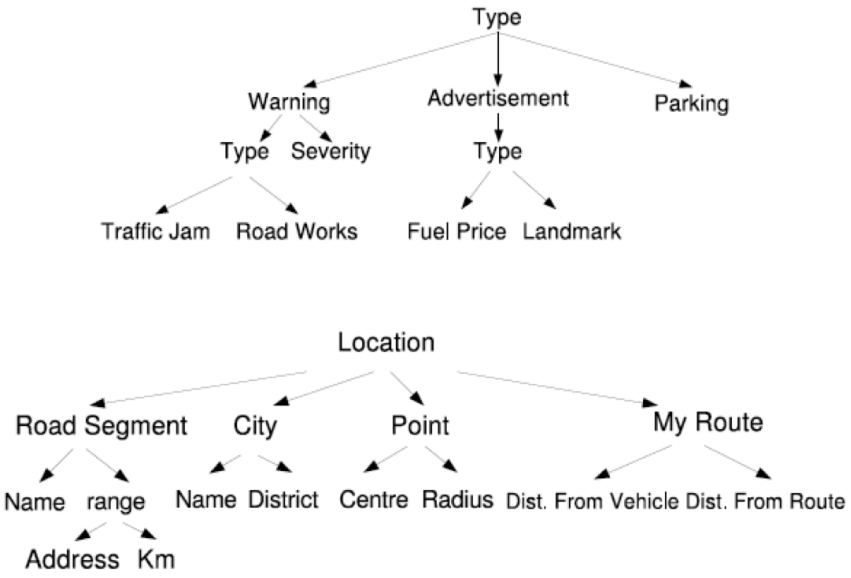


Figure 3.3: Matching and organising attribute/value pairs

We can match Publish/Subscribe attributes (publication's content) using existing matching protocols [Meier 02, Fiege 03]. In addition to these existing methods, in our approach information provided by the navigation system may be used so as to match content that concerns locations:

- *The map database* of existing navigation systems contains various types of information that can be exploited by the applications to create and match topics. For example, street/city/area names can be mapped to locations (e.g., a subscription about ‘‘`warnings about London`’’ can be matched with a publication about a ‘‘`warning about UK`’’).

- *The GPS location of the vehicle* can be used to match local information (e.g., a subscription about ‘‘`fuel stations that are within 1km from the vehicle's current location`’’ can be matched with a publication about an ‘‘`open fuel station on 54 Oxford street`’’).

- *The suggested route* of the vehicle can be used to evaluate information relevance. For example, a subscription about ‘‘`traffic updates on my route`’’ can be matched with a publication about ‘‘`traffic jam on Oxford street`’’. We believe that this is an important primitive as most of the vehicular applications will mainly consider publications about their route.

- *The vehicle's final destination* may also be treated as content. For example, a subscription about ‘‘`any information about my destination`’’ can be matched with a publication about ‘‘`sales on 54 Oxford street`’’.

Therefore, in our model, subscriptions are regarded as a dynamic set of suggestions that, when combined with the navigation information that is available in modern vehicles, can help us to determine whether an application (or the driver) is interested in receiving a message. More specifically, we consider two cases.

Firstly, using traditional content-based subscription matching, as used in SIENA [Carzaniga 01] where selection predicates are employed to identify which messages are relevant. In our implementation each message content is formed as a attribute/value pair and the content-matching engine applies the selection predicate to identify whether the attributes match.

Secondly, to identify the geographic relevance, each geographic content is evaluated in two sub-steps: i) we use the map database to map keywords to possible geographic locations. This function is already available in modern navigation systems and ii) we then examine if these locations overlap (using a K-D tree on a 2D space).

These two mechanisms ensure that the information will only spread in areas where there are vehicles interested in the published content.

Figure 3.3 shows an example of how we organise and characterise the message attributes in our implementation. The content is organised in main attributes that can contain one or more sub-types. When all values match the vehicle is considered as a subscriber.

3.2.3 Publication Semantics

In this section we present an overview of the publication mechanisms that we are going to use in order to publish messages to subscribers in certain geographic areas. In the next two sections, we will examine the routing and dissemination mechanisms in detail.

Our dissemination mechanism can be described in four simple steps (Figure 3.4):

1. **Notification Generation:** The notification is first generated by the publisher

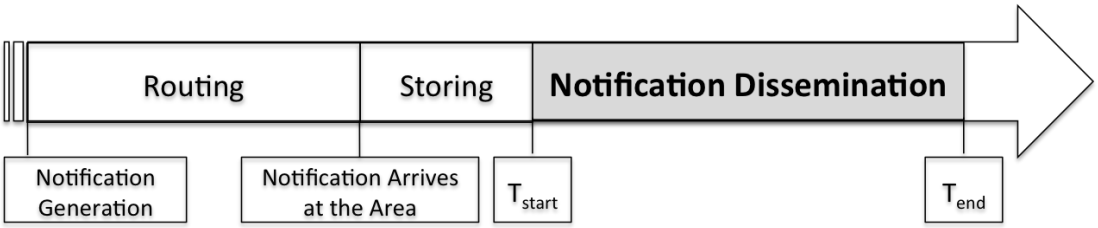


Figure 3.4: Timeline of events during publication of a notification.

by invoking the `publish()` primitive. The publisher defines the POI and all the required information that will help the vehicles to identify if they are interested in this notification or not. Furthermore, it indicates the geographic locations where the information should be persistently disseminated. As we mentioned previously, to guarantee persistence we need to identify some key locations where a copy of the notification will be stored (homeZones). Our framework can automatically cover the dissemination area with replicas and it will calculate the geographic locations of all the homeZones.

2. **Publication Routing:** The message replicas are then routed towards their homeZones using both the infrastructure and the vehicle-to-vehicle communication (as described in Chapter 2).

3. **Publication Storing:** Once the message has been routed to the area, it is stored there (in an infostation, if available, or in a vehicle inside the persistence area) until the start of the publication time. More details about this are following in Sections 3.3.1 and 3.3.3.

4. **Publication Dissemination:** Due to mobility, new subscribers in the area need to be informed. During the publication time, any subscriber that is in the communication range of a replica should be notified. Note that subscribers are not aware that they have entered an active dissemination area: when they enter, they will start receiving notifications from nearby subscribers (or infostations) that already have a copy of the message. The dissemination protocol is described in the following section (Section 3.3).

We will now give details about our push-based dissemination protocol.

3.3 Protocol Description

As we explained before, given the heterogeneity of the scenarios we target, a widespread presence of infostations cannot be guaranteed at any time and any place. Therefore, our protocol seamlessly exploits both infostations (where available) and vehicle-to-vehicle communication to *deliver* and *store* messages in the intended locations.

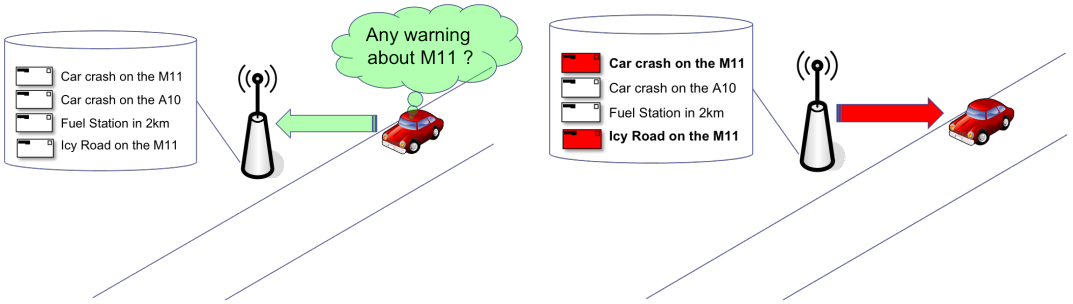
Hereafter, for sake of clarity, we illustrate our approach in separate steps by first describing the basic version of the protocol, assuming pervasive infrastructure availability, and then we show how we can relax this assumption, incorporating opportunistic and ad-hoc communication to i) extend the dissemination range beyond the persistence area and ii) maintain the replicas near their homeZones even in areas without infostations. Nevertheless, these three variants are not independent but co-exist in our protocol to provide a single solution addressing content-based dissemination in heterogeneous environments.

3.3.1 Infrastructure-based Persistence

As detailed in Section 3.1, the aim of our protocol is to ensure that all drivers are promptly informed about events relevant to their route (e.g., traffic jams affecting them or gas prices). To this end, information about these events should be stored at specific locations (*homeZones*) such that all approaching vehicles can be notified (an example is given in Figure 3.2).

The identification of the exact position and the number of these homeZones can be done automatically or in an application specific way and will vary according to the type of information and the road topology. In our implementation we use a simple map coverage algorithm to make sure that there is at least one replica in every path leading to the POI. Alternatively, a highway agency can strategically define the areas where the information should be persistent: for instance, in case of the traffic jam on a highway in Figure 3.1(a), the homeZones sit on the main junctions to access the highway, so that vehicles can avoid entering the highway and choose alternative paths.

For *each homeZone a replica of the original message is created, routed and stored* at the corresponding geographic location. Under the assumption of widespread infrastructure, messages can be sent to the nearby infostations to be disseminated. A simple



(a) First phase: The vehicle advertises any possible interests (subscriptions, navigation information etc)

(b) Second phase: The infostation packs all the publications that match and forwards them to the vehicle.

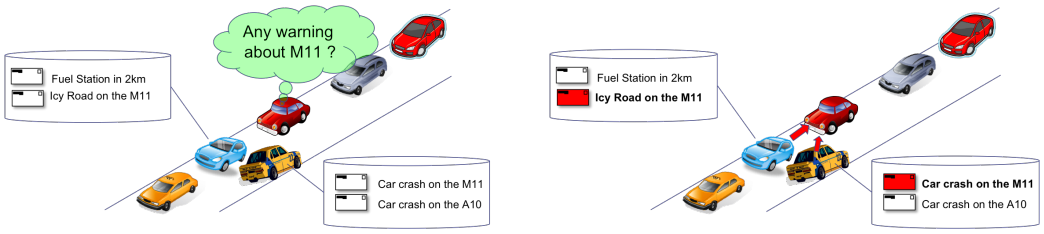
Figure 3.5: Our Two-Phase communication scheme.

yet inefficient solution would be to have each infostation periodically re-broadcast the message to all nearby vehicles. This, however, would incur a significant network overhead because i) messages are transmitted even if no subscribers are around and ii) subscribers are very likely to receive the same message multiple times by encountering several infostations along their paths.

To circumvent this issue and to remove unnecessary transmissions, we devised *a two-phase scheme* where each vehicle periodically advertises its planned route and its additional interests, if any. Through this information, the infostations can derive the actual subscriptions (both automatic and custom) and compare them against the stored messages. An example is given in Figure 3.5(a).

If a match occurs, the corresponding messages are then transmitted to the subscribers around the infostation (Figure 3.5(b)). To avoid duplicate receipts, the subscriber also piggybacks the IDs of the last λ messages received. In this way, before forwarding the message, the infostation could check whether that message has already been delivered. Only subscribers that did not receive the message previously can trigger a broadcast, which, however, can be heard by more than one subscriber in the area.

This two-way communication scheme ensures that although infostations may store a large number of publications, only the ones that are relevant to the drivers passing by them are the ones that will be broadcast and, thus, no overhead will be caused by stored publications that do not concern the vehicles around them.



(a) First phase: The vehicle advertises any possible interests (subscriptions, navigation information etc)

(b) Second phase: Any surrounding vehicle that has matching publications forwards them to the vehicle.

Figure 3.6: Our Two-Phase opportunistic communication scheme.

3.3.2 Opportunistic Dissemination

Even in the presence of infostations, opportunistic vehicle-to-vehicle communication can greatly enhance the performance of the above protocol by enabling the dissemination of messages in a broader area at a very small additional cost. To this end, we allow *any vehicle* to further spread a message, by retransmitting it when it meets a subscriber[1]. An example is given in Figure 3.6. This allows for opportunistic exploitation of vehicles which, in any case, have overheard the information (even if they are not subscribers) and can act as additional carriers for the information in the persistence area and beyond it.

Interestingly, *if no subscriber is encountered, no additional traffic is generated*, thus implementing an *interest-driven* routing scheme in which messages propagate only in areas populated by subscribers, possibly extending the persistence area defined by the application. Going back to the example of Figure 3.1(b), the dissemination will be automatically extended from the homeZones to the highlighted road segments as these are the locations where a large concentration of subscribers will be. There will be very few broadcasts inside the city, as very few vehicles are planning to drive near the location of the accident.

3.3.3 Ad-hoc Persistence

In some scenarios the existence of infostations is not sufficient to allow the dissemination to reach all interested vehicles. This could be for various reasons: i) the infrastructure

[1]Please note that vehicles can overhear a transmission even if they are not subscribers (i.e., when another vehicle in the neighbourhood triggered a broadcast).

has partially collapsed due to accidents or attacks ii) the infrastructure is not covering the whole dissemination area as this may be vast iii) the information has a very fine granularity with respect to the infostation coverage (e.g., parking spots positions, traffic information).

With respect to the approach presented in Section 3.3.1, if infostations are not widely available, we then need to find ways i) to *route* messages from the publisher to the persistence area and to ii) make sure that we can keep notifying drivers that pass by the indicated homeZones by *keeping a replica* of the publication near each one.

While the just described opportunistic dissemination helps to partially solve these issues in a hybrid scenario, it does not guarantee that the dissemination will not fade away. This can happen for a combination of two reasons i) if the density of the vehicles is low (e.g., during the night) and ii) the information is not popular (only subscribers will trigger further dissemination). Clearly, in such cases, the number of vehicles that hold a copy of the message will be very limited and the dissemination will eventually stop.

We need to replace the service that the fixed infostation would have provided. Therefore, a more proactive mechanism is needed to enforce persistence in a semi or totally decentralised scenario, relying on *scalable* and *inexpensive* ad-hoc communication among vehicles. In our approach, we chose to replace the missing infostations with *special copies of the publication* that we call message *replicas*, which should be constantly maintained on (or around) the homeZones whenever this is possible. Any carrier (vehicle) that holds a replica will be considered as a mobile infostation.

Ideally a carrier would be a vehicle that is always near the replica's homeZone. This is unrealistic as, in general, vehicles continuously move from one location to another and, hence, new carriers must be selected. In particular, not only the current location of a node is important but also its future one as carriers moving *towards* a homeZone are much better than those departing *from* it.

And since in vehicular networks information about future movements can be derived from the information provided by the navigation system, we exploit this system to find neighbours that are likely to keep the replica near the homeZone. Note that information about the future route is broadcast, even in the infrastructure-based version of our

Figure 3.7: Routing a message replica to its homeZone

protocol, to derive automatic subscription. Therefore, no additional traffic is required.

Therefore, we *constantly route a replica* to the homeZone using the same principles as GeOpps (Chapter 2). We use the same utility function to evaluate whether a carrier v is likely to deliver and keep a replica m around its homeZone.

More formally, we have:

$$U_v(m) = \alpha T_{NP} + \widehat{T}_{homeZone} \tag{3.1}$$

being T_{NP} the time needed to reach the nearest point, given the current route of the vehicle and $\widehat{T}_{homeZone}$ the estimated time to go from NP to the homeZone. The difference to GeOpps is that *this process does not stop when the replica arrives at the homeZone*, as we need to keep the replica there for a time period. Instead, we just continue to route the message to this location until the replica expires (we keep looking for vehicles that can carry the replica back to the homeZone). Notice that in cases where the density of the vehicles is very low (i.e., no vehicles are available near the homeZone), this process does not guarantee that the replica will be always maintained near the homeZone. However, our utility function ensures that the replica will be "attracted" to the homeZone by hopping to vehicles travelling towards it.

For instance, in Figure 3.7, a message is published at the infostation I and needs to be routed at the homeZone. Both vehicle V_A and vehicle V_B are potentially eligible

as carriers but the latter is preferable because its path will get closer to the final destination, i.e., the message's homeZone. Nevertheless, while approaching NP_B the node encounters another vehicle, V_C, whose route happens to cross the homeZone and hence it takes over the message. As soon as V_c passes the homeZone, it needs to find another carrier going in the opposite direction back to the homeZone.

Thus far, we focused our attention only on carriers but the goal of the protocol is to deliver the message to subscribers. To this end, when a node receives the planned route from its neighbours, beside checking whether there is any potentially better carrier, it also verifies whether its messages are of interest to any of its neighbours (e.g., Figure 3.6), adopting the same approach as in the infrastructure-based version (e.g., replica routing and information dissemination occur at the same time). Therefore, we distinguish between two types of messages: i)replicas and ii)simple copies of the publication. The former are the messages that need to be located as close as possible to the corresponding homeZone and can never be deleted (at least until they expire). Simple copies of the publication, instead, are those messages that have been delivered to subscribers or overheard.

To sum up, in our dissemination paradigm we aim that some replicas of the publication are maintained in key geographical locations, called homeZones (e.g., Figure 3.1). These replicas can be either stored inside infostations (if available) or inside normal vehicles chosen due to their mobility patterns in order to maximise the probability of keeping the replica near the homeZone. Subscribers are notified using our two-phase communication scheme and they can further spread the dissemination if they happen to meet more subscribers on their way (e.g., Figure 3.1(b)).

3.3.4 Protocol Details and Pseudocode

The pseudocode for the protocol is presented in Figure 3.9 along with the necessary definitions listed in Figure 3.8.

Periodically each node broadcasts a $CONTROL$ message containing the description of its planned route to its 1-hop neighbours along with the list of its custom subscriptions as indicated in Figure 3.9 (*Subscription Dissemination).* Also the identifiers of the last λ messages received are piggybacked in this message. The information contained in the message is key for the neighbours to determine the message forwarding

decision.

Indeed, when a $CONTROL$ message c is received (see second block in Figure 3.9), the buffer is re-evaluated to assess whether the originator of c may be interested in some messages either as a new carrier or a subscriber. First, the expiration time of each buffered message is evaluated (lines 2-3). If the message is still valid and the node is a carrier for that message, i.e., the message is a replica, the route $\bar{R}$ contained in the control message is analysed and the utility U' of the neighbour for the given message is compared with the one of the current node (line 6-9). If it is lower, i.e., the neighbour will travel closer (or faster) to the destination than the current node, the message is transferred to the neighbour by means of a unicast transmission and then is removed from the buffer (lines 10-12). Otherwise, the content of the message is evaluated against the neighbour's subscriptions. If a match occurs and the message has not already been received by the node (i.e., $m.ID \notin I$) a copy of the message is sent (lines 14-17). Naturally, a number of optimisations are possible (e.g., packing multiple matching messages in a single transmission) but they are not discussed here to avoid complicating the pseudocode.

Similarly, when a $DATA$ message is received, if it is of interest to the application, i.e., it matches either an automatic or custom subscription, it is delivered to the application layer (lines 1-2 of **receive** $DATA$). Then, regardless of the outcome of this check, the message is inserted in the buffer. Indeed, if the node has been designated to be a carrier ($m.type$ = REPLICA), it stores the message until a better one is found. Otherwise, the message is stored too but it will be delivered only to the subscribers *opportunistically* encountered along the path.

Finally, *Message Publishing* consists simply of inserting the message into the local buffer. The message will then be taken care of and forwarded to the interested subscribers as well as "moved" to its homeZone, possibly through a better carrier, if and when encountered, according to the routing protocol we described thus far. Each replica is routed independently, i.e., whenever a better carrier is encountered only relevant replicas are removed from the local buffer and sent to the new carrier. Alternatively, we could just route one copy that splits it in r replicas when it arrives inside the homeZone. For our experiments, we chose the first method in order to improve robustness.

3.4 Related work

Related work can be found in the areas of Publish/Subscribe systems and Ad-Hoc dissemination systems.

Generally, these approaches employ a *message oriented middleware (MOM)* and a set of dispatchers to store the subscriptions and deliver the publication s to the subscribers (i.e., to the vehicles that are interested in receiving this message). There are many examples of successful notification middleware for fixed systems.

SIENA [Carzaniga 01] is one of the advanced distributed event notification systems. The service is implemented as a network of servers that provide access points to clients. Clients use the access points to advertise the information about events that they generate and to publish notifications containing that information. The service uses access points to notify clients by delivering any notifications of interest. In GRYPHON [Banavar 99], a hierarchical tree is constructed from publishers to subscribers and filter-based routing is used to forward the publications. Other examples are JEDI [Cugola 01], HERMES [Pietzuch 02].

However, these frameworks were mainly designed for fixed networks (e.g., the Internet). They often rely on communication primitives that were devised for traditional distributed systems and that tend to be fixed and location-unaware. Thus, these approaches cannot deal with the dynamics of ad-hoc networks.

Some attempts have been made to build publish-subscribe middleware for mobile, ad-hoc and hybrid systems.

The first category of approaches employ some algorithms to keep the underlying dispatcher *tree overlay updated*. The idea is to update the dispatchers in order to reconfigure the routing of notifications, in response to the network topology changes. Examples that use updated tree overlays are [Mottola 05, Cugola 02a, Cugola 02b, Huang 03, Sivaharan 05]. Furthermore, there are attempts to extend existing P/S systems using that model [Zeidler 03, Cugola 01, Fiege 03, Cilia 03, Carzaniga 01, Caporuscio 03].

A completely different approach is *broadcast-based* Publish/Subscribe. The motivation is that a tree topology over a MANET is hard to implement and keep updated. Thus, these approaches do not employ overlays, dispatchers or brokers to route the

publications; the subscriptions are only stored locally (as a filter between the network and the application). The publications are *broadcast* to the hosts using flooding-based or gossip-based techniques. Examples are [Huang 04, Costa 04, Costa 03]

Between these two approaches there is *semi-probabilistic routing* of the publications [Costa 05]. Routing relies on deterministic decisions driven by a limited view of the subscription information and, when this is not sufficient, resorts to probabilistic decisions performed by selecting links at random. More specifically, subscriptions are propagated only in the immediate vicinity of a subscriber, in contrast to most existing systems. Event routing leverages on this subscription information, whenever available, by deterministically routing an event along the link a matching subscription was received from. If no subscription information exists at a given dispatcher, events are forwarded along a randomly chosen subset of the available links. Although this scheme does not guarantee delivery to all the subscribers, it is able to tolerate frequent topological reconfigurations.

The tree reconfiguration methods are not suitable for frequent topology changes (too much overhead for updating the tree). Furthermore they are not suitable for delivering notifications when there are partitions or disconnections. Whereas the broadcast-based methods are more suitable for a highly dynamic environment but they produce a higher message overhead.

The use of geographical location in P/S systems was also researched. There are two main categories of location aware P/S: publish to subscribers in the *proximity* and publish to subscribers *in specific (remote) locations*. There is a lot of research on how to publish information to the subscribers that are in a given *range around the publisher* or on how to subscribe for notifications from publishers in the vicinity.

In Location-based Publish/Subscribe (LPS) [Eugster 05], the subscribers receive information for local events. The publication space moves around the publisher as the publisher moves. This work targets the use of existing infrastructure (e.g., GPRS, GSM, etc.).

In *Location-based Toronto Publish/Subscribe (L-ToPSS)* [Burcea 03] there is a study of location-based services. In this work, the location is defined as a *range from a subscriber*. The L-ToPSS architecture employs a *Location Matching Engine* that keeps

an updated position of the subscribers and their location matching filter. Furthermore, the idea of generating notifications when two subscribers or a subscriber and a publisher are *collocated* is studied by Z. Xu and H Jacobsen [Xu 05].

STEAM [Meier 02] exploits a proximity-based group communication service [Killijian 01] to deliver a message to nearby subscribers. AdTorrent [Shirshanka 05] broadcasts advertisements to local pedestrians/cars.

G. Cugola and J. Cote extended their existing work on distributed publish-subscribe middleware [Mottola 05] to support location awareness. In [Cugola 05], the authors introduce the `publishTo(message, location)` primitive in order to send a message to subscribers in a specific location. This approach uses a number of brokers to forward the message to subscribers. Thus, they need to update the overlay network when there is mobility.

X. Chen et al [Chen 03] proposed two policies to deliver notifications to subscribers in specific areas. In their first method, they used a server to monitor the location of the subscribers. When subscribers enter the areas defined by the publisher, the notification is routed to them. In the second method, the server sends the publication zone to the subscriber and when the latter enters into the zone, it contacts the server to receive the notification.

To the best of our knowledge, Abiding Geocast [Maihofer 05] is the most related work in terms of delivering a time-stable message in a geographical area. In order to disseminate the message in the area it employs periodic flooding or epidemic dissemination. With respect to this work, our protocol considers a content based approach where subscriptions, with the aid of the navigation system, are used in order to filter and route the message to the interested vehicles (i.e., our approach can be considered as content-based dissemination and not as Geocast protocol). In addition, through the use of ad-hoc persistence, we drastically reduce the overhead as opposed to epidemic approaches.

Works targeting multicast communication in vehicular networks recently appeared in the literature [Sormani 06, Korkmaz 04, Xu 04, Dornbush 07, Eichler 06]. They used different versions of scoped epidemic protocols to constrain the propagation of a message within the given area. Other work [Adler 06, Kosch 02, Caliskan 06], instead, define a

notion of relevance to enable the routing layer to self-identify the areas in which the messages should be delivered. In contrast to these approaches, our work offers a richer semantics in which publishers and subscribers are completely decoupled, as the former define the notification's content while the latter just express their interests so as to automatically spread the notification to areas where there are subscribers.

3.5 Conclusions

We have presented a protocol that allows pushing popular content in hybrid vehicular networks. We have identified the key characteristics of vehicular networks and the properties that such a dissemination system should have: messages should be disseminated according to their content and their geographic relevance, and the dissemination should be persistent for a period of time.

We have designed a framework that allows such a dissemination. In our framework, the publisher initiates a notification that includes, among others, a point of interest and a persistence area (e.g., a set of roads) in which the information needs to be disseminated. In our approach we use the concept of subscribers: vehicles interested in the information based on its content, their location, their future navigation route and their destination. Our mechanism exploits the navigation system information to match content.

Furthermore, in the absence of infrastructure, we can replace infrastructure with ad-hoc communication. We constantly route a number of master copies (called replicas) using our geographic routing protocol that we introduced in Chapter 2. Additionally, we also use opportunistic communication to further disseminate the information in areas where there are subscribers using our two-way communication scheme.

However, not all types of information concern a high number of vehicles. Using push-based dissemination, in areas with low infrastructure density, will just cause too much overhead due to the fact that we have to maintain the replicas in an ad-hoc way (constantly routing them towards their homeZones). In the next chapter we will see how vehicles can request custom information from nearby infostations by exploiting the mobility patterns given by their navigation system.

Variables

- self: node's own id.
- $(\bar{x}, \bar{y})$: neighbour's current position.
- $\mathcal{R}$: node's route, expressed as a ordered list $\langle (I_1, t_1), (I_2, t_2), \ldots, (I_r, t_r) \rangle$ of intersection points I_i and (expected) arrival time t_i.
- $\mathcal{M}$: the message buffer. It contains both replicas (master copies) and simple copies of the publication.

Messages

- CONTROL$< ID, R, x, y, S, I >$: the route advertisement of a node n. It contains:

  ID: n's ID.

  R: the expected route of n (taken by the navigation system).

  x, y: n's current position.

  S: n's subscription set

  I: set of the identifiers of the last λ messages received.

- DATA$< ID, Type, TTL, homeZone, POI, Content >$: this is a data packet stored in $\mathcal{M}$.

  ID: a unique ID for this message.

  $Type$: the type of the message.

  If $Type = Replica$ the message is routed to $homeZone$.

  If $Type = Publication$ message is just considered for notification (no routing).

  TTL: the time to live. When this expires the message is discarded.

  $homeZone$: the geographic destination of the message (constantly routed there).

  POI: the Point Of Interest.

  $Content$: the payload of the message (the data). Used for content-based matching.

Functions

- **send**$(msg) \rightarrow n$: send a unicast of message to neighbour n
- **broadcast** (msg): broadcast the message to all 1-hop neighbours
- **deliver** (msg): deliver the message to the application
- **matches**(m, S, R): returns TRUE if the message m matches a subscription $s \in S$ or is relevant for route R (i.e.,$m.POI \in R$)
- **expired**(msg): returns TRUE if the message expiration time has passed
- **computeUtility**(m, R): calculate the utility for the message replica m given the route R, using formula (3.1)
- **assignHomeZone**(m): returns the pair of coordinate (x, y) representing the assigned $homeZone$ for the replica r

Figure 3.8: Pseudo-code definitions for the push-based protocol.

Subscriptions Dissemination

NeighbourAdvertise()

create new message c: CONTROL$< self, R, x, y, S, I >$
broadcast (c)

Invoked on receipt of a CONTROL message from neighbour n.

receive CONTROL$< n, \bar{R}, \bar{x}, \bar{y}, \bar{S}, I >$

for all $m \in \mathcal{M}$ **do**
  if expired(m) **then**
    $\mathcal{M} \leftarrow \mathcal{M} \setminus \{m\}$
  else
    $sent \leftarrow$ FALSE
    if $m.type = replica$ **then**
      $U \leftarrow$ **computeUtility**$(m, \mathcal{R})$
      $U' \leftarrow$ **computeUtility**$(m, \bar{R})$
      if $U' < U$ **then**
        send(m)
        $sent \leftarrow$ TRUE
        $\mathcal{M} \leftarrow \mathcal{M} \setminus \{m\}$
    if $\neg sent$ **then**
      if matches$(m, \bar{S}, \bar{R}) \wedge m.ID \notin I$ **then**
        create a copy m' of message m
        $m'.type \leftarrow publication$
        broadcast$(m') \rightarrow n$

Invoked on receipt of a DATA message from neighbour n.

receive DATA$< ID, Type, TTL, homeZone, POI, Content >$

if matches$(m, \mathcal{S}, \mathcal{R}) \wedge m.ID \notin \mathcal{I}$ **then**
  deliver (m)
$\mathcal{M} \leftarrow \mathcal{M} \cup \{m\}$

Message Publishing.

publish$< m, expirationTime >$

create a set of replicas $\Gamma = \{m_1, m_2, \ldots, m_r\}$ of the published message DATA m
for all $m_i \in \Gamma$ **do**
  $m_i.type \leftarrow replica$
  $m_i.homeZone \leftarrow$ **assignHomeZone**(m_i)
  $\mathcal{M} \leftarrow \mathcal{M} \cup \{m_i\}$

Figure 3.9: Pseudo-code for the push-based protocol.

4

Pull-Based Information Dissemination

Push-based dissemination may not efficient when the information is not required by a large number of vehicles in an area. Therefore, we also need mechanisms to allow vehicles (or their navigation systems) to request and later receive custom information. For example, a Street View-like application would be useful on cars, as landmarks and intersection shapes could help one's sense of direction and provide a better visual aid to navigation systems rather than traditional 2D maps. Unfortunately, with the current state of technology, the usability of these visual tools is debatable. Indeed, the huge amount of data required (hundreds of gigabytes, if not terabytes) clearly excludes the possibility of pre-loading images on board.

Furthermore, users may require fresh and fine-grained information about their route or destination (e.g., recent images of intersections or of re-routing points to help them navigate through these landmarks).

We assume that there might be some scattered gateways (i.e., infostations or wifi

Figure 4.1: Our envisioned application.

hotspots) but these might not entirely cover the whole road network (e.g., as in Figure 4.2). Therefore, we would like to enable vehicles to request information from remote infostations using V2V communication to fill the gap. To implement such a pull-based framework we need to solve two challenges:

1. How to route the request from the vehicle to the nearest known infostation. This requires a geographic routing protocol such as GeOpps (Chapter 2).

2. How to route the reply back to the *moving vehicle*: This is a very important problem as it can take from milliseconds to minutes until the vehicle receives the reply.

Although the first step is fully answered in Chapter 2, the second step is very challenging and still unanswered.

4.1 Motivation

In order to explain the functionality of pull-based dissemination we start by illustrating a reference application that we will use throughout the chapter. Imagine users willing to travel through a route they are not familiar with. The navigation system already provides valuable help by indicating what path to follow, where to turn next and how long to stay on a specific road. However, without auxiliary information such as imaging, the task of finding the correct road to turn into or finding the desired point of interest

(e.g., a shop or a restaurant) may remain challenging. This is even more true if there are deviations due to temporary roadworks or accidents. Recent images of important locations and the subsequent turns would further facilitate driving.

We envisage a system where data can explicitly be requested by drivers or implicitly by the in-car system. An example of the interface that we have in mind is illustrated in Figure 4.1 where the navigation system requested a fresh picture of a diversion, possibly uploaded by another vehicle in the area. Similarly, vehicles can subscribe to receive information (e.g, subscribe to receive any warnings about their route), and later, when a matching occurs, local infostations can forward the matching notifications.

We assume the existence of a WiFi infrastructure to which vehicles can sometimes connect and which is connected to a central server: however, as we also explained in the previous chapters, it is quite realistic to assume that no complete coverage of this infrastructure exists.

We further assume that vehicles (or, better, their navigation systems) know the location of the infostations and, at any point in time, they know which one is the closest. When in need, a vehicle may request information for a specific location or point on the route (possibly just tapping on a specific point on the screen displaying the location of interest on the map). The request is then routed opportunistically through other vehicles to the closest infostation. The reply is then routed back to the requesting vehicle: the routing considers the fact that this will have progressed on its path. The details of the routing protocols are distilled in Section 4.3.

Information (e.g. Images) about the relevant points can already be available on the Internet, or can be uploaded dynamically on the server through an opportunistic mechanism akin to the one just described. When the data requested are not available on the server, they are sought by sending a query to the location of interest: the query is routed to the location in the same way as a reply is (i.e., by reaching first the closest infostation and then opportunistically): when a vehicle in the relevant location is reached, the in-car system collects the information and sends it back to the server in the same way that an information query is sent.

4.2 Pull Architecture

In this section we examine how we can design a framework that allows drivers to pull information from a nearby infostation/location. Although these primitives are much simpler than the ones of the push-based approach, the underlying semantics and protocols are quite complex.

4.2.1 Primitives

There is only one basic primitive that this method exposes to the application:

```
request(request Query, gatewayLocation)
```

This primitive creates a query message Q to be forwarded to the specified gateway location (e.g., the location of the nearest known infostation). Our framework will automatically include any information needed in order to route the query and the reply back to the vehicle.

Furthermore, the query Q can be a set of subscriptions (e.g., subscribe to receive warnings about the vehicle's route) that can later trigger a reply.

4.2.2 Semantics

In this section we present an overview of the pull mechanisms that are employed when the `request` primitive is used. For sake of clarity we now illustrate how the approach works in a simple scenario.

Lets assume that:

1. A vehicle v wants to receive images of a junction some miles ahead on its path. These can, for instance, be requested automatically by the navigation system periodically or prompted by a driver when needed.

2. The query Q is injected in the system which isolates the closest infostation and issues a request message which will be routed to its geographic location. An example is provided in Figure 4.2 where infostation C is selected as the closest to the current position of the vehicle. The request message also *contains V's sug-*

gested route to destination. This information can be extracted from the navigation system.

3. The request message opportunistically (either by hopping from vehicle to vehicle like in Chapter 2, or by being carried by the same car) reaches the closest infostation and then the backbone.

4. When the reply R, containing the data requested, is ready, our system selects an appropriate infostation to inject the answer into the vehicular network. This infostation will be the one which is estimated to be the closest and beyond the requesting vehicle V considering its route. Note that the infostation may not be on the requesting vehicle's path as we will use vehicle-to-vehicle communication to fill this gap. For example, in Figure 4.2 infostation B will be selected as it is close to V's path and ahead of its route.

5. The reply R is then opportunistically routed (V2V) from the infostation to a point *on vehicle V's route.* Our aim is to deliver R on the path ahead of the vehicle and keep it there in order to maximise the chances of delivery. Furthermore, the reply travels towards the vehicle (always on the path) so as to minimise delay.

   In Figure 4.3 we show an example of how the reply is routed back to the vehicle from an infostation which is not on the vehicle's path. Initially (Figure 4.3(a)) the information is routed from the selected infostation B to intercept the route of the vehicle at point NP.

   When the packet arrives at NP it then starts to move towards the vehicle but always on the route. This is shown in Figure 4.3(b) where the packet is routed from NP backwards along the route until it meets the vehicle at location L_3.

6. Eventually, the vehicle is reached and the data delivered.

Our framework exploits the vehicle's navigation system i) to route information/query into specific geographic locations by identifying the best carriers (i.e., those nodes which have more chances of reaching the intended destination), ii) to route a reply towards a predicted vehicle's position using the vehicular network iii) to keep it on the requesting vehicle's path while moving it towards the vehicle.

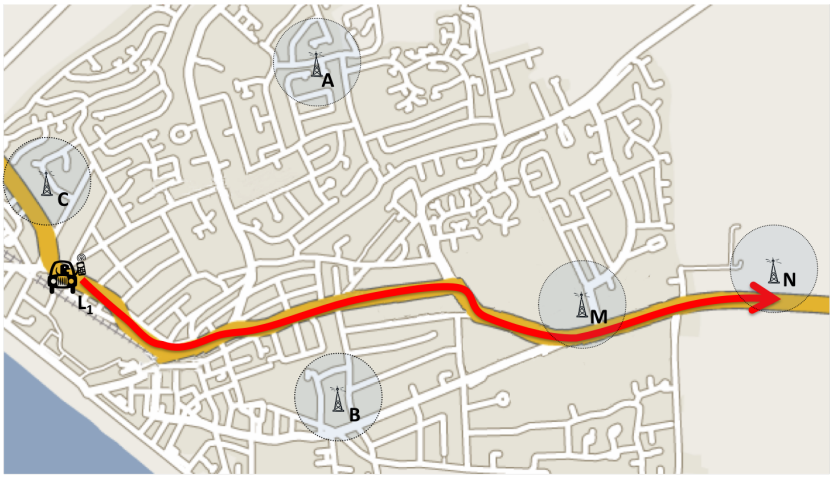


Figure 4.2: Infostations are represented by antennas while the shadowed areas represents their transmissions range. L_1 is the position of the vehicle when the query is generated.

4.3 Protocol Description

We now describe the protocol's steps in detail.

4.3.1 STEP 1: Query Routing

When a query is generated, the node selects the closest infostation indicated on its map (e.g., infostation C in Figure 4.2) and creates a new message. If the infostation is in range, the message is immediately transmitted to the infostation. Otherwise, the message is kept on the vehicle until a neighbouring vehicle that is travelling closer to the target infostation is found (as we saw in Chapter 2). Once the request is routed to an infostation, it is then collected in a central location on the backbone, where it is processed. The same protocol is also used to collect content (e.g., images) generated at a vehicle, back to the server through the closest infostation.

4.3.2 STEP 2: Reply Infostation Selection

The reply is prepared by the server and contains the relevant data (e.g., images) which has been previously collected, or which was requested on the fly if not already available. Once the reply is ready it will need to be routed to an infostation on the road network so that the interested vehicle can be reached. As we described before, the information about the route and position of the vehicle when the request was sent, which is pig-gybacked on the request packet, and the information about predicted travelling times

on roads, are used to determine the best infostation to receive the reply data. The selected infostation is the one closest to the vehicle's predicted position and is chosen to be *beyond* the vehicle's predicted position.

More precisely, to evaluate whether an infostation can potentially deliver the message to the vehicle:

1. We calculate the Nearest Point (NP) on the route of the vehicle v to the infostation (i.e., the closest point between the route of the vehicle and the infostation). An example is given in Figure 4.3(a).

2. Calculate an estimation of the time t_I needed to route the message from infostation I to NP using the GeOpps METD utility (Section 2.2.2). In Figure 4.3(a), this is the time required for the message to be routed from B to NP.

3. Calculate the time $t_v(I)$ when the vehicle will arrive at NP based on the information from its navigation system (i.e., the time required for the vehicle to drive from L_2 to NP in Figure 4.3(a)).

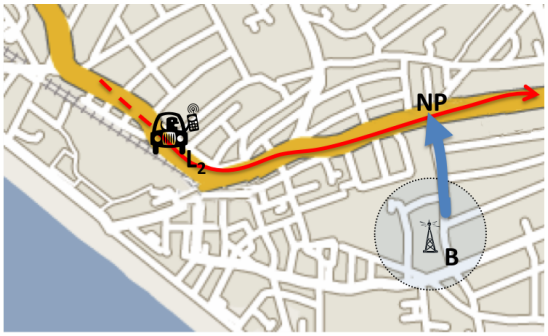
4. An infostation I is considered if the time taken by the message to reach the vehicle's route (t_I) is shorter than the time the vehicle would take to reach this point ($t_v(I)$). If multiple infostations fulfil these constraints, the one with the lowest $t_v(I)$ is chosen in order to minimise the time for the vehicle to reach NP (i.e., the infostation that can deliver the message earlier to the vehicle):

$$I = \underset{I}{\operatorname{argmin}}\{t_v(I) | \forall I : t_I < t_v(I)\} \tag{4.1}$$

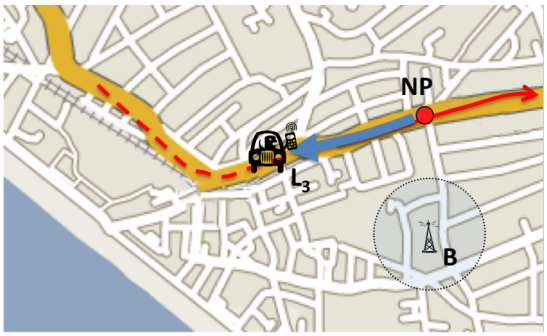
Once the infostation is chosen, the reply message is routed there through the backbone network. An example is provided in Figure 4.3: infostation B is the closest infostation to the vehicle route among those depicted in Figure 4.2 and, hence, the reply originates there.

4.3.3 STEP 3: Reply Opportunistic Routing

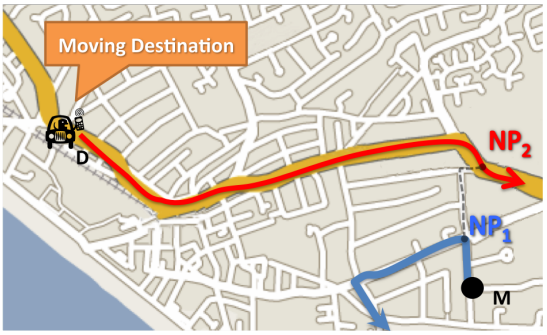
Once the reply message is on the appropriate infostation, it will need to be routed to the requesting vehicle. We consider two sub-steps: i) the message has to be routed along the destination's path (Figure 4.3(a)) and then ii) it has to be kept on it and



(a) Intercepting route.



(b) Keeping the message on route and moving backwards.



(c) Neighbour selection when message is outside destination's path.

(d) Neighbour selection when message is on destination's path.

Figure 4.3: Routing the reply. L_2 and L_3 indicate the vehicle's progressive positions. NP is the nearest point between the vehicle's route and the infostation.

preferably moved backwards in order to maximise the probability to meet the vehicle (Figure 4.3(b)).

The first part is performed differently to the forwarding of the request packet: instead of routing a message to a specific point[1], it is routed towards the whole destination's *path*.

This time, as you can see in Figure 4.3(c), there can be up to *two nearest points*: NP_1 on the neighbour's path and NP_2 on destination's path [2]. These two points are selected to minimise the driving time between *the two paths* (dotted line in Figure 4.3(c)). Then our framework estimates the driving time t_1 required from the current position to NP_1 and, afterwards, the time t_2 required from NP_1 to NP_2. If the message is estimated to be at NP_2 earlier than the vehicle-destination then it is considered for

[1]GeOpps is only used to deliver a packet to a specific geographic location.
[2]These two points can overlap when the two routes cross each other.

forwarding.

The second sub-step starts once the reply has reached the path, it is presumably beyond the vehicle's position (e.g., Figure 4.3(b)). Our target at this stage is primarily to *keep the message on the path* to maximise the delivery ratio. Nevertheless, to minimise delay, our framework hops the reply backwards on the requesting vehicle's route until the vehicle is reached. When a vehicle carrying a reply meets a neighbour it evaluates whether or not to forward a message based on how much the path of the neighbour overlaps with the destination's:

1. It checks if the neighbour is on the requesting vehicle's path.
2. It finds the last waypoint P before it will deviate from the destination's path. This is the earliest point on $D's$ route until which this carrier would hold the message without drifting away from the destination's path (for example, in Figure 4.3(d) the green vehicle (marked as N_3) will stay on D's path until location N_3). In case of one-way roads P is usually the current location of the vehicles.
3. It estimates the time t_{vn} it will take for the destination D to reach P (e.g., driving time between D and N_3 in the previous example).
4. It calculates its own last point Q before deviating and the time t_v it will take for the requesting vehicle to reach Q;
5. If t_{vn} is lower than t_v, then the neighbour is considered as better carrier and the message is handed over. Intuitively, this means that the neighbour will carry the message closer to D before deviating from D's route. In the one-way road case described before, neighbours driving behind the current carrier will be preferred because their current location is considered as the deviation point P.

An example is sketched in Figure 4.3(d). Once message M has reached the vehicle's route (black dot in the picture), three different neighbours, respectively N_1, N_2, N_3, are available. Among these, N_3 is selected because, as illustrated in Figure 4.3(d), it is going in the right direction (i.e., towards the vehicle) and its expected route has a higher overlap with the vehicle's one. Along the path, other neighbours may appear and the message will be transferred to one of these if it is going closer to the destination until the message is finally delivered. Sometimes it may occur that a vehicle holding

the message is deviating from the destination's navigation path and there is no suitable candidate. In this case, our protocol re-initiates the procedure to bring the message back to the route as described above in the case of the infostation.

You can find a more detailed and technical description of the algorithm in Figures 4.4 and 4.5.

4.3.4 Enhancements

The described protocol is based on the idea that the estimation of the travelling time of vehicles is quite precise. Here we propose a mechanism to mitigate the existence of errors in these estimations. As we have detailed above, when the vehicle sent the query, it included information about its future route. Our framework uses this estimation to predict the location of the vehicle and, eventually, route the reply back to it. However, it might happen that the requesting vehicle is moving faster (or slower) than expected: the reply will reach the NP and move backwards, but the vehicle will have passed NP before the message reached it (or will still be very behind). The reply will continue going backwards until it passes the point where, according to time estimates, it was supposed to meet the requesting vehicle. To solve this, we introduce the following fail-safe mechanisms. When the vehicle reaches the point where it detects that the requesting vehicle should have already been met, it splits into two copies. Copy A will continue moving backwards (for a little longer) just in case the estimates were wrong and the vehicle was indeed still late. Copy B will start moving forward (i.e., opposite direction) trying to catch up with the vehicle that is now estimated to be ahead[3]. However, since the reply is expected to chase a vehicle going towards the same direction, a more rapid (greedy) forwarding scheme is selected: the reply is just forwarded to any neighbour that is on the path of the vehicle as long as it is moving "forwards". Again the route of the requesting vehicle is used to make sure that the neighbour is making progress towards the right route. This "greedy" forwarding results in a higher hop count (since the reply is just forwarded every time a vehicle is a little closer) but increased speed. A TTL on each message is also added to halt the propagation if the target vehicle is not met within a given time.

It may happen that a reply gets lost due to transmission errors or an unexpected

[3]This mechanism ensures that at most there will be only two copies of the same reply in the system.

Variables

- self: node's own id.
- $(\bar{x}, \bar{y})$: node's current position.
- $\mathcal{R}$: node's route, expressed as a ordered list $\langle (I_1, t_1), (I_2, t_2), \ldots, (I_r, t_r) \rangle$ of intersection points I_i and (expected) arrival time t_i.
- $\mathcal{IS}$: the set of all infostations installed. Each infostation $i \in \mathcal{IS}$ contains the id ($i.ID$) and the geographic location $i.(x, y)$.
- $\mathcal{M}$: the message buffer.
- τ: the time threshold.
- α: the weight putting more emphasis on distance.

Messages

- CONTROL$< ID, R, x, y >$: the route advertisement of a node n. It contains:

  ID: n's ID.

  R: the expected route of n (taken by the navigation system).

  x, y: n's current position.

- DATA$< ID, Type, TTL, R_v, D_{id}, Content >$: this is a data packet stored in $\mathcal{M}$. When a better carrier is found it is forwarded n. It contains:

  ID: a unique ID for this message.

  $Type$: the type of the message. In this section $Type$ = RouteReply(route on path).

  TTL: the time to live. When this expires the message is discarded.

  R_v: the destination's route.

  D_{id}: the destination's ID.

  $Content$: the payload of the message (the data).

Functions

- **send**$(m) \rightarrow n$: send a unicast message m to neighbour n
- **broadcast** (msg): broadcast message m to all 1-hop neighbours
- **deliver** (msg): deliver message m to the application
- **computeNP**(R, x, y): find the point (NP) on route R which is closest to the location (x, y)
- **computeNP_R**(R_1, R_2): find the point (NP) on route R_1 which is closest to route R_2
- **computeIP_R**(R_1, R_2): find the last point on route R_1 before it deviates from R_2
- **computeETR**(x_a, y_a, x_b, y_b): compute the estimated time requested (ETR) to opportunistically move a message from A(x_a, y_a) to $B(x_b, y_b)$
- **computeETR_R**(R, x, y): compute the estimated time requested (ETR) for a vehicle on route R to reach (x, y)

Figure 4.4: Pseudo-code definitions of the pull-based algorithm.

change of route by the vehicle. In this case, the time-out on the vehicle's navigation system would expire and a new query would be triggered.

Part A. Infostation Selection (vehicle)

REPLY$< R_v >$

1: $\hat{\imath} \leftarrow \perp$
2: $\hat{t} \leftarrow \infty$
3: **for all** $i \in \mathcal{IS}$ **do**
4: $(x_n, y_n) \leftarrow$ **computeNP**$(R, i.x, i.y)$
5: $t \leftarrow$ **computeETR**$(i.x, i.y, x_n, y_n)$
6: $t_v \leftarrow$ **computeETR_R**(R_v, x_n, y_n)
7: **if** $t_v - t > \tau$ **then**
8: **if** $t_v < \hat{t}$ **then**
9: $\hat{\imath} \leftarrow i$
10: $\hat{t} \leftarrow t_v$
11: **return** $\hat{\imath}$

Part B. Send packet on vehicle's route

receive CONTROL$(< n, \bar{R}, \bar{x}, \bar{y} >$:

1: **for all** $m \in \mathcal{M}$ **do**
2: **if** $(\bar{x}, \bar{y}) \notin m.R_v$ **then**
3: $\bar{NP}_1 \leftarrow$ **computeNP_R**$(\bar{R}, m.R_v)$
4: $\bar{NP}_2 \leftarrow$ **computeNP_R**$(m.R_v, \bar{R})$
5: $\bar{U} \leftarrow \alpha$**computeETR_R**$(\bar{R}, \bar{NP}_1) +$ **computeETR**$(\bar{NP}_1, \bar{NP}_2)$
6: $NP_1 \leftarrow$ **computeNP_R**$(R, m.R_v)$
7: $NP_2 \leftarrow$ **computeNP_R**$(m.R_v, R)$
8: $U \leftarrow \alpha$**computeETR_R**$(R, NP_1) +$ **computeETR**(NP_1, NP_2)
9: **if** $\bar{U} < U$ **then**
10: **send**$(m) \rightarrow n$
11: **else**
12: **if** $(\bar{x}, \bar{y}) \in m.R_v$ **then**
13: $\bar{NP} \leftarrow$ **computeIP_R**$(m.R_v, \bar{R})$
14: $\bar{t} \leftarrow$ **computeETR_R**$(m.R_v, \bar{NP})$
15: $NP \leftarrow$ **computeIP_R**$(m.R_v, R)$
16: $t \leftarrow$ **computeETR_R**$(m.R_v, NP)$
17: **if** $\bar{t} < t_v$ **then**
18: **send**$(m) \rightarrow n$

Figure 4.5: Pseudo-code of the pull-based algorithm.

4.4 Related Work

Previous work on routing protocols for vehicular networks (e.g., [Lebrun 05, Fubler 03]) mostly addresses the issue of delivering the message from a mobile vehicle to a fixed destination (i.e., a sink), while our system instead allows also for the information to be routed back to a moving vehicle, once this has requested the information.

Other solutions [Ko 02, Maihöfer 04] perform a restricted flooding, called GeoCasting, of the reply around the last location of the vehicle hoping that one of the copies will actually reach the mobile destination. This, however, would significantly increase the overhead, thus making this approach infeasible for densely populated urban areas.

A large body of work recently appeared in the literature addressing this kind of content dissemination in vehicular networks [Sormani 06, Korkmaz 04, Xu 04, Dornbush 07, Eichler 06]. They use different versions of scoped epidemic protocols trying to eventually deliver the message to the destination. Finally, inspired by the work on geographic routing in Mobile Ad Hoc Networks [Mauve 01, Maihöfer 04], position-based routing protocols targeting vehicular networks also appeared in the literature, e.g., [Lebrun 05, Zhao 04]. Unfortunately, none of these approaches fully satisfy our requirements as all of them assume that the geographic position of the final destination, whether an area or a specific host, is known a priori. Vice versa, in our scenario, replies must be forwarded to a mobile vehicle and, hence, its position is continuously changing. Also, while most approaches offer a push-based interaction, we opted for a pull-driven paradigm which seems to suit better the applications we target.

In the close research area of Delay Tolerant Networks [Jain 04], several routing protocols have been devised to deliver messages in a store-and-forward fashion based on opportunistic contacts [Shah 03, Zhao 04]. These protocols exploit different mechanisms to route a message to the destination such as statistics of previous encounters, or precise mobility schedules to find the best routes. However, none of them has been specifically designed for vehicular networks and it is not clear how they can cope with the peculiar mobility patterns of these networks.

Finally, some recent papers (e.g., Drive Through Internet [Ott 05, Bychkovsky 06] and Cabernet [Eriksson 08]) focus on techniques to improve the 1-hop communication between vehicles and infostations, by operating several optimisations at both the link

layer and transport layer. While the goal of such work differs from ours, these techniques are complimentary to our approach and integrating them in our framework is part of our future research agenda.

4.5 Conclusions

In this chapter we introduced the architecture and the protocols required to perform pull-based content dissemination. Our framework allows vehicles to pull and receive information from nearby infostations, effectively extending their range. The vehicles' movement patterns, known through the navigation system, are exploited to allow timely delivery of the content. We envisage that the navigation system will automatically pull relevant content (using the navigation information) or implicit information that is required by the drivers.

The Pull-based framework integrates completely with our Push-based approach (Chapter 3) and our Geographic Opportunistic routing protocol (Chapter 2) to provide one solution for routing and disseminating information in a hybrid vehicular network[4].

In the next chapter we will see how we implemented and tested a prototype of this framework.

[4]Notice that all these mechanisms require the same advertised information between neighbours

5

Implementation and small scale evaluation

In this chapter we present the implementation that incorporates all three core components of our framework 1) GeOpps, our geographic opportunistic routing component (Chapter 2), 2) Push-based dissemination (Chapter 3) and 3) Pull-based dissemination (Chapter 4). Furthermore we describe the results of a small-scale experiment that we performed in the area of Cambridge-UK using a small number of vehicles. These experiments are just a proof-of-concept and the results and observations helped us to tune our simulation evaluation that is presented in the following chapter.

5.1 System Architecture

Our system provides a novel type of interaction between network protocols and the vehicle's navigation system: our dissemination and routing strategies depend on navigation data and, similarly, the navigation system can use the network to automatically receive valuable information.

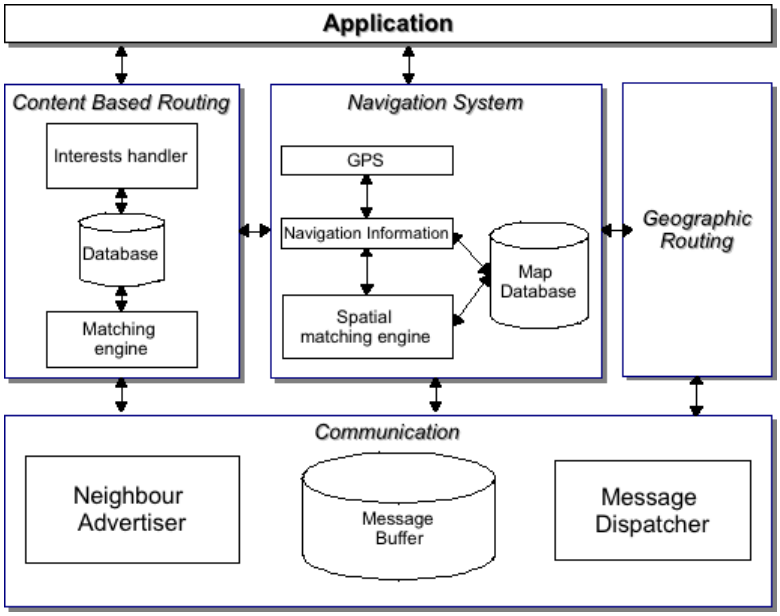


Figure 5.1: System architecture.

Our proposed architecture, shown in Figure 5.1, is composed of a number of components that interact: 1) The navigation system, 2) the geographic routing protocol, 3) the communication/dissemination protocols, 4) the network component and 5) the application that is above our framework. We will now examine these components in detail.

5.1.1 Application

Our framework provides a number of primitives to the application:

- `publish()`: to push information about an event in an area. This primitive, as we analysed in Chapter 3, can take a number of parameters that assign the persistence area, the dissemination time, etc.
- `subscribe()`: to express interests about a set of notifications (e.g., gas prices).
- `unsubscribe()`: to remove interests.
- `handle message()`: the application can specify a call-back method to be invoked whenever matching notifications or the requested data are received.
- `request()`: to pull information from the backbone. Our system will automatically route the request using direct connection or opportunistic routing.

- `route()`: to route information to a specific geographic region. We exposed this method to the application layer just for debugging purposes.

There are also a number of other, less important, functions exposed to the application, however, we will not analyse these here (e.g., to list subscriptions, to perform a local query about whether a subscription matches, etc).

5.1.2 Navigation System

The navigation system is a core part of our architecture. It holds the navigation information of the vehicle (suggested route, GPS position, estimated time required, map database etc.). In addition to these existing functions, we implemented additional methods so as to provide the required API to the other components of our system:

- `spatial match()`: This engine is used to check whether a notification is relevant to a specific location. Our framework is able to treat location as any type of content (e.g., `publish(everybody in` *`London`*`)`) can be mapped to the appropriate geographic locations). Therefore, this module provides the mechanisms to search the map database for a postcode, street name, city name, etc. and match this context to specific geographic regions. Afterwards, it is able to determine whether two locations (given as abstract context) overlap. We use the map to match location to geographic co-ordinates and K-D trees to quickly perform the spatial matching.

- `calculate utility function between path and location()`: This is used by GeOpps in order to route a message (Chapter 2). As an input it accepts the geographical destination of the message and the route of the vehicle. Afterwards, the nearest point and the utility function are calculated with the aid of the map database. Notice that this method is invoked twice when we want to compare the utility between two vehicles.

- `calculate utility between two paths()` : This is used by our pull-based dissemination component to intercept the destination-vehicle's route.

- `return navigation information()`: We have added functionality that allows information to be extracted such as the vehicle's suggested route, estimated time

of arrival, the current speed and bearing, speed limits, intersections on the vehicle's route, GPS location, etc.

5.1.3 Geographic Routing

The geographic routing component is invoked every time the vehicle holds a *replica* in the message buffer or when a vehicle has a message that needs to be routed to a specific *location* or on a *route*. When an advertisement is received by one of the neighbours, this component, with the aid of the navigation system, determines whether this vehicle should keep or forward a stored message.

This module exposes these APIs:

- `route()`: When a message or a replica needs to be routed to a specific geographic location this method registers this information to calculate the vehicle's utility and compares it with any other neighbour. This is exposed directly to the application.
- `route to path()`: This is used when a message needs to be routed to another vehicle's driving path (e.g., to intercept a vehicle as we saw in Chapter 4).
- `keep on path()`: This method is invoked when a message should be kept on a route and moved towards the destination.
- `handle message()`: Used when a new message arrives to be routed. This component decides if this host is the final destination or if additional forwarding is required. It then buffers the message and routes it appropriately.

5.1.4 Content-Based Routing (CBR)

This component supplies the application with the calls to handle a node's interests. Additionally, it determines whether a notification is relevant to this vehicle. Here we implement the CBR matching engine with the help of the navigation system's spatial-matching component. When a new matching notification is received it forwards the message to the application. This module exposes the following methods:

- `notify call-back()`: The application can register a call back for a matching notification.

- `subscribe()`: Handles subscriptions and interests from the application.
- `unSubscribe()`: Handles un-subscriptions.
- `match()`: To evaluate whether the context of a message is matching. This method evaluates whether the vehicle is a subscriber, a non-subscriber or if it was already notified.
- `handle message()`: This is called when a new message arrives through the communication module or the application. This module evaluates whether the notification should be further forwarded down for dissemination or up to the application. Furthermore, if a new notification was generated it allocates the required replicas and hands them over to the routing component.

5.1.5 Communication

This component is responsible for handling the exchanged messages. It automatically advertises the vehicle's interests, routes, and known notifications. It is also used to forward the required messages and replicas with the aid of the routing component and to filter out the incoming information.

This component exposes the following methods:

- `handle CONTROL message()`: When a new neighbour advertisement is received from the network component, this module examines the message buffer to determine if there are any messages that should forwarded to the neighbour. The message dispatcher is responsible to query the upper layers in order to make this decision.
- `add data message()`: Adds a new message to the buffer. This can be either invoked from the network component (i.e., a message received from another host) or from the upper modules (i.e., the content based routing and the geographic routing component). Notice that the message might need further routing (i.e., a replica). Additionally, it will notify the upper layers if a matching notification arrives, or if the message has reached its final destination.
- `query buffer()`: Used to evaluate if a message or a notification has been already received and to receive a list of the buffer's messages.

Publisher ID	P_{id}
Notification ID	N_{id}
Persistence Area	P_{area}
Replicas Left to Create	N_{rep}
Publication time	T_{start}, T_{end}
Message content	m

Table 5.1: Notification message.

Replica ID	Rep_{ID}
Expiration time	T_{end}
Hop Count	$Hops$
HomeZone Location	H_x, H_y
Notification Message (encapsulated)	m

Table 5.2: Replica message. Please note that it encapsulates a notification message.

The message dispatcher module coordinates the handling of all the different types of messages and calls the appropriate modules (CBR, geographic routing, etc).

5.1.6 Network

The network can be any type of wireless network that supports broadcasts to neighbours (e.g., 802.11b). It is forwarding the received information to the communication component to further handle it. Various types of messages can be handled by this component. You can find their content in tables 5.1, 5.2, 5.3, 5.4.

5.2 Implementation

We implemented our framework in C# 3.0 using the Microsoft MapPoint 2006 platform. An application screenshot of our prototype is shown in Figure 5.7. The main window (Figure 5.7(a)) displays the current route of the vehicle augmented with information that was opportunistically disseminated as described in the previous sections. MapPoint provides information such as route waypoints, list of intersections, street names, speed limits, and the estimated time of arrival at each waypoint.

The left side window provides the interface to i) route a message to a specific

Sender ID	S_{id}
Sender Location (used for reply)	S_x, S_y or $Route(< W_1, T_1 >, < W_2, T_2 > ...)$
Receiver ID (can be ANY)	R_{id}
Message ID	N_{id}
Destin. Type (Location, Route)	D_{type}
Destination	D_x, D_y or $Route(< W_1, T_1 >, < W_2, T_2 > ...)$
Destin. Thresh. (if $R_{id} = ANY$)	D_r
Hop Count	$Hops$
Payload Message	m

Table 5.3: Routing Message.

Vehicle ID (can be random or temporary)	V_{id}
Vehicle's Route	$Route(< W_1, T_1 >, < W_2, T_2 > ...)$
Interest/Subscription set	$< S_1, S_2, ... >$
List of Known Notifications	$< N_1, N_2, ... >$

Table 5.4: Vehicle advertising Message.

location (GeOpps), ii) Publish (push) information, iii) pull information by firstly routing the request to a known location (infostation) and later receiving the reply back on the vehicle's route. Finally, it allows the user to define interests (subscriptions).

For experimental purposes, we enable the user to select a picture from the local file system or take a new picture by means of the connected webcam (emulating the behaviour of a vehicle updating pictures to an infostation), or just typing the text that needs to be disseminated. The user also defines the POI and dissemination areas. Similarly, we allow the user to request information for a specific location or subscribe to receive information about the vehicle's route etc.

The calculation map window (depicted in Figure 5.7(b)) shows insights about the forwarding process. For each message in the buffer, the system computes the utility function of the current host and compares it with the utility function values of every neighbour, based on the route piggybacked in the neighbour's advertisement. In the example reported in Figure 5.7(b), both the host and the neighbour are located in position 1 while the message should be delivered to position 3. In this case, the current host has a better utility (i.e., a lower Minimum Estimated Time of Delivery) than the

neighbour because its route exhibits a larger overlap with the message's one. This is shown in the two bottom windows where the dark line represents the route of the host (resp. the neighbour) whereas the light line indicates the ideal route of the message. Hence, the host should keep the message until a better neighbour, i.e., one with a lower utility, is encountered.

Location information are provided by the GPS component. To this end, we implemented a simple NMEA parser that reads data from any serial port (most USB or Bluetooth receivers support this). We extract all possible GPS information like location, bearing, speed, and global time. We use this information both to update the vehicle's location in the Navigation System and to provide accurate location and time information to the framework.

As for network communication, we relied on an unmodified TCP/IP stack and therefore any 802.11a/b/g card can be used. While more efficient and tailored solutions (e.g., the upcoming 802.11p MAC layer) could be put in place, our aim was to show the feasibility of the approach, rather than providing a fully-fledged implementation. Yet, as discussed in the next section, the results we obtained from our prototype are good enough to make IEEE 802.11g with off-the-shelf network cards a viable option even for a market release.

To advertise the route of the vehicle and its interests, periodically every node transmits a UDP packet. Nearby vehicles receive the advertisement and evaluate which packets in the buffer are to be transferred (either because a subscription is matching, or because a replica/message needs to be routed due to a better utility value).

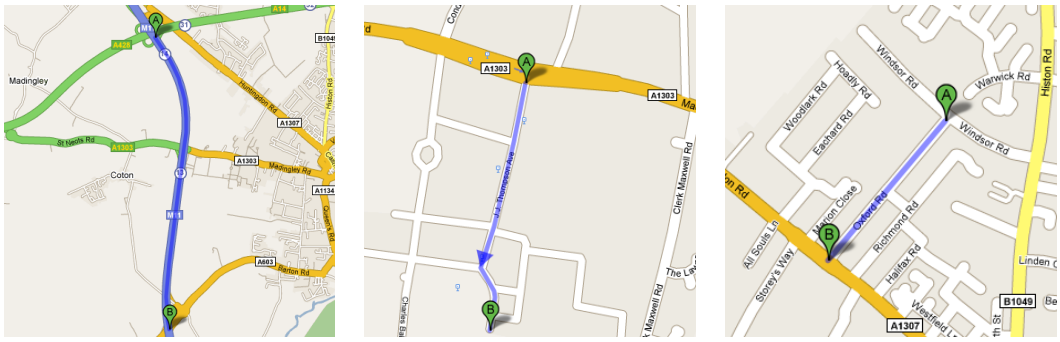
Selected messages are then transmitted using TCP unicast. If the TCP connection is lost we try to re-establish the connection up to three times. If the connection is still not possible the transmission may continue only if another advertisement is received.

5.3 Experimental Evaluation

In order to assess the actual feasibility and efficiency of our approach, we evaluated the implementation described in the previous section in different road scenarios. Unfortunately, due to the intrinsic difficulties of setting up a large scale vehicular test bed, we restrict our analysis to measure the exchange performance when two vehicles encounter

System	Intel Core 2 Duo 2.4Ghz Laptops
Operating System	Windows XP SP2
Radio Type	802.11a
Radio Channel	Pre-assigned to 52 (modified driver)
Channel Scanning	Disabled (modified driver)
Atheros Sleep Timers	Disabled (modified driver)
Rate Adaptation Algorithm	Default Atheros
IP	Pre-assigned (no DHCP)
BSSID	Pre-assigned (modified driver)
Antenna	External Omni-Directional 3db
GPS	external NMEA USB/BT Receiver
Message Buffer Limit	Memory Limited to 2GB
Route Advertise Interval	1 second
Advertise Msg: Max Number of Subscriptions	30
Advertise Msg: Max Number of Known Notif.	30
Route Update Interval	10 second
Utility Function Re-Eval. Angle Threshold	10 deg
Utility Function Re-Eval. Time Threshold	20 second
Utility Value α	0.5

Table 5.5: Experimental Settings.



(a) Highway (120km/h). (b) Urban (50km/h). (c) City (30km/h).

Figure 5.2: Testbed locations.

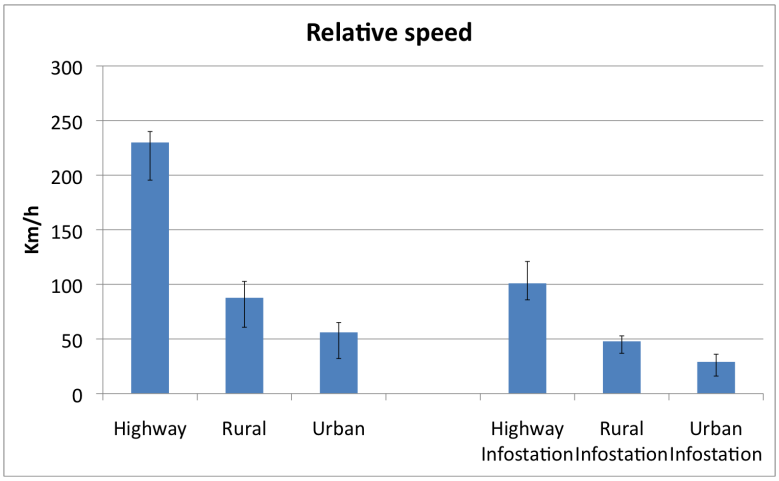


Figure 5.3: Relative speed.

each other. The correctness and efficiency of the protocol in large-scale settings are instead thoroughly evaluated by simulation in the next section.

In our experiments we leveraged off two cars equipped with Dual Core Intel based laptops running Windows XP with Atheros 802.11 cards and external GPS receivers. The laptops, operating in ad hoc mode, had pre-defined IPs (no DHCP) and network settings. You can find detailed settings in Table 5.5.

We selected three different locations in order to represent three recurring scenarios in vehicular networks: highway, urban and city (see Figure 5.2). For each location we designed two different experiments: one with the two vehicles going in opposite directions and the other with a static laptop, representing an infostation, and a mobile vehicle. Each experiment was executed multiple times and median results are presented here.

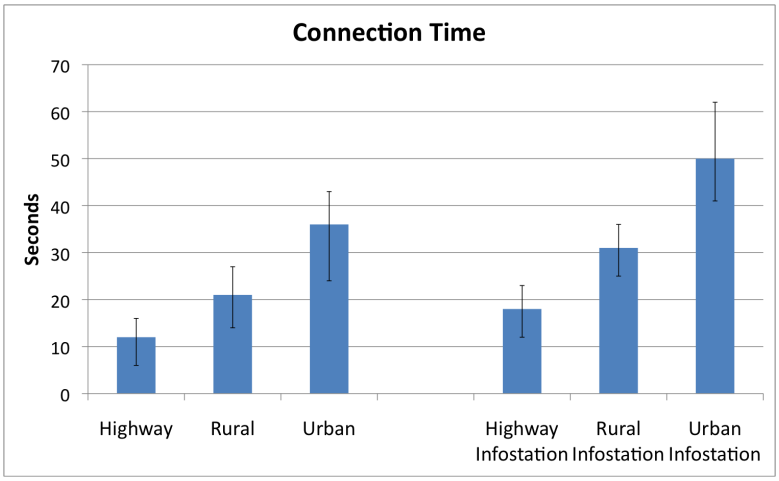


Figure 5.4: Connection Time.

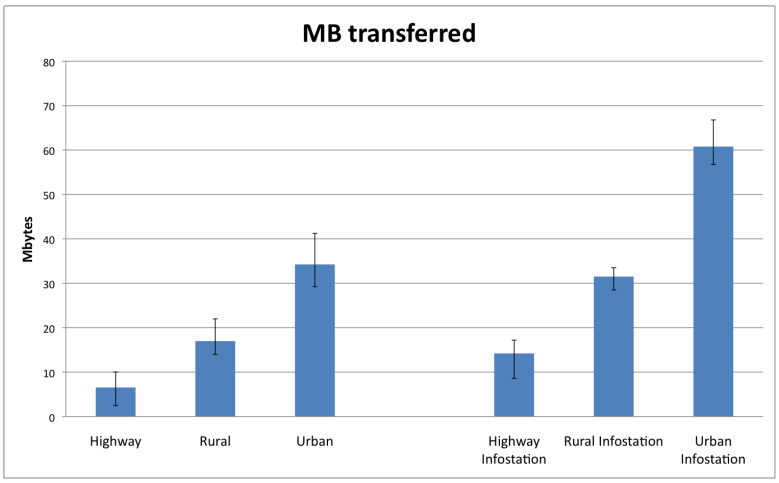


Figure 5.5: Transferred Data.

This represents the worst case for our protocol as the connection between the two vehicles lasts only a handful of seconds. Conversely, in the most favourable case, the two vehicles are proceeding in the same direction and, hence, the connection can last for several minutes, thus allowing for much larger exchange.

For each of the scenarios we created a number of notifications about various locations. We pre-selected these locations so that when the vehicles meet a large number of messages should be evaluated and forwarded (i.e. there are always enough data to forward).

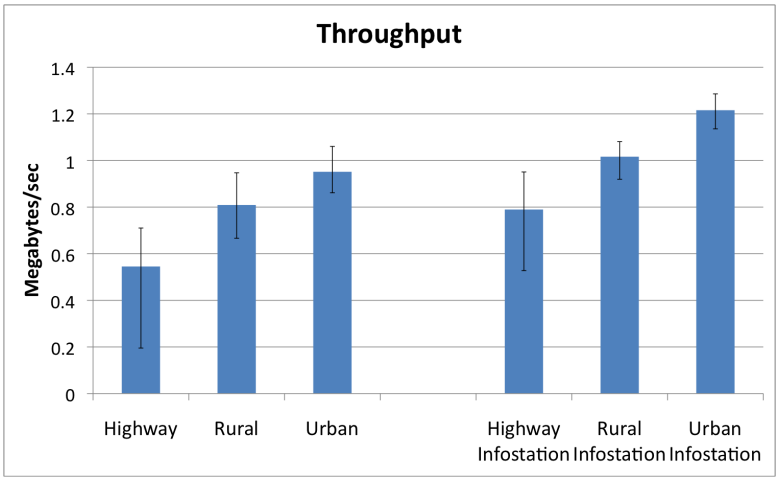


Figure 5.6: Average Throughput.

5.3.1 Highway

Our first experiment was carried out on a main highway with the two vehicles travelling at speeds up to 120km/h which yielded an average relative speed of 229 km/h and an average connection time of 13.1s in the vehicle-to-vehicle (V2V) experiment and 18s in the vehicle-to-infostation (V2I) one (see Figures 5.3 and 5.4).

This represents a very challenging scenario because the channel coherence time is very short and the rate adaptation algorithms are often taking the wrong decision. In fact, we noticed that our Atheros driver frequently overestimated the broadcast rates leading to lost packets and TCP ACKs that required re-initialisation of the TCP connection.

Nonetheless, as reported in Figure 5.5, the amount of data exchange is still significant. We could transfer 6.5MB per pass in the V2V experiment and 14MB in the V2I configuration.

Note that these results also account for the time required to perform the initial content matching and evaluate the utility functions. The setup time (the time required between receiving the first advertisement and starting to transmit the first packet) was 0.83s. Apart from the setup time, no further delay due to processing was noticeable because these calculations were performed in parallel to the transmission.

5.3.2 Urban

For the urban scenario we selected a suburban area with very few buildings that are far from the road, thus minimising the impact of interference and fading effects. The average relative speed between vehicles in our V2V experiment was 85 km/h while 46 km/h was the average speed in the V2I experiment (the road-traffic conditions were average). Connection times are significantly higher than the highway scenario reaching up to 31s for the V2I. This favourably increases the volumes of data exchanged, allowing to transfer 17MB between vehicles and 34MB to the infostation.

5.3.3 City

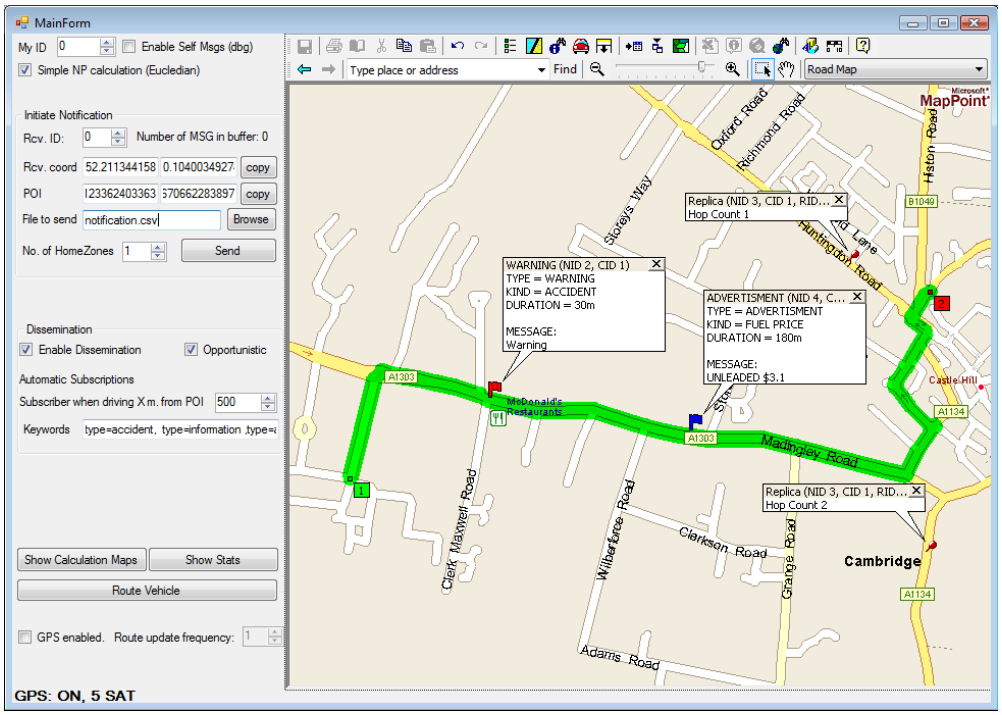
Finally, we evaluated our approach in a busy and densely populated road with two storey terraced houses on each side of the road. The average relative speed during our experiments was 51km/h between the two vehicles and 26km/h with the infostation resulting in connection times of 31 and 51 seconds respectively. The very low speeds and the reflections from the houses led to an impressive volume of transferred data. Vehicles exchanged 31MB and managed to transfer to the infostation 56MB.

5.4 Conclusions

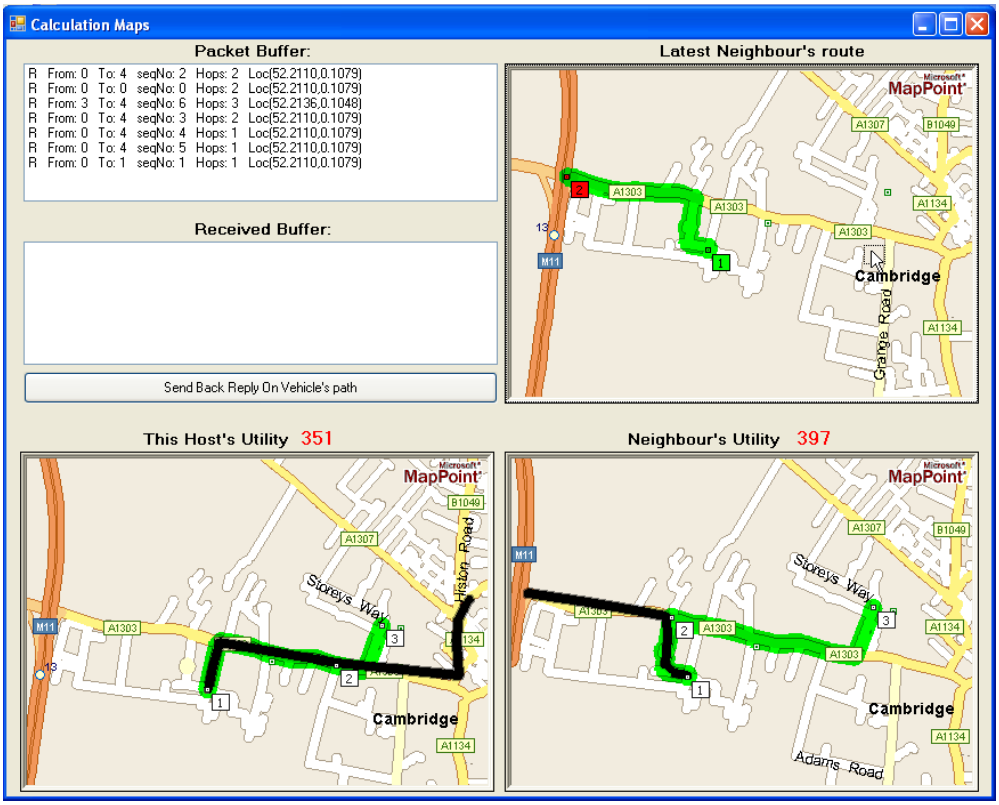
We have designed and implemented our framework using .NET and Microsoft Visual Studio. This system provides a proof-of-concept and a powerful platform for the development of network protocols and applications that can interact with the navigation system to get navigation information, exploit the map database, and acquire accurate location information using GPS and the network at the same time.

We tested our framework in six possible scenarios that provide typical examples of vehicle interaction. These tests allow us to get an intuition of the technical issues that might arise in a real implementation and gain some experience about the transfer speeds that we might be able to achieve when our system is used. The performance is quite satisfactory and establishes the plausibility of such a system.

In the next chapter we further examine the correctness and the performance of our framework in a simulated large scale environment. We transferred the experience gained from these experiments to tune our simulations to make them as realistic as possible.



(a) Navigation/Application window.



(b) Utility calculation window.

Figure 5.7: Prototype implementation.

6

Large Scale Evaluation

In this chapter we discuss the methodology followed to prove the validity of our approach and evaluate its performance in *large scale settings* over several simulated scenarios, generated from realistic vehicular traces.

Simulation offers the opportunity to test our protocols using a large number of simulated vehicles, in different scenarios and under different parameters. Furthermore, it is a widely used and established method for the evaluation of MANETs.

We used the experience gained from the real implementation to make the simulations as realistic as possible i) by tuning the 802.11 radio settings, 2) by avoiding making unrealistic assumptions (for example about what information each vehicle can access, about how accurate it is, or about how quickly some calculations may be performed). Furthermore, we use realistic mobility models in the simulation to add credibility to the evaluation scenarios.

6.1 Simulation

In this section we present the simulator tools that we used, the mobility patterns of the simulated vehicles, our methodology and simulator settings.

6.1.1 Simulator

To simulate our framework and protocols realistically, we used *OMNet++* [Varga 01, OMNET++ 09], which is a popular discrete event simulation environment. Its primary application area is the simulation of communication networks. OMNet++ provides a component architecture where components (modules) are programmed in C++, then assembled into larger components and modules that communicate by message passing through gates and connections. It provides advanced user interfaces that visualise the model and allow control over simulation execution. These interfaces also facilitate demonstration of how a model works. Additionally, the core framework provides basic modules that can be extended in order to implement our own modules (e.g., application layer, middleware, mac layer, etc.). Furthermore, we used the *mobility framework plug-in* [Drytkiewicz 03] which supports node mobility, dynamic connection management and a wireless channel model, that are essential to simulate a hybrid vehicular network.

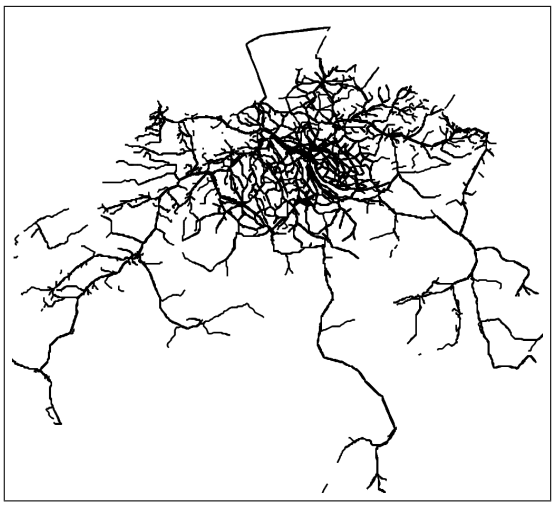
We selected OMNet++ for several reasons. First, it is flexible enough to simulate any network environment. Its modular architecture allows us to implement and simulate from a simple algorithm to a complete middleware framework. Moreover, the graphical interface, the statistics libraries and the C++ programming language enabled us to easily set-up any simulation scenario and collect the results. Additionally, there are a lot of plug-ins that can be used for the simulation (like the mobility framework, parsers for mobility models etc.). Finally, it is open source and free for academic use.

In order to accurately evaluate our protocols in the context of vehicular networking, it would not make much sense to use any random mobility model [Camp 02]. We have evaluated our approach by using traffic traces generated by a multi-agent microscopic traffic simulator (MMTS) developed by K.Nagel at ETH, Zurich [Naumov 06, Traces 09]. MMTS models the behaviour of people living in the area, reproducing their movement. Travel plans are based on road congestion, which in turn depends on travel plans. These traces contain mobility patterns of 260,000 vehicles over real road maps

(a) City (3x3km).

(b) Urban (3x3km).

(c) Rural(3x3km).



(d) MMTS traces for Zurich and surrounding highways.

Figure 6.1: Simulation scenarios.

in the canton of Zurich within a period of 24 hours. Furthermore, they contain dense populated areas (the city of Zurich) and the surrounding highways, which enable us to run our simulations in different settings (Figure 6.1(d)).

Additionally, we used the GMSF generator [GMSF 09] to produce GIS traffic-light traces for the rural, urban and city scenarios, which have finer granularity (3x3km). Although these traces are not based on realistic people behaviour, they offer much higher granularity by providing mobility information on much more detailed maps.

Consequently, the scenarios considered are:

- *City scenario*: High vehicle and street density scenario where up to 880 vehicles are concurrently present (default 700). Average speed is 20km/h and maximum is 60km/h. An example of this is provided in Figure 6.1(a).

  Furthermore, we used a larger 50x50km scenario that contains the centre of Zurich (part of Figure 6.1(d)). In this scenario up to 2000 vehicles are in the simulation area at the same time and more than 25.000 vehicles participate in each experiment. We mainly used this scenario to evaluate our routing protocols as it allows routing of information to distant areas.

- *Urban scenario*: Medium street and vehicle density. 420 Vehicles are present at the same time (default). Maximum speed is 60km/h but average speed is 25km/h (Figure 6.1(b)).

- *Rural scenario*: This is a low density scenario where only 100 vehicles are concurrently present. In the scenario of Figure 6.1(c). Average speed is 28km/h (max is 60km/h).

- *Highway scenario:* This is a much larger 50x50km area as illustrated in Figure 6.15(m). We extracted this scenario by selecting a highway from the Zurich traces 6.1(d) and different times of the day. The simulation includes a default 830 concurrent vehicles. The average speed is higher than the previous scenarios (93km/h) and the highest 120km/h.

6.1.2 Simulation Settings

Although some settings change from one experiment to the other (depending on the parameter that we evaluate each time), here we will present our default settings:

First of all, we use the 802.11 [IEEE 03] wireless radio interface. We used this radio interface because it is similar to the new 802.11p standard (Wireless Access for the Vehicular Environment or WAVE) and because it is already widely used both for simulation and in real life. Furthermore, we set the TX power and the attenuation settings in order to match the measured connection distances and times that we observed during the implementation testing. More specifically, the average range where communication is possible is 250m in every direction (we only set the TX power and the antenna gain on the simulation). Additionally, all the broadcasts occur on the same frequency channel.

OMNet++ is responsible for handling any collisions and retransmissions. Additionally, the position of each vehicle is updated every 1s (based on the traces). The default TTL for the messages is set to 1800sec (30min) and the default neighbour advertising interval is 10sec. Finally, the default α value is 0.5 which, as we will show, strikes a good balance between making robust carrier choices and minimising delivery delay.

For the individual settings of the routing component, the push and pull based dissemination, refer to Sections 6.2, 6.3 and 6.4 respectively.

6.1.3 Simulation Goals

The goals of our simulation can be summarised below:

- Measure the performance of our protocols in terms of delivery ratio, delay, resource consumption, etc.
- Measure the sensitivity of our protocol for various settings (density, number of vehicles, dissemination area sizes, number of infostations, number of vehicles with suggested routes, mobility patterns, etc.).
- Compare our results with existing solutions.

To make the simulation results statistically significant we simulated the same settings at least 20 times and averaged the results (more if there was high variance).

6.1.4 Assumptions

There are some assumptions that we made when we designed these simulations. First, we assumed that hosts can potentially broadcast a message to all their neighbours within a given range: the range depends on the signal to noise ratio but there were no obstacles or directional antennas. Furthermore, there are no hardware failures, nodes that become offline due to power depletion. There were no malicious nodes or non-cooperative nodes. we also assume that hosts can acquire their geographic location with reasonable accuracy (e.g. $\pm$10 meters) as we also verified from our implementation.

In the following three sections we will present our results. We run independent simulations for each of the three components of our framework. This allows us to

evaluate their performance more accurately as it allows us to focus on each protocol separately.

Therefore, in Section 6.2 we present the results of our geographic opportunistic protocol presented in Chapter 2. In the following section (Section 6.3) we show the results of our push-based dissemination protocol (Chapter 3) whereas in Section 6.4 we analyse the performance of the pull-based approach (Chapter 4).

6.2 Simulation Results: Geopps

To evaluate GeOpps, apart from implementing the routing protocol in the simulation environment we also emulated the navigation system. We compare our protocol with two other approaches: *Location-Based Greedy* routing and the *MoVe* routing algorithm:

- *Location-Based Greedy:* A DTN variation of existing location-based greedy algorithms [Zhao 04, Mauve 01] where the packet is forwarded to the neighbour that is closest [1] to the destination (if closer than the position of the current carrier). This process is repeated until the message reaches its destination. The messages are stored in a infinite buffer, until they expire. When local minima are reached the vehicle either keeps the message (until its mobility patterns help to escape) or we can use GPSR [Karp 00] to go around the obstacle. We have implemented both these methods for our simulations but we present the results with perimeter mode enabled, as it achieves better delivery ratio in high-density scenarios.

- MoVe [Lebrun 05] uses information about relative velocities of the current vehicle and its neighbours to predict the closest distance that the vehicles are predicted to get to the destination, following their current trajectories (straight-line paths). More specifically, we measure the angle θ_a between the current trajectory of a neighbour a and the destination D. The neighbour that minimises this angle is given the packet. Similarly, an infinite buffer is used to store messages until they expire.

The notification packet payload size is set to 10Kb and the route advertise message is 60 bytes. The mobility framework also adds the 802.11b broadcast headers to these

[1] In terms of Euclidean distance

messages.

During the simulation, 1,000 random vehicles are selected and from each, a packet is sent through each of the three protocols to the same destination D. These packets are then routed using the three algorithms. We measure the delivery ratio, hop count and delay. To calculate the nearest point and evaluate the utility we use the simplified version described in Section 2.2. Vehicles always follow their suggested routes and poll their neighbours every 5 seconds. The results that we present are averages of 20 runs.

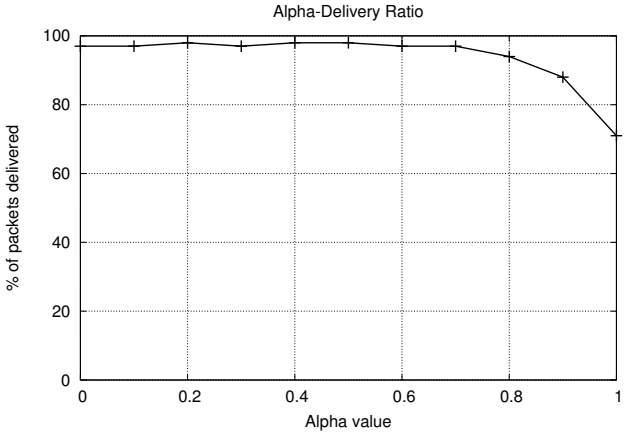


Figure 6.2: Delivery ratio for different α values.

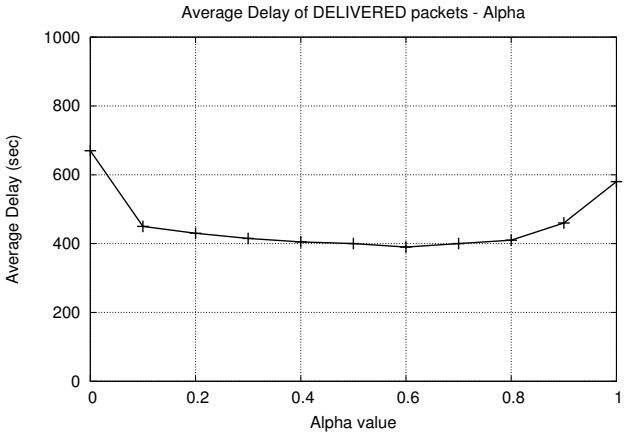


Figure 6.3: Delivery delay for different α values.

Before evaluating GeOpps we will examine how the α parameter, used in our utility function (Section 2.2.2), affects its performance. In Figure 6.2 we show the delivery ratio for different α values, ranging between 0 and 1. When α is 0, we only take into account how close the candidate message-carrier will drive compared to the packets destination D. As expected, in this case the delivery ratio is very high, as we make solid choices, by

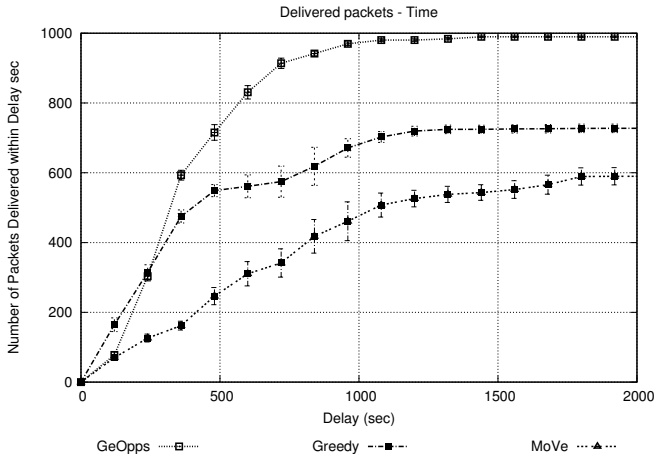


Figure 6.4: Delivery ratio through time.

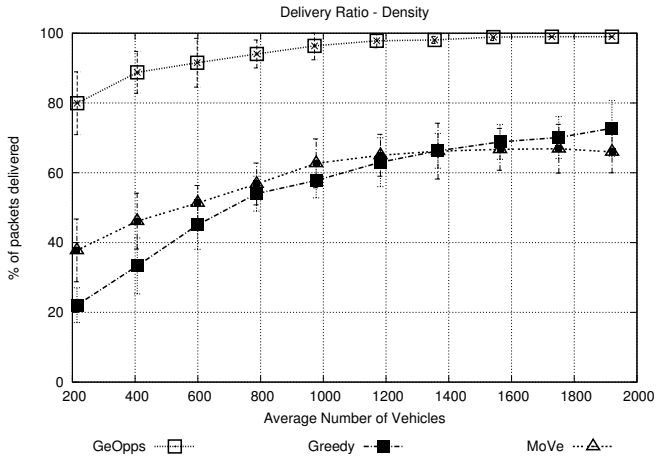


Figure 6.5: Delivery ratio for different densities.

preferring carriers that will drive closer to D. However, we completely ignore how fast they will get there (e.g., they may select a sub-optimal route or they might be driving very slow). On the other extreme, when α is 1, we only examine whether the driving distance between the carrier and D is minimised. In other words, we only evaluate if the carrier is on the optimal path and not if it will eventually drive close to D (similarly to VADD [Zhao 06]). Therefore, we believe that using $\alpha = 1$ is not a good choice in terms of delivery ratio.

In terms of delay, as we see in Figure 6.3 , when α is close to 1 we observe a drop in performance (higher delay) due to the fact that making less robust choices is causing the messages to delay in reaching their destination. There is a special case when $\alpha = 0$: only the distance between the nearest point NP and the messages destination D is

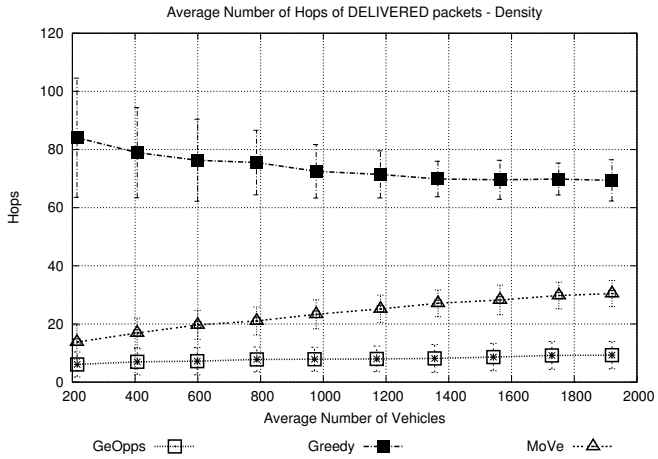


Figure 6.6: Average number of hops for different network densities. Smaller is better.

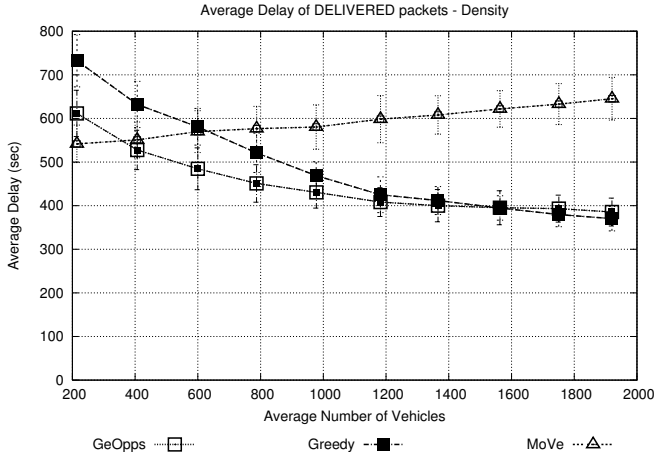


Figure 6.7: Average packet delay for different network densities. Smaller is better.

considered. In this case the messages do not hop forward when more vehicles share the same NP and it will be only routed using the carriers mobility speed, unless a vehicle travelling closer to D is found. In our experience, values between 0.3 and 0.7 seem to exhibit the best results in terms of both delivery ratio and delay. And this is why we choose the value 0.5 for our following experiments as an effort to strike a balance between selecting a robust carrier that might be sub-optimal in terms of delay, or a less-robust carrier that drives via the shortest path.

Figure 6.4 illustrates the cumulative number of packets delivered within a certain time after sending. More specifically, we send all the messages at the same time (from various locations), and we measure how many messages where delivered after the evaluated time period.

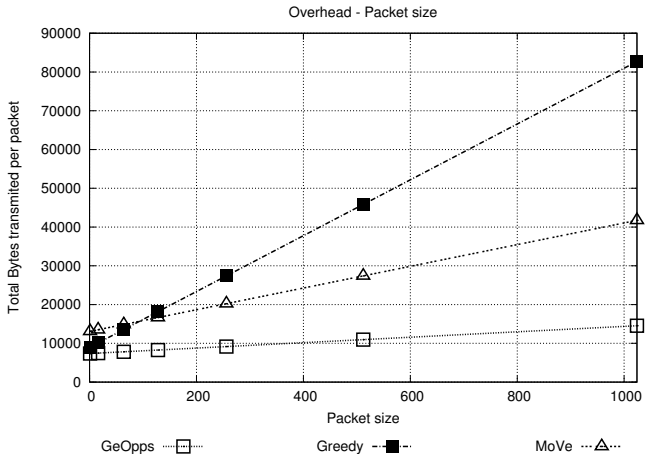


Figure 6.8: Overhead for different packet sizes.

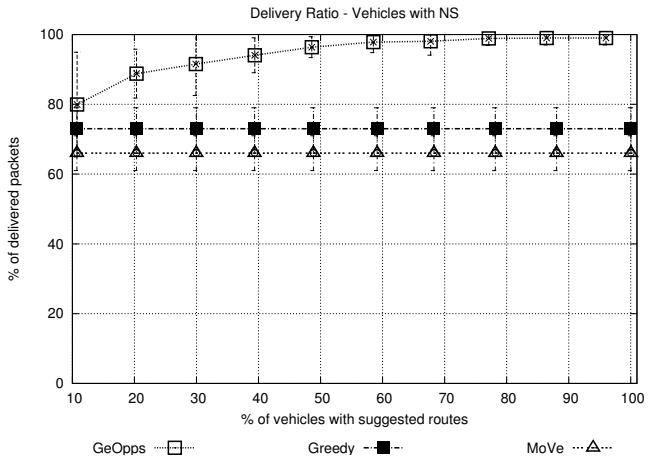


Figure 6.9: Delivery ratio for different percentages of vehicles that shared navigation information (penetration of navigation systems).

We observe that GeOpps is able to deliver nearly 98% of the packets within twenty minutes in the large-scale scenario. At the same time, Greedy delivered 72% of the packets whereas MoVe 53%. These results indicate that GeOpps can deliver the vast majority of the packets to the final destination. MoVe shows poor performance due to the fact that the current trajectories of the vehicles do not actually indicate their final destination because vehicles have to follow the road topology. Greedy delivers some of the packets quickly (mainly packets generated near the destination) but the total delivery ratio is only 73% because of the highly partitioned (and mobile) vehicular network (messages sent from remote -not directly connected- areas were not delivered). In fact, GeOpps delivers 73% of messages earlier than greedy. Also note that initially, Greedy delivers packets faster than GeOpps (the first 250 packets). This occurs because

these are packets that are generated near the destination that greedy is able to quickly deliver whereas in our case GeOpps prefers more reliable, but slightly slower, forwarding decisions by only selecting vehicles that report that they will actually drive closer to the destination D (and this is why GeOpps is able to eventually deliver the majority of the packets).

In Figure 6.5, we have plotted the delivery ratio of the algorithms for varying densities (TTL is 1800sec). Greedy shows acceptable performance only in dense networks (peak-time) due to the fact that it requires the presence of neighbours that are closer and closer to the final destination. In fact, MoVe outperforms Greedy in sparse road traffic conditions where trajectory information is more important than the position of the neighbours. However, GeOpps is able to outperform both algorithms in any network condition. It is adequate to find only one vehicle that will carry the message to its destination and thus, it is not required to have very frequent encounters like Greedy and MoVe. More encounters increase the probability of finding an ideal carrier.

We can further support this observation by evaluating the number of hops required to deliver a message, shown in Figure 6.6. We notice that the number of hops required for Greedy is much higher than for the other two algorithms, because it constantly attempts to forward the message to neighbours that are closer to the destination. However, GeOpps requires only a few encounters before finding a vehicle that drives near to the destination of the packets. Furthermore, this number does not depend on the density of the network but only on the road topology (e.g., the probability of finding a vehicle that is going close to the destination of the packet in this road segment).

Additionally, Figure 6.7 depicts the average delay of *delivered* packets. As we can see, the delay of our algorithm is lower than that of the other two algorithms which is another indication that the minimisation of utility value is effective. Furthermore, the delay drops as the density increases because the probability to find a better carrier is higher and because the packets hop to leading vehicles as we discussed in Section 2.2.3. MoVe delay increases because in low density situations the direction information is more important than location (thus MoVe is better than Greedy). As density increases more and more messages are delivered even from further away resulting in higher delivery delay. Greedy requires a dense network to deliver messages. The higher the density the less messages are likely to be trapped in areas and the larger distance they cover per

hop.

Figure 6.9 indicates the delivery ratio for different penetration of navigation systems. When we compare the delivery ratio of Greedy and MoVe using 2000 vehicles [2] to GeOpps using only 200 vehicles (10% of the drivers use their NS), we notice that GeOpps still delivers more packets (about 80% compared to 70% of Greedy). This occurs due to the fact that GeOpps uses Greedy as a failsafe method (when no navigation information is available) and further improves delivery when there is.

Finally, Figure 6.8 demonstrates the transmission overhead of the messages for various packet sizes including overheads. As we can see the message overhead of Greedy is high due to the fact that packets require a high number of hops before delivery. The results indicate that our algorithm is able to deliver almost 99% of large packets with less than one fifth the overhead of Greedy.

6.3 Simulation Results: Push-Based Dissemination

For this scenario we disseminate a number of notifications considering one specific location of the map (POI). We analysed our protocol under a synthetic load of both automatic and custom subscriptions. In particular, for automatic subscriptions, all vehicles with planned routes intersecting the POI are considered subscribers. This is the typical situation with traffic warnings, which are of interest to any vehicle en route towards the affected destination. Conversely, custom subscriptions (e.g., hotel or restaurants) are not relevant for everybody but will involve only a fraction of vehicles travelling towards the POI.

In the default configuration the advertise interval is equal to 10 s and we have 10 replicas. Each simulation lasts for 2 hours of simulated time and results are averaged over multiple runs.

To put our work in the context of related efforts and to capture the trade-offs involved, we compared our solution with an epidemic approach, reminiscent of [Vahdat 00], in which all nodes store each message received and re-broadcast it to all neighbours, which have not heard that message yet.

[2]Notice that Greedy and Move do not require any navigation information (just the position/bearing of the vehicle). Therefore we assume that all the 2000 vehicles participate

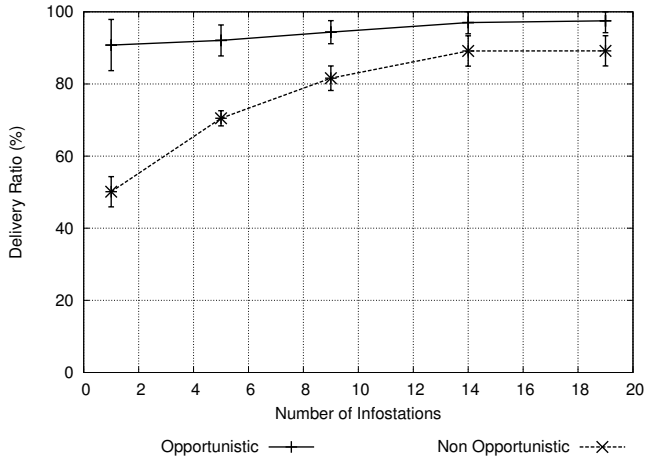
Hereafter, we will first present results achieved in the city-base scenario, with and without infostations, as this represents the more challenging case for our protocol, given the complex road topology. Then, we will show the performance obtained in the urban, rural, and highway scenarios to demonstrate the suitability of our approach for different environments.

In all our experiments, we measured the *delivery ratio*, expressed as the fraction of subscribers that successfully received the messages; and the *network overhead*, defined as the number of transmissions received per minute by each vehicle.

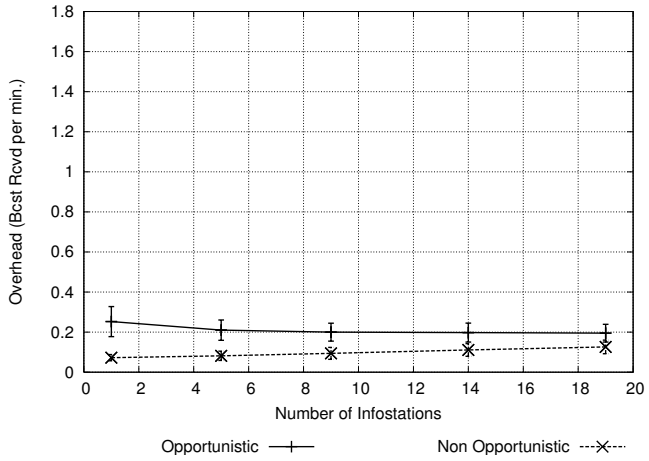
Infostations: As a first experiment, we focus on a fully infrastructure-based scenario in which the persistence area is instrumented with several infostations. Our goal is twofold: on one hand we want to demonstrate the correctness of our protocol and on the other hand we want to assess the impact of the additional opportunistic dissemination in such a scenario (where any vehicle that heard the information can further spread it). To this end, in Figure 6.10(a) we measured the delivery ratio of our protocol under two different configurations, i.e., with and without opportunistic dissemination.

Remarkably, through the opportunistic dissemination introduced in Section 3.3.2, delivery is above 90% even with just one infostation. On the other hand, if opportunistic dissemination is not used, at least 14 infostations are needed to achieve similar performance. This is a prominent result as it proves that even in a fully infrastructured environment, opportunistic dissemination represents an improvement to our approach. Indeed, although the network overhead does not change with the number of infostations (see Figure 6.10(b)), still resorting to opportunistic dissemination enables the reduction of the number of infostations, thus simplifying their deployment.

Ad-hoc: Despite the above results, assuming a widespread availability of infostations is unrealistic in many scenarios. Hence, to ensure efficient content-based dissemination in hybrid scenarios, as those targeted in this paper, it is fundamental to support infrastructure-less communication. In our work, this is achieved by means of the *ad-hoc* persistence solution, described in Section 3.3.3. To avoid any bias and to isolate the contribution, in the rest of this section we assume that no infostation is present and that all communication relies on vehicle-to-vehicle technology. In the case of semi-infrastructure environments, we can have an interplay of the two approaches.



(a) Delivery Ratio (high density).

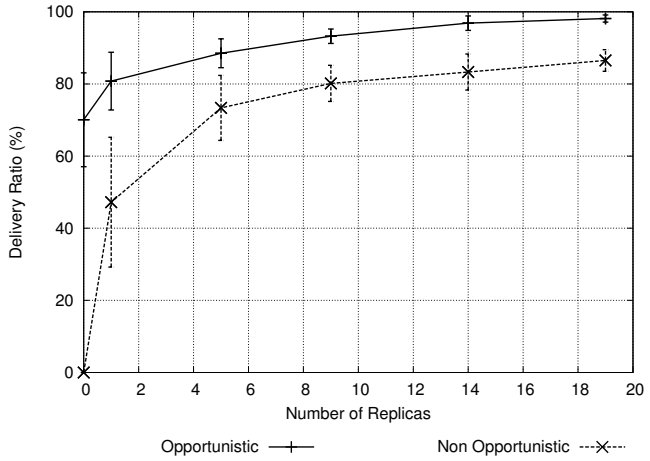


(b) Overhead (high density).

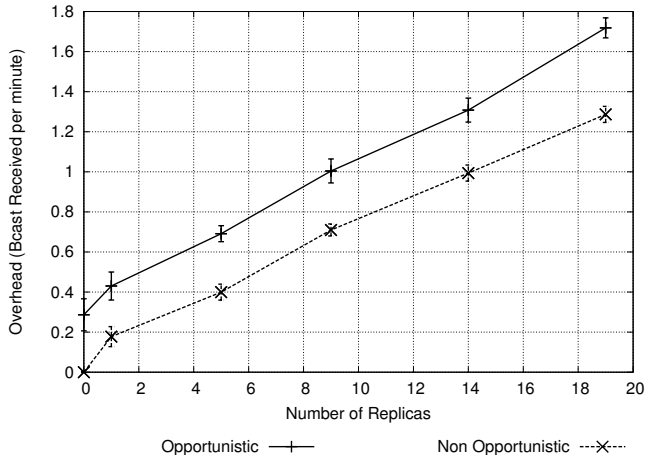
Figure 6.10: Number of Infostations.

Number of Replicas: The first parameter we explore is the number of replicas created to guarantee the persistence of the message within the specified area. Also, as we did in the infostation scenario, to assess the impact of the opportunistic dissemination, we run two different versions of our protocol: the former relying only on replicas to disseminate messages and the latter exploiting also the opportunistic routing. Since results strongly depend on the density of vehicles, we tested it both in a low and high density scenario (200 and 700 vehicles).

Results in Figure 6.11(a) confirm our claims. When the density is high, even a small number of replicas is sufficient to achieve a high delivery. Interestingly, however, this result is due to the combination of two different strategies: the ad-hoc persistence and the opportunistic dissemination. Indeed, when the opportunistic dissemination is



(a) Delivery Ratio (high density).

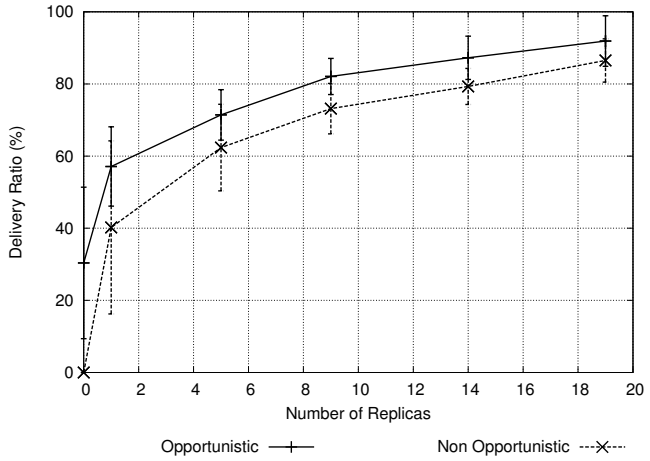


(b) Overhead (high density).

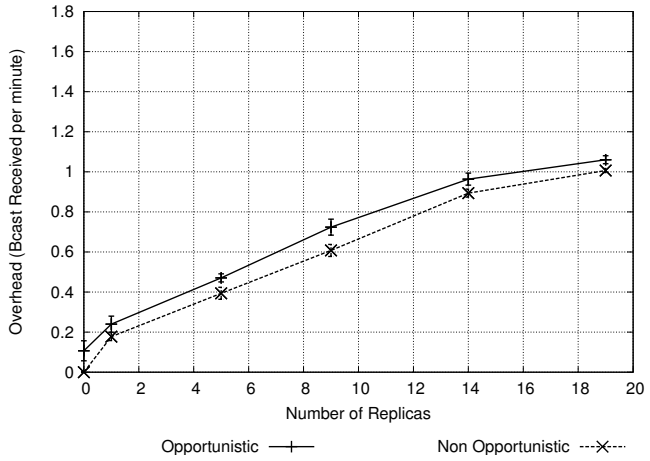
Figure 6.11: Number of Replicas - High Density.

not used, the delivery drops to 50%, unless many more replicas are introduced. This however, as shown in Figure 6.11(b), generates a significant overhead. Indeed, to achieve the same delivery of 80%, 9 replicas are needed without opportunistic dissemination (instead of just 1) with almost doubled overhead (0.7 against 0.4 broadcasts per minute). Notably, the opportunistic dissemination only slightly affects the overhead because most of it is due to keeping replicas in the persistence area. Furthermore, if opportunistic dissemination is not used, even a high number of replicas does not bring significant improvements to the delivery.

In case of low density (Figure 6.12), as expected, the overall improvement provided by the opportunistic dissemination decreases as there are fewer vehicles around. Hence, the main transmissions will occur from replica carriers and this explains why the de-



(a) Delivery Ratio (low density).



(b) Overhead (low density).

Figure 6.12: Number of Replicas - Low Density.

livery is mainly impacted by the number of replicas. Nevertheless, the opportunistic dissemination is still useful because it yields an improvement in terms of *delivery ratio* regardless of how many replicas are used (Figure 6.12(a)).

Looking at these results, one might argue that the main contribution to the message delivery comes from the opportunistic dissemination while the ad-hoc persistence plays only a marginal role. This, however, is strongly contradicted by performance achieved with zero replicas, both in the high density and, especially, in the low density scenarios. Indeed, in the former, opportunistic dissemination alone delivers the message only to 70% of subscribers while in the low density scenarios only 30% of subscribers are notified. This is consistent with the conclusions drawn above: opportunistic dissemination provides a valuable contribution only in dense scenarios while in sparse scenarios it

becomes less useful. Nevertheless, even in dense networks, to get reasonable results, it must be coupled with a persistence strategy since, otherwise, if the message disappears from the area, by no means can later subscribers be notified.

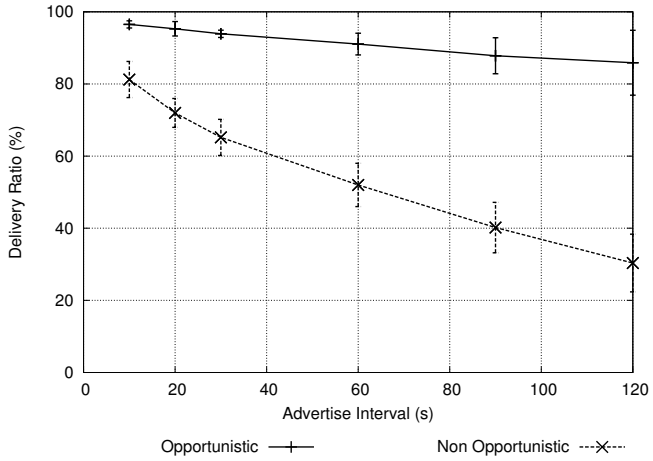
The results in the high density scenario (Figure 6.11(a)) closely resemble the ones with infostations in Figure 6.10(a). Not surprisingly, however, overall performance is slightly worse: This behaviour stems from the fact that now replicas are hosted on vehicles, as opposed to infostations. Hence, even non-subscribers play a key role to ensure proper persistence, by continuously passing replicas from one vehicle to another. Delivery of subscribers is also affected because in some cases, replicas may abandon their homeZones (e.g., because no alternative carriers were found). Consequently, incoming subscribers may miss the notification, thus demanding more replicas to be in place.

Advertise Interval: Advertise interval is a complementary parameter with respect to the number of replicas. If we keep the number of replicas fixed, we can reduce the advertise interval to improve the message delivery. In this way, the probability for a subscriber to miss a replica is lower because subscribers advertise their interests more frequently.

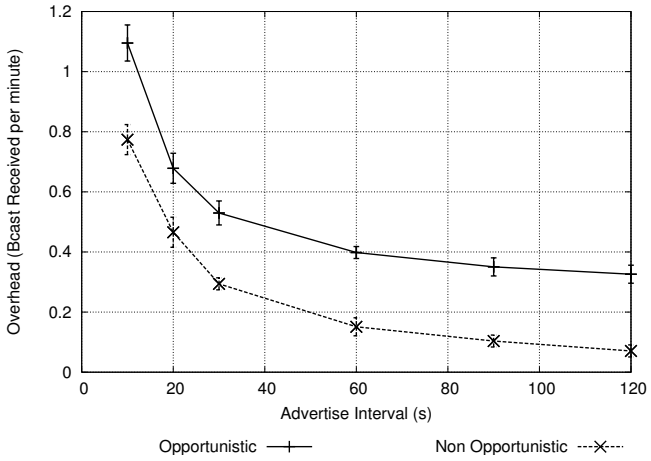
This property is charted in Figure 6.13 in which we studied the protocol behaviour over different advertise intervals. As described above, decreasing the advertise interval is beneficial to the delivery which increases to almost 100% (here we used 10 replicas). Interestingly, the improvement in terms of delivery is more evident when opportunistic dissemination is not used: without opportunistic dissemination, missing a replica is far more critical because the chances to encounter another one are few. Conversely, opportunistic dissemination alleviates this issue since messages can be obtained also from other vehicles and not exclusively from replica carriers.

Note, however, that reducing the advertise interval comes at a cost. Beside incrementing the advertisements per minute, it increments the overall number of broadcasts received. Indeed, given that information about nearby vehicles is more accurate, replicas will hop more frequently from one vehicle to another because better carriers are found. This explains why the number of broadcasts exhibits a steep trend as soon as the advertise interval gets small.

Custom Subscriptions: Thus far, we concentrated our attention only on automatic



(a) Delivery Ratio (high density).

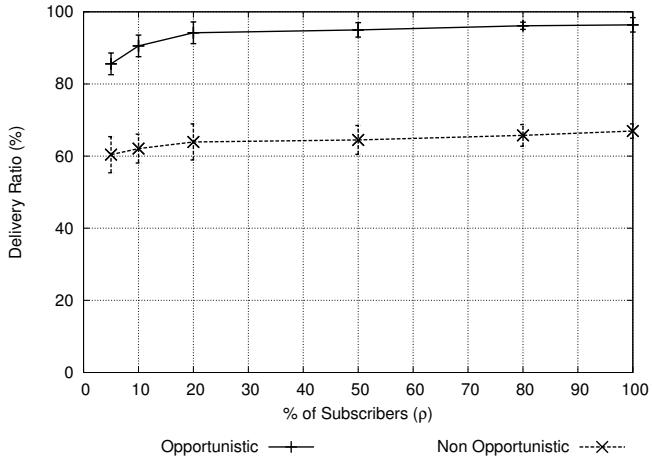


(b) Overhead (high density).

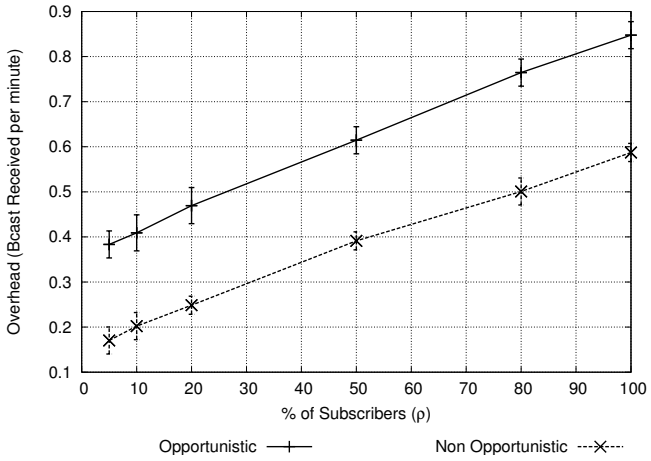
Figure 6.13: Advertise Interval.

subscriptions. Nevertheless, a prominent feature of our approach is the ability to incorporate also drivers' interests, which are not necessarily shared by all other drivers. To model this scenario, we assume that only a fraction ρ of vehicles going towards the POI are interested in the message and we analyse our protocol under different values of ρ (see Figure 6.14).

Remarkably, as reported in Figure 6.14(a) our protocol shows high event delivery, even for small values of ρ. This means that regardless of the fraction of subscribers, our protocol ensures that the vast majority (e.g., 90% for ρ =10%) of them receives the message. Furthermore, we also observe that when there are more subscribers, the message overhead increases. This verifies that low interest messages are spread less than more popular ones (i.e., the spread/overhead depends on the interest about an event).



(a) Delivery Ratio (high density).



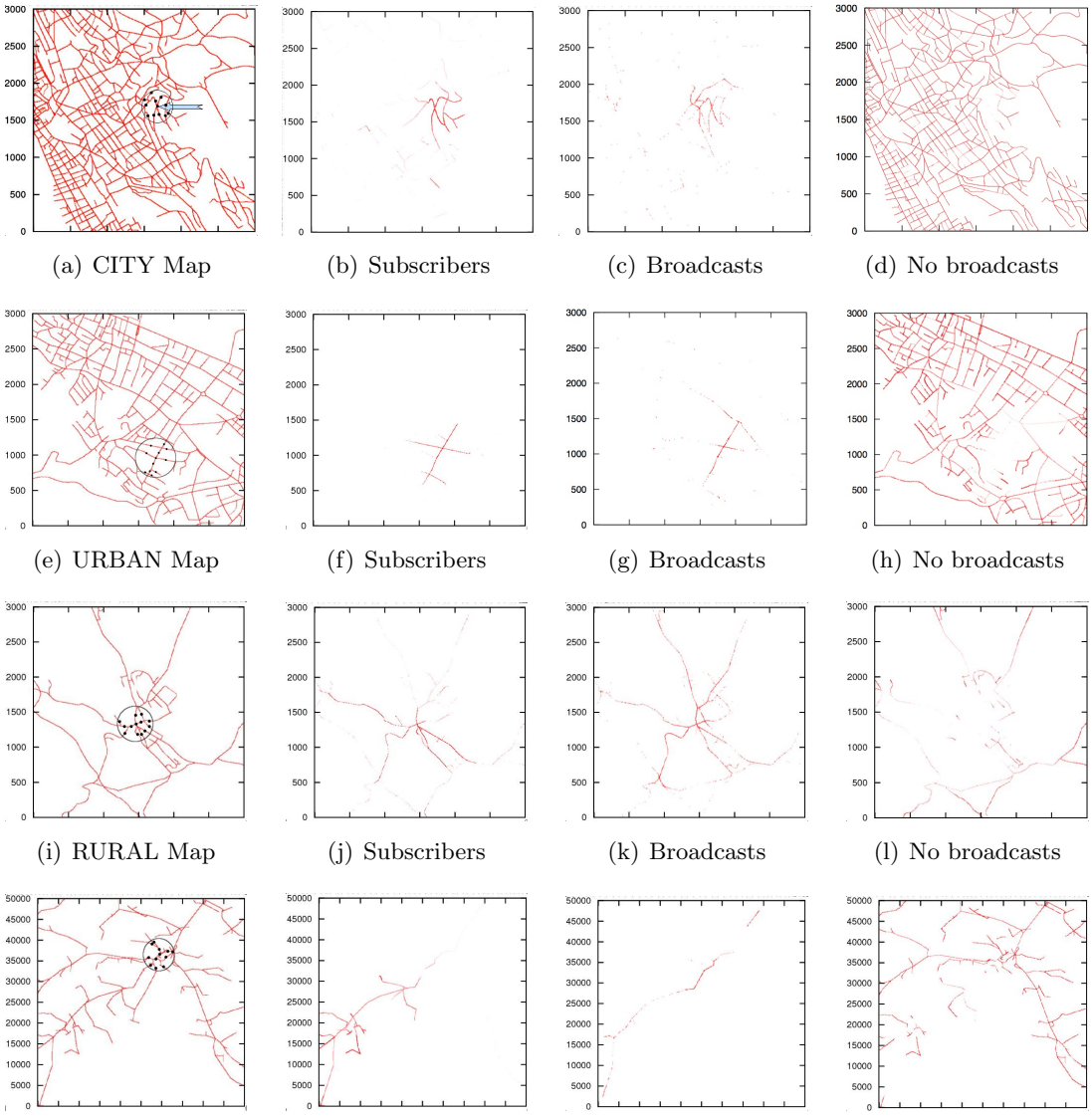
(b) Overhead (high density).

Figure 6.14: Custom Subscriptions.

These charts demonstrate the high flexibility of our protocol, which is able to be tuned to network conditions and to selectively contact almost only intended subscribers.

Distribution: In all previous charts, we focused on the city scenario, since this represented the most challenging test. Nevertheless, to carefully evaluate our protocol, we experimented also with other traces, available at [GMSF 09], representative of an urban, a highway, and a rural scenario and compared them with results obtained in the city scenario.

We first plot the distribution of informed vehicles to get a visual intuition of the performance of our protocol in the three scenarios, as depicted in Figure 6.15. Looking at the Figure 6.15(a), 6.15(e), 6.15(i), and 6.15(m), the different topolo-



(a) CITY Map (b) Subscribers (c) Broadcasts (d) No broadcasts

(e) URBAN Map (f) Subscribers (g) Broadcasts (h) No broadcasts

(i) RURAL Map (j) Subscribers (k) Broadcasts (l) No broadcasts

(m) HIGHWAY Map (n) Subscribers (o) Broadcasts (p) No broadcasts

Figure 6.15: City (a-d). urban (e-h), rural (i-l), and highway (m-p) scenarios. First column illustrates the Map, POI, replicas (black dots), and persistence zone (circle). Second contains road segments with high percentage of subscribers. Third depicts the broadcast distribution while the fourth demonstrate road segments with no broadcasts.

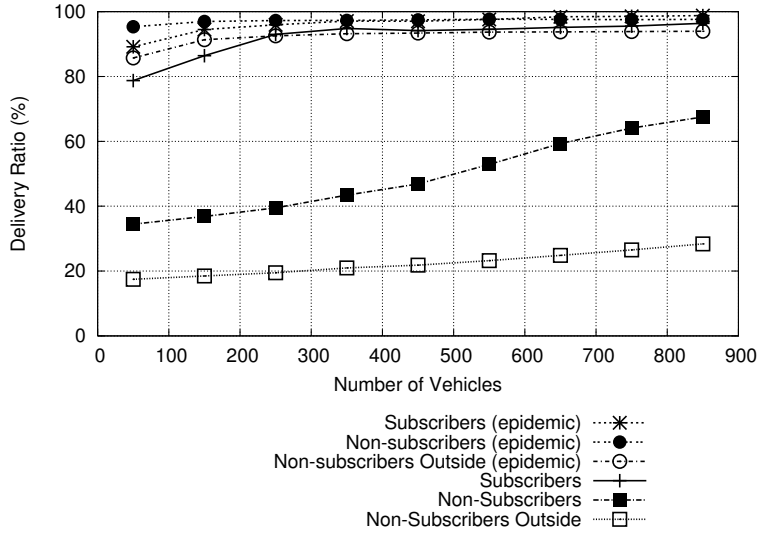
gies of the four scenarios emerge. In the city scenario, many more roads and potential routes are present while in the latter three, the topology is simpler. Figure 6.15(b), 6.15(f), 6.15(j), and 6.15(n) depict the distribution of subscribers across the whole simulation area. Note that these include all nodes travelling towards the POI depicted in the leftmost charts. Due to the more complex topology, in the city scenario, only nodes close to the POI are actually subscribers while in the other scenarios, since there are fewer roads, all nodes travelling on the main road are subscribers, i.e., all nodes are going towards the POI.

Regardless of the underlying topology, the main contribution from the delivery, as already outlined, comes from the replicas in the persistence area. Indeed, the distribution of broadcasts (see Figure 6.15(c), 6.15(g), 6.15(k), and 6.15(o)) is higher in the persistence area than in the rest of the chart, as plotted in Figure 6.15(d), 6.15(h), 6.15(l), and 6.15(p).

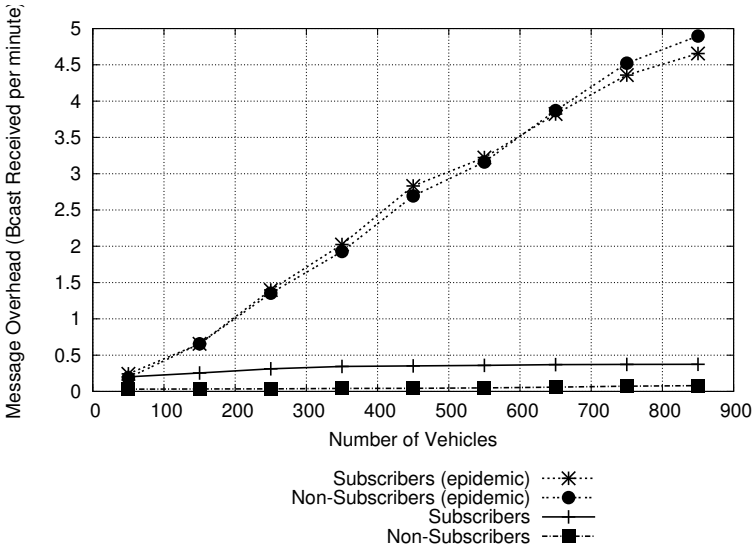
Note that message propagation extends also beyond the persistence area but almost only subscribers are reached by the message. This behaviour is due to the opportunistic dissemination which keeps on informing new subscribers, exploiting vehicles which overheard the message in the persistence area. In this way, subscribers are informed, at virtually no cost, much earlier than the time they would enter the persistence area, thus enabling them to take the proper actions, e.g., in case of a traffic congestion or emergency, in advance. This is even more evident in the highway and rural scenario because, given the scarcity of roads, subscribers leaving the persistence area are much more likely to travel on the same road, but in the opposite direction, to a subscriber going towards that area, thus increasing the probability of opportunistically exchanging messages.

Finally, if vehicle density is low, e.g., in the rural scenario, replicas can leave the persistence area because the current carrier might not find any suitable vehicle to forward the replica and, hence, the replica is kept until a better carrier is encountered. This explains why in Figure 6.15(k) we have some broadcasts in areas where there are no subscribers.

Epidemic Dissemination: To get further insights on the efficiency of the protocol, we compared it against an epidemic version, inspired by [Vahdat 00]. In this protocol,

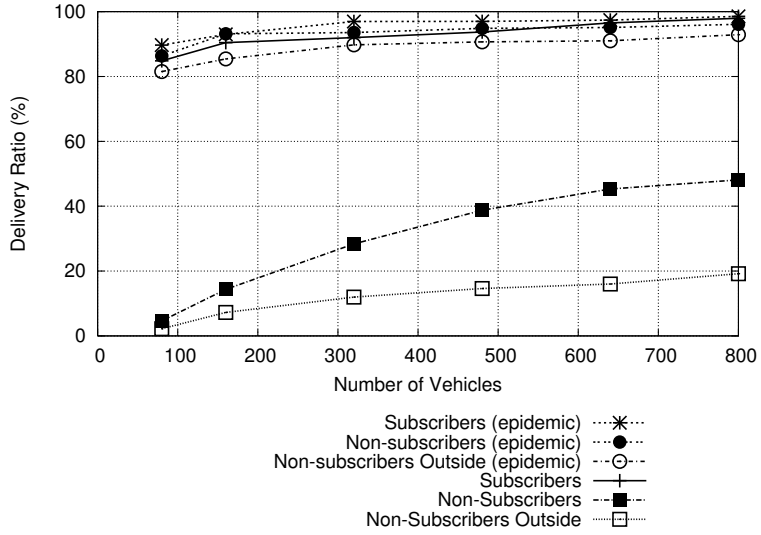


(a) Delivery inside and outside the persistence area.

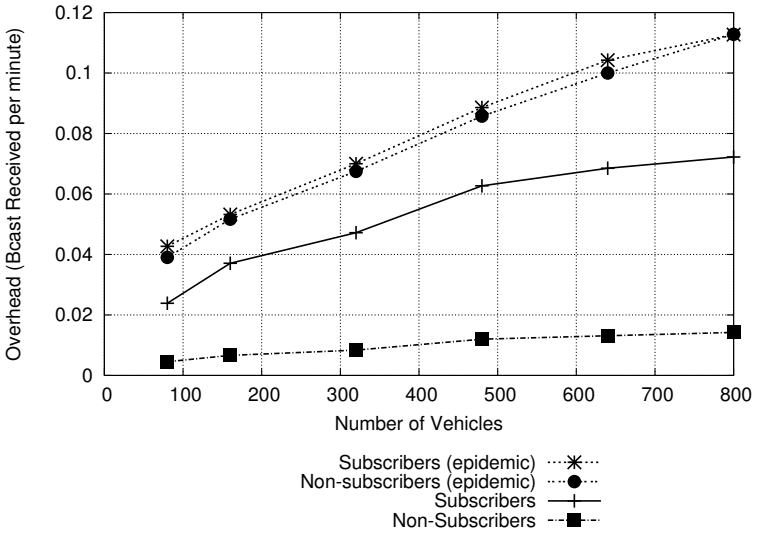


(b) Overhead

Figure 6.16: Density of vehicles (City)



(a) Delivery inside and outside the persistence area.



(b) Overhead.

Figure 6.17: Density of vehicles (Highway)

all nodes gossip to all neighbours which have not previously received the message. In this way, the epidemic infection is kept alive and eventually all vehicles get informed. This protocol can be seen as an extension of our opportunistic dissemination in which all vehicles, not just subscribers and vehicles which overheard it, receive the message. We have already shown in Figures 6.11 and 6.13 that opportunistic dissemination is not sufficient unless coupled with persistence (either infrastructure-based or ad-hoc). Here we make a further step in this direction and show that epidemics provide good performance in terms of delivery but the overhead is an order of magnitude higher than ours. This is observable in Figure 6.16(a) and 6.16(b): although delivery is quite high, the overhead increases enormously. Furthermore, while the overhead of the protocol increases sub-linearly with the density of vehicles, the epidemic overhead increases linearly. The difference is due to the selectivity of the protocol which delivers messages only to proper subscribers and hence is less impacted by the density of vehicles. On the other hand, the epidemic protocol infects all vehicles, not just subscribers, as illustrated by the much higher delivery ratio of non-subscribers. This becomes even more critical if we extend our analysis of non-subscribers outside the persistence area. Indeed, while outside the persistence area, the protocol affects around the 20% of vehicles that are non-subscribers, the epidemic protocol has to contact *all* vehicles, which is unacceptable in real situations.

The same trends are observed in the highway scenario in Figure 6.17(a), although here most nodes are subscribers and hence the fraction of non-subscribers informed is much lower with our approach. The overhead in Figure 6.17(b) follows a behaviour akin to the one observed in the city scenario, although the absolute values are lower. On the highway the set of neighbours changes less frequently and, hence, broadcasts are less triggered. Similar tradeoffs also emerged in the urban and rural scenarios (not shown for space reasons).

These results further confirm that the protocol deals effectively with the characteristics of hybrid vehicular networks, ensuring high event delivery ratios with reasonable overhead in a heterogeneous set of realistic scenarios.

Delivery against time: Lastly, we analysed how the delivery ratio evolves in time, to understand how rapidly the information is disseminated. In this scenario, we published a notification 1km from the dissemination area. Afterwards, we measured the delivery

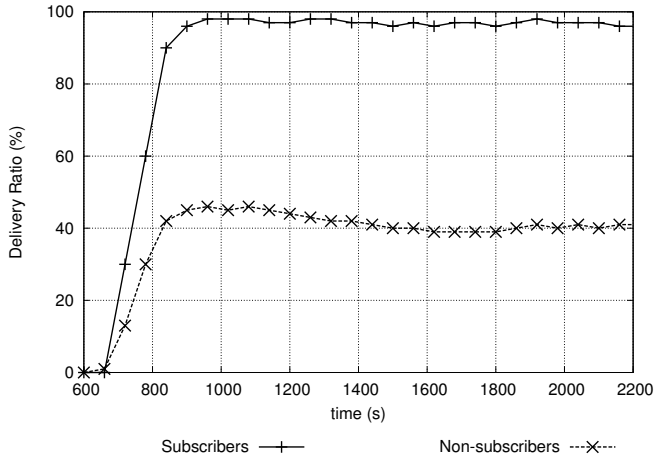
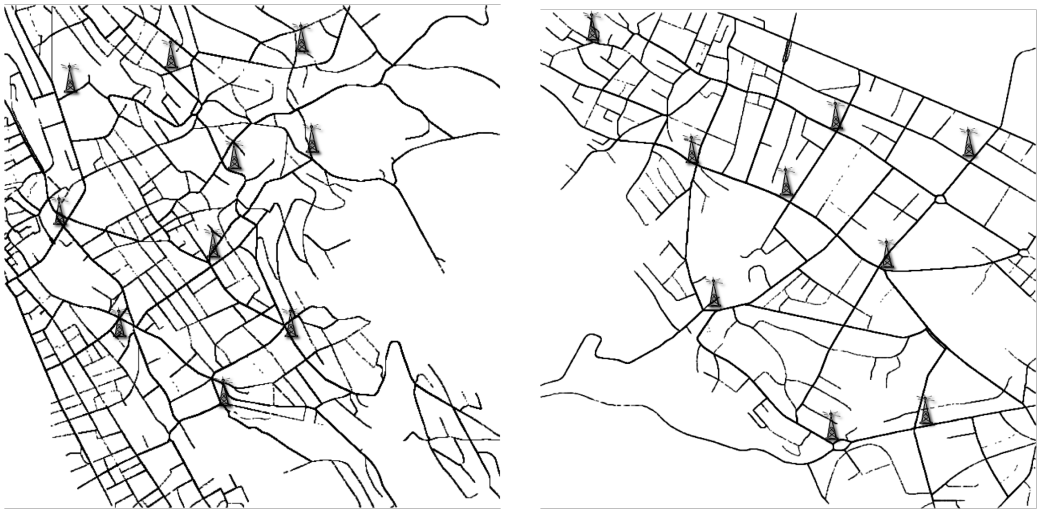


Figure 6.18: Delivery against time.



(a) City. (b) Urban.

Figure 6.19: Simulation scenarios.

ratio in different snapshots of the network: at each snapshot, we measure the number of vehicles inside the dissemination area that have received the message. In Figure 6.18 we can observe that the delivery ratio remains very low for the initial 40 seconds and then experiences a sharp rise. This behaviour is attributed to the time required for the message (published at $t = 600s$), to be routed inside the disseminated area. However, after this transitory start, the dissemination expands very fast and the delivery ratio remains near 100% throughout the publication time (30 min).

6.4 Simulation Results: Pull-Based Dissemination

To evaluate our pull-based approach, we included a number of infostations (1 to 9) located at major road intersections. You can see examples of such configurations in Figure 6.19. We experimented with two ways of placing the infostations: i) randomly chosen and ii) fixed at major intersections. Here we present the results for placing the infostations in the 10 most busy intersections of the map that do not overleap: We rank the intersections, based on how many vehicles drive through it during the simulation. Afterwards, we place the first infostation at the first intersection, the second infostation at the next intersection that doesn't overlap with the radius of the first one, etc. We selected this placement strategy as it maximises the coverage of the population of vehicles.

In these scenarios, a vehicle periodically issues a query to retrieve data from the nearest infostation. This query is routed opportunistically towards the selected infostation. When the reply is available, it is routed back to the vehicle following the expected route, piggybacked to the query (Section 4.3). To properly account for computation overhead and network delays, we model the interval between the time a query is received by the infostation and the time a reply is generated as a random period between 0.2 and 15 seconds.

In our experiments we analysed the performance of the following strategies to route back replies:

- *Infostations only (No V2V):* No opportunistic inter-vehicular communication is used. The future route of the vehicle is only evaluated to find the first infostation that will be in range of the vehicle. The packet is routed there and waits for the vehicle to collect it. This approach is used as a baseline to evaluate the benefits of the opportunistic strategies described next.
- *Reply Route (RR):* The infostation is selected (Section 4.3.2) and the packet is then routed to the nearest point NP, opportunistically. Once it arrives there, it is kept around the NP (by constantly routing it back to NP) until a vehicle arrives to collect it. This is a simplified version that only marginally exploits the knowledge of the vehicle's expected route.
- *Reply Route and and back-Tracking (RRTrack):* After the reply reaches the nearest

point, instead of waiting for the vehicle, as in the case of *RR*, it starts moving towards the estimated position of the requesting vehicle (on its known route). This strategy fully implements the algorithm that we described in Section 4.3.3 and represents our proposed solution for bulk-content dissemination in vehicular environments.

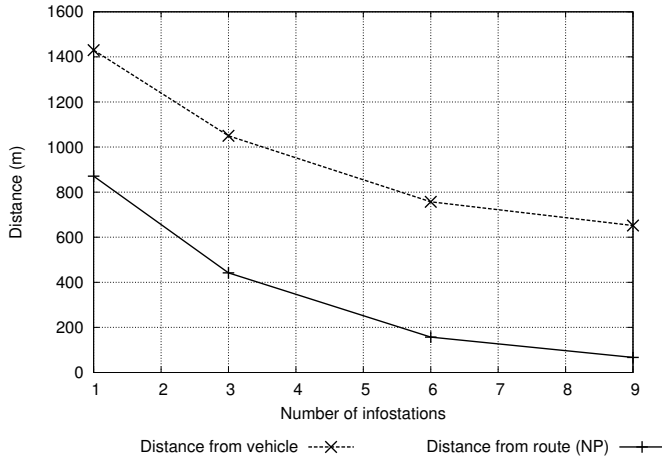
As in our experimental test bed, we considered a 802.11b wireless radio interface, and UDP to broadcast advertisements and TCP to transfer messages. The packet size was also set to the same value (30 KB). The maximum possible communication range is 250 m and broadcasts occur at the same channel frequency. Unless otherwise stated, we use 500 vehicles and 9 infostations. We stopped the simulation after we issued 1000 queries and all the replies have been received or expired.

6.4.1 City Scenario

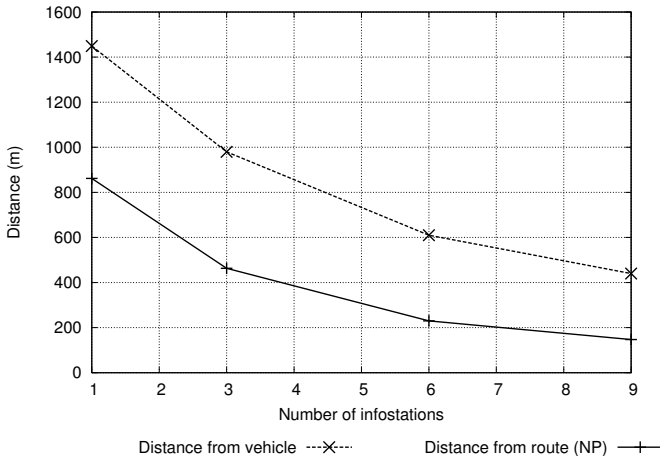
We begin our analysis by discussing the performance of the above strategies in the City scenario. In order to assess the impact of the scenario parameters, we evaluate our approach under different densities of infostations and vehicles.

Number of infostations: In this set of measurements, we varied the number of infostations available in order to examine the density of infrastructure required to achieve a certain delivery ratio or small delay. Increasing the number of infostations (from 1 to 9) affects i) the probability for a vehicle to meet an infostation, ii) the average delay required to meet an infostation, and iii) the average distance between the path of a vehicle and the closest infostation. Results are charted in Figure 6.20.

As expected, the more infostations, the lower the distance to be covered by the message. For example, in Figure 6.20(a), we observe that, when 6 or more infostations are used, a vehicle is likely to drive within 200 meters of an infostation. Furthermore, we observe that when a query/reply was issued the vehicle was on average within 800 meters from the selected infostation (but it will eventually drive within 200 meters). If no-V2V is used, the first infostation that will be *on the path of the vehicle* rather than the infostation that is actually closest to the vehicle is selected. In the example in Figure 4.2, the infostation M would be selected instead of B because the latter, albeit physically closer, is outside the vehicle's range.



(a) Without V2V. The infostation that is closer to the whole path of the vehicle is selected.



(b) With V2V. The infostation that is closer to the current location of the vehicle is selected.

Figure 6.20: Distance between the vehicle path and the selected infostation against the number of infostations (City).

On the other hand, if V2V communication is used, infostations that are closer to the current location of the vehicle are preferred because opportunistic communication can fill the gap between the vehicle and the selected infostation. Therefore, as we observe in Figure 6.20(b), when a message is issued, the distance from the selected infostation is now smaller. But since the selected infostations are not any more on the vehicle's path, the distance from the vehicle's route is slightly higher. For example, when 9 infostations are used, a vehicle should travel 680 meters before it meets an infostation on its path (fig 6.20(a)) but the actual closest possible infostation is only 410 meters away and can be reached through V2V (see Figure 6.20(b)).

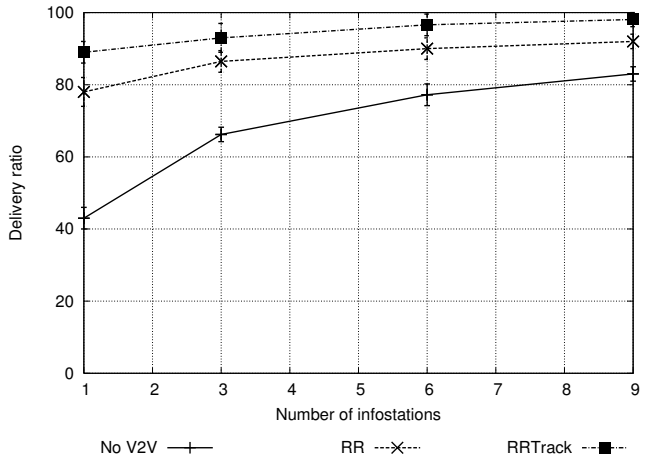


Figure 6.21: Delivery ratio against the number of infostations (City scenario).

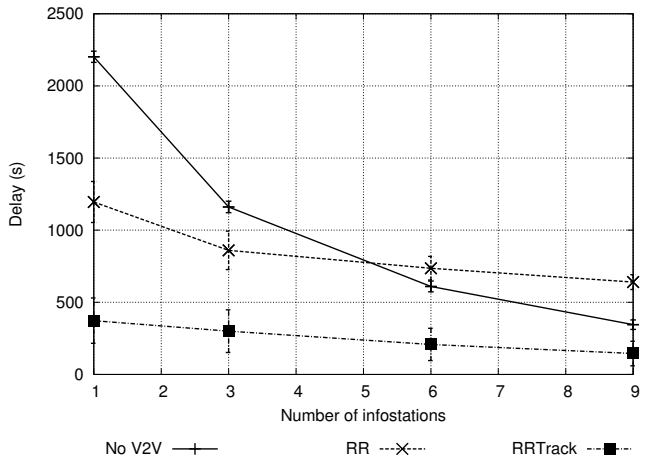


Figure 6.22: Delivery delay against the number of infostations (City scenario).

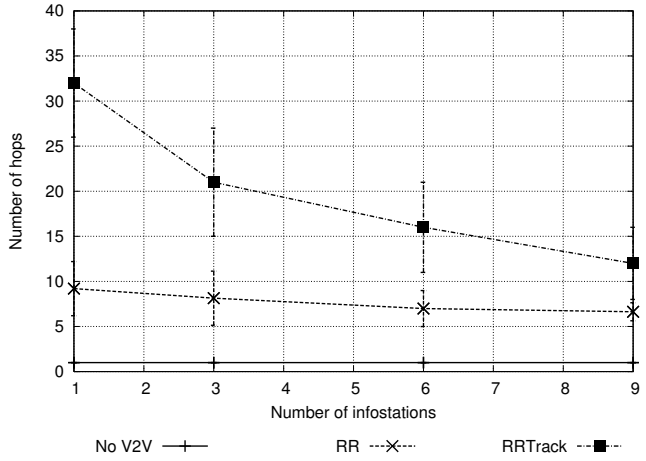


Figure 6.23: Delivery hop count against the number of infostations (City scenario).

In Figures 6.21- 6.23, we plot the delivery ratio (defined as the percentage of correctly delivered messages) and the delay of delivered packets, as well as the average hop count against the number of infostations. Not surprisingly, the delivery ratio increases with the number of infostations. Interestingly, however, if no V2V connectivity is used, the delivery ratio remains quite low (below 80%). On the other hand, opportunistic solutions (*RR* and *RRTrack*) virtually extend the range of the infostations enabling correct delivery, even when vehicles are far from the infostations. Indeed, with just a single infostation in the whole area, 89% of the packets are delivered. Additionally, when backtracking (i.e., the *RRTrack* line) is used, the delivery ratio is even higher because packets move towards the destination along the route (i.e., less packets get lost or expire).

Similar trends can also be observed for the delay plotted in Figure 6.22. If we rely on infostation-only communication, the number of infostations has a huge impact on packet latency. Indeed, since only infostations in the proximity of a vehicle can be used to collect replies, it may take long before a vehicle encounters one on its route. Conversely, when V2V communication is used, the delay rapidly drops since the reply travels backwards on the route of the destination hopping from car to car. If we combine these results with the ones about the delivery, it turns out that the *RRTtrack* solution yields very good performance both in terms of delivery (always above 90%) and delay (always below 400s), regardless of the number of infostations deployed. This is a significant result because it shows that our approach is indeed successful even if very few infostations are deployed.

Interestingly, the delays of all the three approaches are inversely proportional to their average hop count (Figure 6.23). This is a consequence of the fact that messages travel much faster than vehicles and, hence, it is generally better to forward a message to another vehicle rather than waiting for it to pass-by this infostation. This is particularly evident if we compare the performance of the *No V2V* solution against the one of *RRTrack* when only one infostation is available. *No V2V* requires just one hop transmission (i.e., the broadcast between the vehicle and the infostation) but exhibits an average delay of more than 2,000sec (33 minutes), due to the time required for the vehicle to reach the infostation. *RRTrack*, instead, is able to deliver almost all messages (as opposed to only 40% for *No V2V*) in less than 500sec (8 minutes), although on average each message has to travel across 32 vehicles.

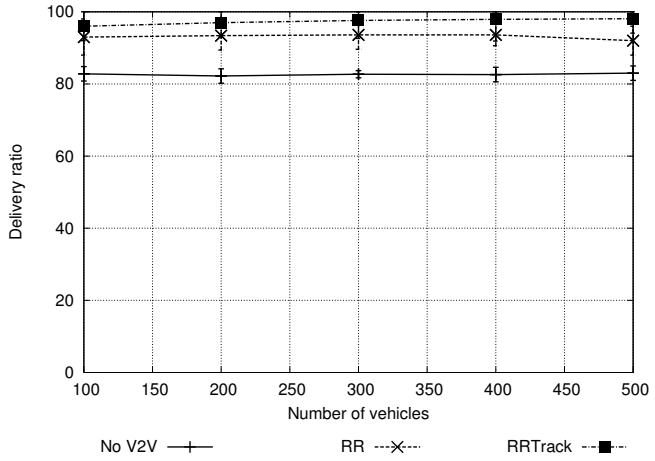


Figure 6.24: Delivery ratio against density (City scenario).

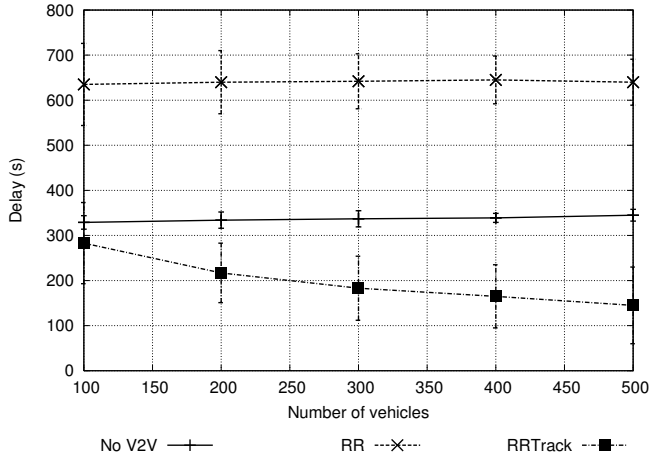


Figure 6.25: Delivery delay against density (City scenario).

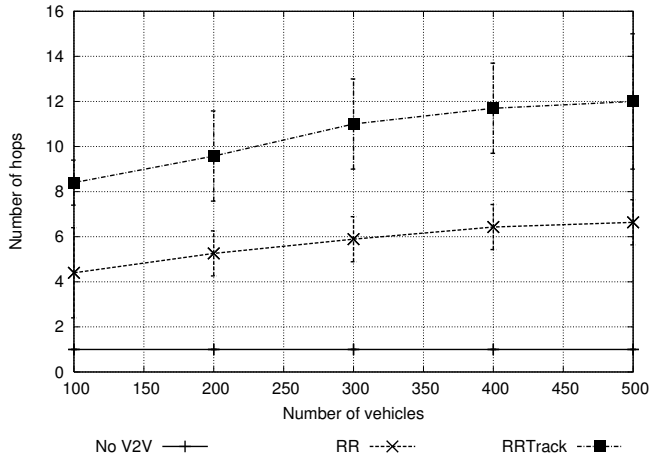


Figure 6.26: Delivery hop count against density (City scenario).

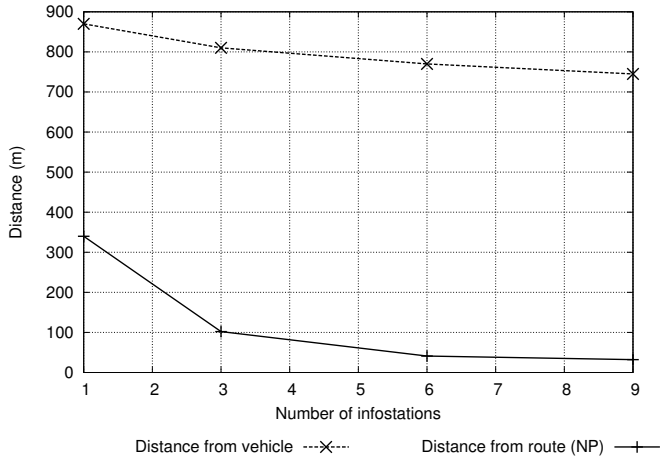
Density of vehicles: We also investigated the effect of vehicle density by varying the number of vehicles that participate in every simulation run. Clearly, this has no impact on the infostation-only approach as the latter depends only on the mobility patterns and the placement of the infostations. V2V communication's performance, instead, benefits from higher density as more neighbours' choices are available for our routing protocols. In particular, high densities enable the *RRTrack* solution to both slightly improve the delivery (Figure 6.24) and significantly reduce the delay (Figure 6.25) because more carriers are available. In fact, the hop count (Fig. 6.26) increases linearly with the density because more message hand-overs occur. Notice also that the hop count of *RRTrack* is higher compared to *RR*. This happens for two reasons: first of all in *RRTrack* the message, in most cases, quickly hops backwards multiple times to reach its destination in a manner similar to greedy. Furthermore, in *RR* the message is only routed back to the the NP when it escapes the route or when it is moving away from the destination (e.g., no unnecessary transmissions are made).

6.4.2 Urban Scenario

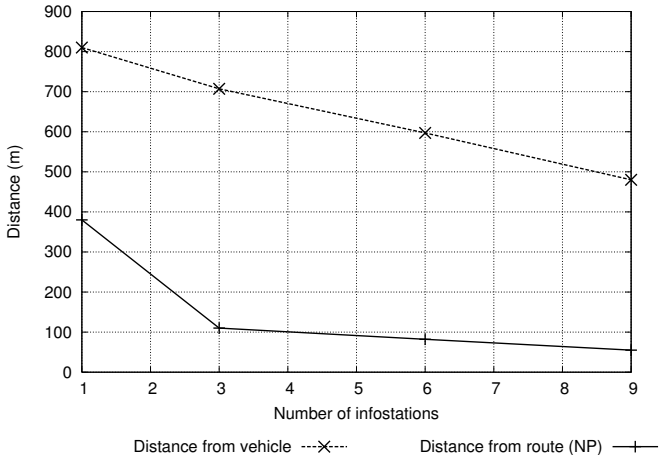
We repeated the same set of experiments in the urban scenario to show the behaviour of our protocol in larger areas and with faster mobility. The results generally mirror the ones of the city scenario with some differences due to the higher speed of vehicles and the lower number of routes.

A first difference concerns the average distance from an infostation reported in Figure 6.27. Indeed, regardless of the approach adopted, vehicles are usually further from an infostation than in the City scenario because the area is much larger and fewer routes are available. Conversely, the distance from the route NP is much lower because, since there are fewer roads, the probability of a vehicle coming across an infostation along the path is much higher.

The delivery ratio chart in Figure 6.28 confirms the results for the city scenario: the infrastructure-only approach improves the case when more infostations are present but it never goes above 90%. This value is higher than the one achieved in the Urban scenario (around 80%) as a consequence of the fact that fewer routes are available and, hence, there are more chances to find an infostation along the path. This also impacts the delivery with and without backtracking, which are now very similar. Indeed, by



(a) Without V2V.



(b) With V2V.

Figure 6.27: Distance between the vehicle path and the selected infostation against the number of infostations (Urban).

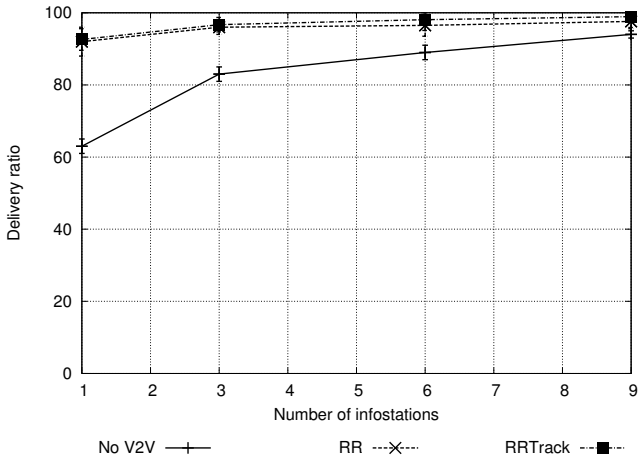


Figure 6.28: Delivery ratio against the number of infostations (Urban scenario).

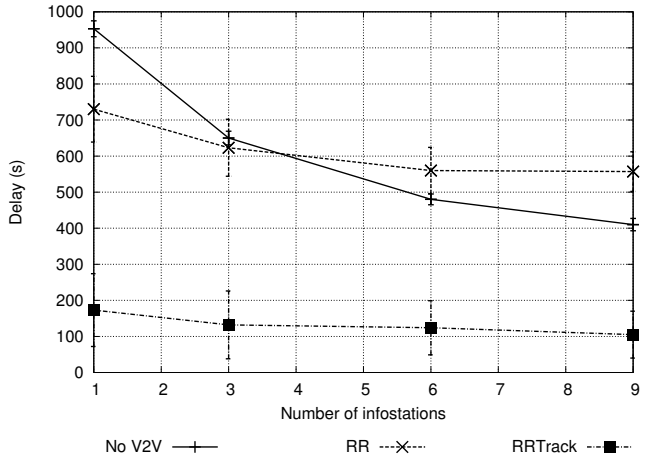


Figure 6.29: Delivery delay against the number of infostations (Urban scenario).

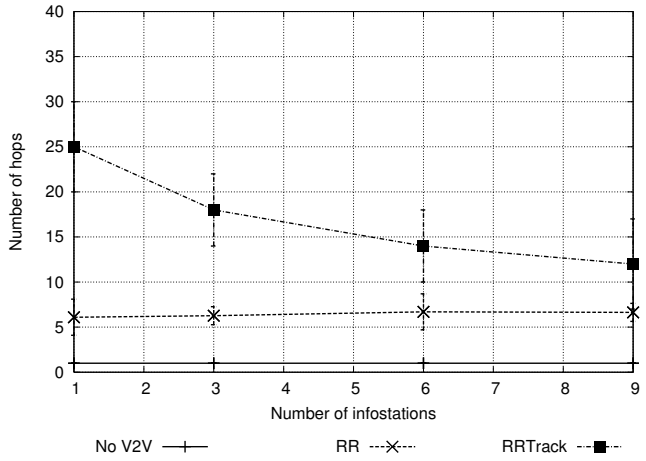


Figure 6.30: Delivery hop count against the number of infostations (Urban scenario).

having fewer routes, the impact of backtracking becomes less evident because a vehicle will reach the NP soon anyway.

Nevertheless, backtracking is still useful to reduce the delay. Indeed, as shown in Figure 6.29, RRTrack largely reduces the delay compared to the other two approaches. Also, while the relative trends are similar to the ones in Figure 6.25, the absolute values are lower due to higher speeds of vehicles in the Urban scenario. Furthermore, in RR, the delay is sometimes higher than not using V2V due to the fact that the opportunistic approach is able to further deliver messages that wouldn't have been delivered, but require higher delay.

Finally, the number of hops in Figure 6.30 closely resembles those for the City scenario (Figure 6.23). This happens because the number of hops depends mainly on

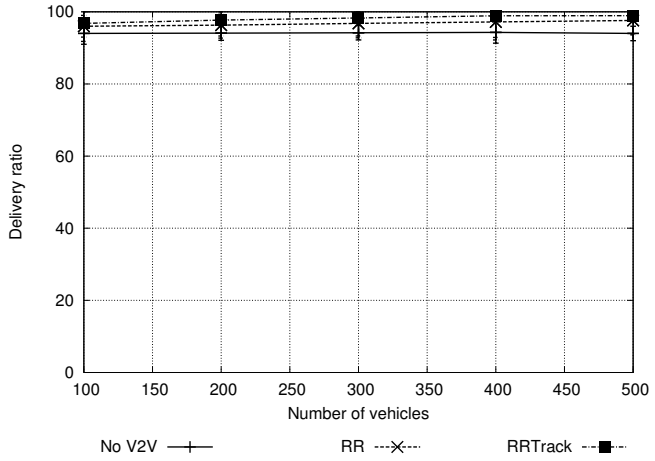


Figure 6.31: Delivery ratio against density (Urban scenario).

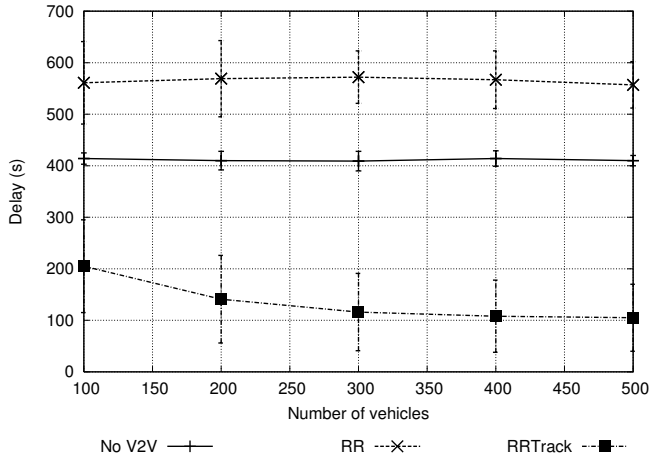


Figure 6.32: Delivery delay against density (Urban scenario).

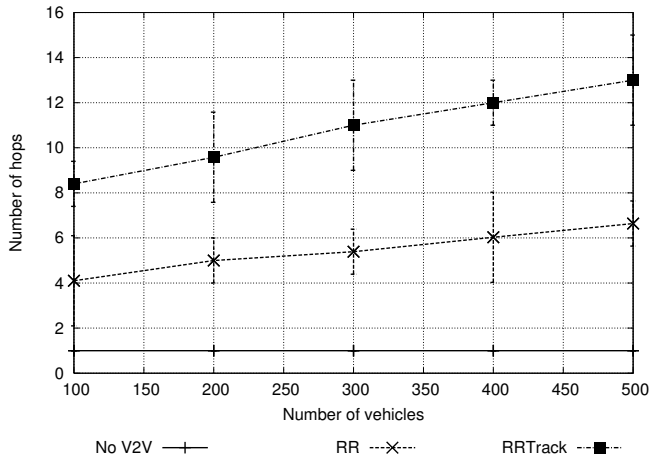
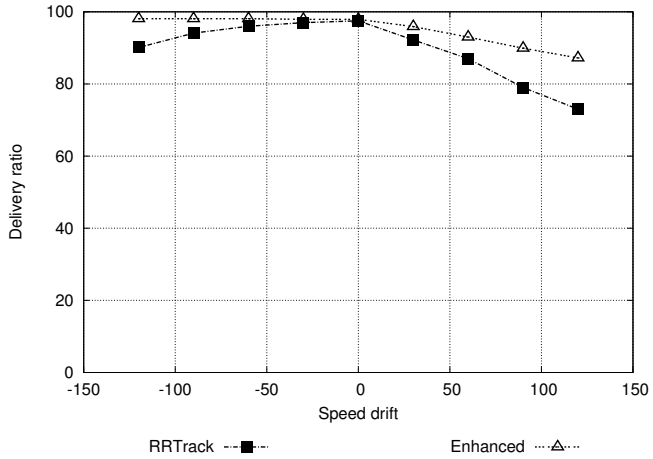
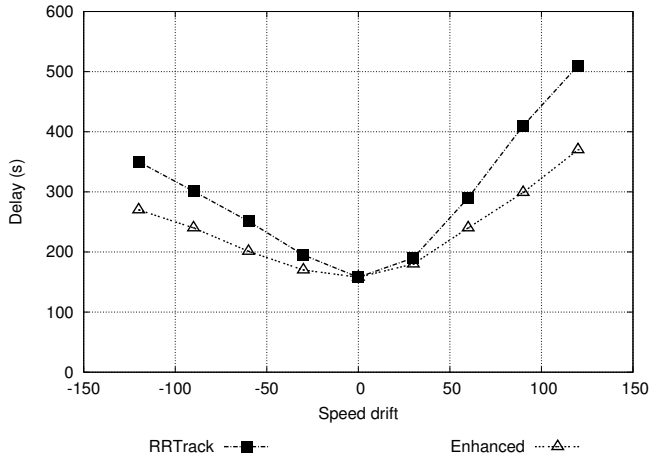


Figure 6.33: Delivery hop count against density (Urban scenario).

the distance from the infostations and the density of vehicles.



(a) Delivery Ratio.



(b) Delay.

Figure 6.34: Wrong speed estimation (City). Negative values = vehicle is late. Positive values = vehicle is earlier than reported. Zero = vehicle is moving according to plan.

For the same reason, varying the density of vehicles provides approximately the same results as obtained in the City scenario. Indeed, since the relative speed does not change, both delivery and number of hops are unaffected. Only the delay shows some variations because of the higher speeds used.

6.4.3 Impact of Position Estimation Error

In the last experiment, we investigated the impact of wrong time estimations due to vehicles moving faster or slower. To this end, we deliberately added errors to the arrival time reported by vehicles' navigator systems and measured the delivery ratio and delay obtained by *RRTrack* and by its enhanced version introduced in Section 4.3.4.

As depicted in Figure 6.34(a), if a vehicle arrives later than the time predicted, i.e., it drove slower than expected (negative values in the chart), the delivery ratio of *RRTrack* slightly drops. This occurs because when the reply arrives at the point where it was expecting to meet the requesting vehicle, it will start going forward (instead of backwards), trying to chase the estimated location of the vehicle. However, since the destination is moving more slowly than expected, it will still be behind. This results in having the reply ahead of the target vehicle (the requesting vehicle is now chasing the reply), thus explaining the delivery drop.

Similarly, if a vehicle is moving faster (positive values in the chart) and arrives at the NP before the reply, the impact on the delivery is even worse because i) the selected infostation is wrong (meaning that the packet does not have enough time to get to NP before the vehicle) and ii) the packet is routed towards the estimation that is behind the actual vehicle. However, if we apply the enhanced technique described in Section 4.3.4, consisting of splitting the reply into two different packets, one moving forward and the other moving backward, the delivery ratio significantly improves. In particular, the delivery ratio for slower vehicles becomes as high as when vehicles are on schedule. Indeed, since one copy of the reply is moving backward, it will eventually meet the delayed vehicles and deliver the message. Similarly, also the delivery ratio for faster-moving vehicles increases (about 15% for vehicles that are 2 minutes ahead of schedule). In this case, the gain stems from the fact that since the copy moving forward adopts a more aggressive "greedy" routing (i.e., the message is always forwarded if a neighbour ahead is found), there are more chances to catch up with the vehicle. Nevertheless, some replies may still get lost because if the vehicle arrived too early it may have moved too far away before the reply arrives and the reply may expire before reaching the vehicle. Analogously, also the delay (see Figure 6.34(b)) largely benefits from the enhanced approach. Indeed, by moving the reply backward as well as forward, the time to meet a slower vehicle lying behind is reduced. On the other hand, thanks to the more aggressive "greedy" forwarding, even faster vehicles can be approached earlier.

These results show that even in the presence of wrong estimation our solution is able to ensure high delivery, especially in the case of delayed vehicles, which we expect to be more likely, especially in a city environment where many events (e.g., traffic-lights and road congestion) can decrease cruise speed.

6.5 Conclusions

In this chapter we evaluated all the aspects of our framework: 1) the geographic routing protocol (Chapter 2), 2) the push-based dissemination (Chapter 3) and 3) the pull-based dissemination (Chapter 4). The simulation results indicate that our framework can provide good quality of service without inducing significant overhead under a number of different conditions dictated by the mobility scenarios used.

7

Evaluation of the Impact of our Framework

In this chapter we evaluate the impact of our dissemination framework on vehicular mobility dynamics. More specifically, we consider *every vehicle as a sensor that collects and disseminates road traffic information.* Consequently, the *vehicles' mobility patterns will be affected by the collected information* (i.e., we would like to allow the navigation system to re-calculate a route based on the collected traffic information so as to avoid congested areas and reduce trip times).

While in the previous chapters we have concentrated on how to disseminate information, in this chapter we aim to prove that the use of our dissemination system can benefit the drivers. Therefore, here we do not evaluate performance in terms of delivery ratio, hop count, etc., but we focus on how the vehicles can self organise so as to reduce their overall trip times when such an ad-hoc dissemination mechanism is employed.

To disseminate traffic information, both our push and pull based protocols could be used, however, since traffic information concerns multiple vehicles, the push based

approach (presented in Chapter 3) is more suitable. Therefore, in our scenario vehicles periodically publish their collected traffic information so that interested vehicles (i.e., vehicles planning to drive through these locations) can act accordingly.

To analyse an environment where dynamic mobility decisions are allowed, we need to design an evaluation system that consists of two independent components that constantly interact: 1) A *network simulator* that implements the dissemination system so as to share and correlate traffic information and 2) a *dynamic mobility generator* that plays the role of a dynamic navigation system and emulates the mobility of the vehicles using the provided map and the estimated traffic conditions. These two simulators constantly interact to simulate scenarios where mobility decisions are affected by the disseminated information and, conversely, where the network dissemination is affected by the mobility decisions (as the mobility patterns affect routing/dissemination protocols).

The second part of this system (the mobility trace generator) has been provided by our collaboration with the Network Research Lab (NRL) of the University of California Los Angeles (UCLA). More specifically, *Mobidense* [G. Marfia 07] a microscopic vehicular mobility simulator has been developed in UCLA. As we will examine in detail later, Mobidense produces static traces based on real maps, traffic light databases, driver behaviour, road capacities, etc. Our aim has been to collaborate with this group so as to make such a mobility generator work together with a network simulator in order to enhance the mobility decisions with the information collected by an ad-hoc dissemination system.

The contribution of this chapter is to provide some support to the argument that content-based information dissemination can help drivers and, thus, justify the use of such a framework in order to build vehicular applications.

7.1 Motivation

Every day millions of vehicles flow from residential areas to business areas in the morning and back in the evening. Various traffic measurement systems are deployed to support smart vehicular routing around accidents or heavily congested areas. These are limited in a number of ways and they provide limited and coarse grained traffic information

for a very small subset of roads. Traffic control decisions are taken by observing traffic flows and are enforced with traffic light synchronisation systems and dynamic message signs.

As we examined in the previous chapters of this thesis, an ad-hoc dissemination system can quickly spread traffic information to all the interested vehicles. In fact, this information can be measured by the vehicles themselves: vehicles may act as sensors that record and afterwards share the *traversal time* for each road segment along their route. Later, based on the collected information (through the network) and the navigation system's knowledge of the area, the vehicle can update its navigation route in order to avoid congested areas. As this system is fully decentralised, re-routing decisions are taken individually by each vehicle based on its individually collected knowledge about traffic conditions on its route.

However, it is not always clear if such an ad-hoc dissemination framework can help the drivers: the question that we will try to investigate is whether this decentralised approach can help the drivers to make correct decisions, that minimise the *global* traffic congestion, or if this instead causes more problems (e.g., traffic fluctuations, traffic jams in previously unproblematic areas).

The main performance metric that will be used is the average trip time. In particular, we will see how the overall average trip time evolves with and without the use of a VANET. We also compare the dissemination protocol to existing state of the art: information derived from induction loops and cameras.

Induction loops are placed in the asphalt and provide instantaneous measurements for speed and traffic flow for each location. This metric suffers from a number of problems that limit its reliability. Intuitively, induction loops record speed information at certain locations on a street, thus their results may be misleading in urban stop-and-go traffic conditions. Video cameras are slowly replacing induction loops, but their widespread deployment is limited by their cost. The advantage in using video cameras is of recording end-to-end times rather than instantaneous speed samples and, therefore, provide more valuable statistical information. In comparison, in our model each vehicle is able to measure end-to-end trip times for any road segment.

7.2 Application Overview

We designed an application (to measure, share and interpret (correlate) incoming traffic information. In this section we present its three key components and the issues that we had to deal with in each one of them:

1. **Traffic Sensing**. It is important that the vehicle can accurately sense traffic conditions while it travels, as this information may affect the driving decisions of numerous other vehicles when these observations are shared. To implement this module, traffic metrics should be defined (e.g., speed, traffic volume, traffic density and trip time).

2. **Traffic Information Dissemination**. The observed information needs to be shared with other vehicles. This is handled by our dissemination framework presented in Chapter 3.

3. **Traffic Estimation**. When vehicles receive information through the network they need to update their maps so as to represent the estimated traffic conditions (based on their collected information). Questions about how traffic information should be interpreted need to be answered. This interpretation has a fundamental impact on the performance of the system.

It is not possible to satisfy any of the previous requirements without being able to evaluate the consequences of each design choice on performance. The complexity and the scale of the real system makes "on field" evaluation prohibitive. There is a need for tools capable of producing a realistic traffic emulation and handling dynamic routing, i.e., the assessment of collected data and the correction of the mobility of the vehicles at run time. This is the most important piece of the puzzle. Informed navigation does not necessarily lead to the optimum (i.e. minimum travel time) solution, and that can worsen performance ([Jayakrishn. 90]). It is then important to have tools that resemble reality as much as possible and that give a deep understanding of the consequences of each design decision. Therefore, along with the application, we also designed an evaluation platform that is able to dynamically re-route vehicles based on the collected information.

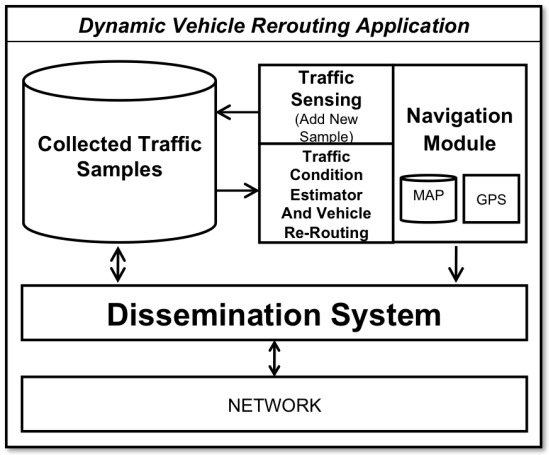


Figure 7.1: *Our Application's architecture.*

7.3 Application Description

The application that we implemented around our dissemination system is composed of various modules as we can see in Figure 7.1. We now describe the components in detail.

7.3.1 Traffic Sensing Module

Every vehicle collects traffic information. Many different metrics may be collected (e.g., coarse/fine grained samples, speed samples, density estimations, average speeds). We select a simple, yet efficient way which may be found in many other works in this field: we model the street topology as a directed graph, where each link connects two intersections, and measures the time required to traverse each link. This choice is consistent with the majority of map databases [U.S.C.B. 09] and navigation systems. Therefore, it is easy for existing navigation systems to collect information about traversed road segments and the same information can be used by other vehicles to estimate traffic conditions. Note that in this model a two-way street is modelled with two directed links, each direction having its own link ID.

We choose the delay incurred in driving through a road segment as an estimate of the traffic conditions. This is the information drivers are most interested in. By correlating the samples that are generated by different vehicles, it is possible to compute and track, in real time, the trip time for each vehicle.

In our implementation every time a vehicle exits a link it creates a sample of type

{`linkID delay timeStamp carID`}. The `linkID` should be unique per street segment and direction throughout the vehicular network. The delay is measured as the time spent by the vehicle on the link with the given `linkID`. The timestamp is the GPS time. The `carID`, inserted to ensure that samples may be uniquely identified, may introduce privacy concerns. For this reason we can instead use a random number. Choosing an integer random number in $[0, N]$, where N is much bigger than the number of vehicles in the considered area, the only risk is to have multiple vehicles with the same `carID`, `linkID` and `timeStamp`. The risk of this happening is clearly very little. When leaving an area a vehicle would then generate a new `carID` compatible with the new area it is travelling in.

7.3.2 Dissemination Module

Our aim is to use our dissemination techniques to publish the collected traffic information. The primary objective of the module is to disseminate the collected information throughout the vehicular network. It should: 1) Collect as much information as possible for each link on the map, 2) Propagate the most recent information and 3) Limit communication overhead.

For the dissemination we used the Push-based approach presented in Chapter 3, as the disseminated information is useful to a number of drivers travelling towards the sample's area. However, pull-based approaches could be also used.

A key element of the dissemination module is the sample selection algorithm (i.e., which part of the information that a vehicle keeps should be published to the neighbours, assuming that we can only transfer a fraction of a vehicle's knowledge). Therefore, when a packet is built, it needs to include only the tuples {`linkID delay timeStamp carID`} which provide the highest map coverage and sample freshness. Even though a limited amount of information can be propagated per link, accuracy of information should be preserved. In our implementation we use a simple utility metric that maximises both coverage and information freshness. We select, in round robin fashion on links, the newest sample that was not already selected in the previous round, until the published message is full.

7.3.3 Traffic Estimation and Vehicle Re-Routing Module

Each vehicle receives messages from the dissemination module in the form {`linkID delay timeStamp carID`}. This information is then used by each vehicle to compute the shortest path (in terms of time required) to destination. Such choice has been shown [Beckmann M. 56] to be non-optimal in minimising the total trip time. In order to minimise aggregated trip time, each vehicle should follow the path that minimises the sum of delays and the delay increments generated by its choice of following such a path. The estimation of this second quantity is an open research topic outside the scope of this work. We therefore implement a routing algorithm that chooses the shortest delay path to the destination. This is performed on a local (selfish) basis and does not necessarily lead to average travel times that are lower than navigating with no information.

To select the route of the vehicle we use a modified version of the Dijkstra algorithm [Sedgewick 84] on a weighted graph that represents the map with the current known traffic conditions. Therefore, before running Dijkstra we generate weights per `linkID` based on the collected information. For each street segment *the weight represents the traffic conditions* (i.e., how much time a vehicle would require to travel through each road segment).

The problem of estimating the traffic conditions based on the collected information is not trivial due to various reasons:

- *Noise in observations:* Although traffic conditions do not change rapidly over time (traffic jams usually build up relatively slowly) the measurements can be quite noisy. The main reason for this is that each vehicle might drive through this segment at a different pace: stop or not at a traffic light, pause to pick up a passenger, different driving habits, etc. The result is that the collected samples show some variations even if they are collected within a short period of time. Clearly, some kind of correlation is required to indicate the long term traffic trends.

- *Collection rate variation:* There is no guarantee that the traffic samples will be collected at a standard rate (it depends on the traffic and the dissemination strategy). For example, we might receive numerous samples that are older than 10 minutes and just a few recent (less than a minute): The question is which

samples are more important to estimate the current traffic conditions. In other words, how can the samples, that are taken at various rates and are of different age, be weighted.

- *Absence of information:* Finally, the last problem is how we treat absence of information (i.e., if no recent information is received). One solution is to use weights that represent either historical data (e.g., usual average speed at a road segment) or speed limits. However, questions about how quickly the information becomes obsolete need to be answered.

We evaluated a few simple solutions aimed at interpreting the collected information into current traffic conditions and at evaluating their impact on traffic patterns:

- *Default:* The weights are determined by the length and the speed limit for this segment: $W_{linkID} = \frac{LinkLength}{SpeedLimit}$. This weight is also used in all previous methods when there is no collected information about this link.

- *Most Recent Estimate:* for each link we select the most recent sample (i.e., in terms of creation time): $W_{linkID} = Delay(linkID, mostRecent)$. This approach is subject to fluctuations, given that samples can vary rapidly or be erroneous. As we shall see in Section 7.4, however, this solution is very close in performance to more complex solutions and proves to adapt well to the bursty nature of vehicular traffic.

- *Bayes Estimate:* We use a simple Bayesian estimator to predict the traffic conditions using a large number of samples taken at different time instances:

$$\begin{aligned} W_{linkID} &= (1-w) * Delay_{NewSample} \\ &+ w * CurrentWeight_{LinkID} \end{aligned}$$

  where w is parameter that is calculated based on the age of the sample (so that older samples do not greatly influence the weight).

- *Bayes with Ageing Estimate:* The same as Bayesian but in this case absence of information "ages" the weight back to the default value (given by the free flow traver-

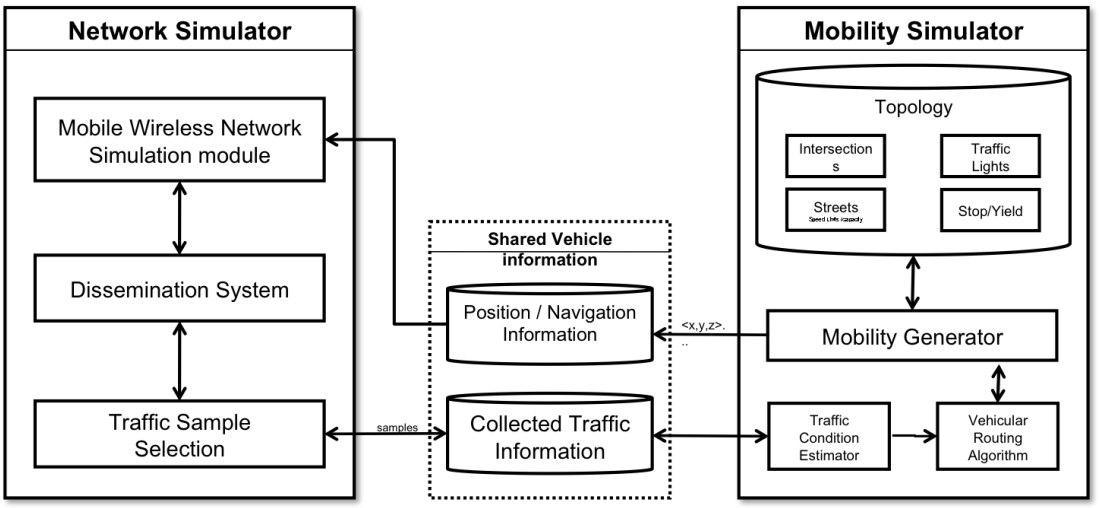


Figure 7.2: *Interactions between the network and mobility simulators.*

sal time): $W_{linkID} = (1 - c) * CurrentWeight_{linkID} + c * DefaultWeight_{LinkID}$. Where c is an ageing factor calculated by:

$$c = \frac{Min(curTime - recentSampleTime, maxAge)}{maxAge} \tag{7.1}$$

7.4 Evaluation

To evaluate such an application, we designed and implemented a tool that couples together a mobility and a network simulator. Similar approaches can be found in [Lochert 05a, Wang 07a, Sommer 08]. Differently from previous work, we integrated QualNet [SNT 09], a communications network simulator specifically designed for wireless networks, and MobiDense [G. Marfia 07], a mobility simulator. These two simulators constantly interact: *future mobility decisions are influenced by the network dissemination (e.g., collected information), and the network dissemination is influenced by the mobility patterns (location of the vehicles).* An illustration of our evaluation system and of the interactions between the simulators is shown in Figure 7.2.

We will now describe the implementation in more detail and, afterwards, we will present the results of our evaluation.

7.4.1 Mobility Simulator

To emulate the vehicles' mobility patterns we used MobiDense [G. Marfia 07] [1]: a mobility simulator developed in UCLA. Mobidense combines topology and traffic flow information to generate a mobility trace. MobiDense requires the following topology inputs:

- Intersection positions;
- Street descriptions, which define the properties of streets segments that connect two intersections (e.g. positions of two endpoints, speed limits, number of lanes, if the street is one-way or not);
- Intersection stop probabilities (e.g., priorities, stop signs) ;
- Traffic light definitions, where traffic light positions, timings and phases are given;

The flow of each street segment is tuned by adapting intersection stop probabilities and red/green time phases of traffic lights. MobiDense models queues at intersections, so that if a vehicle stops and is the first in queue, it waits for one second at the intersection and recomputes whether it should wait more or move on. If a vehicle reaches a queue, it will wait until the queue empties in front of it and it is its turn to pass the intersection. Queues can propagate backwards at ingress streets under heavy traffic conditions so that an approaching vehicle cannot enter a full street segment.

Traffic flows are constructed by providing a source-destination file that defines the origin and end positions for each vehicle and the time at which a vehicle begins its journey. The streets traversed by each vehicle depend on the routing algorithm that is implemented. MobiDense allows interchangeable behaviours for the vehicular re-routing and the traffic data aggregation. In our testing process these behaviours are part of the traffic estimation modules. The traffic estimation modules compute the estimated travel time between the current position and the destination on the available paths, given the received data, and recompute the best route (see Section 7.3).

[1] Note that the implementation of this mobility trace generator is not part of the contribution of this thesis. The following description is added so as to make the comprehension of the evaluation system easier. However, during this research, in collaboration with UCLA, we modified this simulator in order to support dynamic mobility decisions based on the network dissemination.

7.4.2 Network Simulation

We implement the dissemination modules in QualNet, a well known mobility simulator that is particularly suited for wireless networks. Each vehicle selects the information to be propagated based on various possible algorithms (gossip, epidemic, push-based dissemination described in Chapter 3). This information is a subset of the information that has been collected from the vehicle itself (acting as a sensor) plus information received from other vehicles through the wireless network. When new information is available (either by local observation or through the network), it is stored in a shared database, shown in Figure 7.2, as both simulations will need it: the mobility simulator to evaluate the current road-traffic conditions and the network simulator to further disseminate it.

Consequently, QualNet and MobiDense continually interact: Qualnet receives position updates from Mobidense (to update the locations of the hosts), and Mobidense receives traffic samples collected in Qualnet (using our push-based dissemination scheme).

7.4.3 Evaluation Settings

For our evaluation we used a detailed map of Portland, Oregon. The area is approximately 4 x 7 km and includes downtown Portland, (a map is shown in Figure 7.5). This area includes $4,968$ streets, $3,429$ intersections and $16,490$ vehicles throughout the simulation. Start and end points of the journey of each vehicle are based on the traces generated at the Los Alamos National Laboratories, using TRANSIMS. The realism of these traces lies in the fact that they were created by examining the real activity location information. Activity location information, such as information on where residential areas and business areas are, is used to define start and end points of traffic flows at a particular time. These traces represent a typical morning pattern in Portland and this is the behaviour we would certainly expect. We analyse the traces to derive topological information about Portland, such as traffic lights position/delay, intersection stop probabilities, speed limits and road capacities. The MobiDense simulator is tuned to produce, in the absence of information, traces which are as close as possible to the original TRANSIM traces.

7.4.4 Traffic Information Evaluation

The performance results aim at understanding whether an informed navigation system, used by all vehicles, can show an improvement in terms of overall average travel time with respect to an uninformed vehicular network. The main performance measure is the total vehicle's trip time, but there are also other factors that should be considered. It is also important to understand: i) how long the chosen routes are, compared to the shortest routes (in seconds); ii) how information delay affects trip delays; iii) what amount of network traffic such a system produces.

As we described before, each vehicle receives a number of traffic samples and stores them, grouped by `linkID`, in a local buffer. In case no information is known about a link, all the strategies we implement assume that there is no traffic on this link (we assume that vehicles traverse the link at free flow speed).

We here compare the 3 different traffic estimation algorithms described in Section 7.3: i) *Most Recent Estimate*; ii) *Bayes*; iii) *Bayes with Ageing*. We additionally compare these strategies to the case where no information is disseminated, in such case expected free flow travel time is used to compute the shortest route to a vehicle's destination.

Traffic flows are adjusted to 33% of the traffic that is found in the original traces to the actual flow values. To make this clear, if an average of 100 vehicles per hour enter the map from a certain intersection in the original traces, we begin simulating with a flow of 33 vehicles per hour. We simulate the network from very low density scenarios to normal morning traffic scenarios that are directly extracted from the traces.

As we see in Figure 7.3, in the absence of information feedback, when density increases overall trip times quickly rise from 400 seconds, about 7 minutes, to 1200 seconds, about 20 minutes, on average. When our application is used, trip times are reduced. This drop is higher when there is more congestion (the trip time dropped to 13 minutes compared to 20). This result radically differs from what is found in [Jayakrishn. 90, Arnott 89, Al-Deek 98, Kobayashi 99]. An intuitive explanation may be found observing Figure 7.5. As we can see in Figure 7.5.a, traffic is mainly localised on Fwy 5, 405 and on the bridges that traverse the river. But the traffic is not all generated by vehicles that need to traverse the river or that necessarily need to enter a freeway

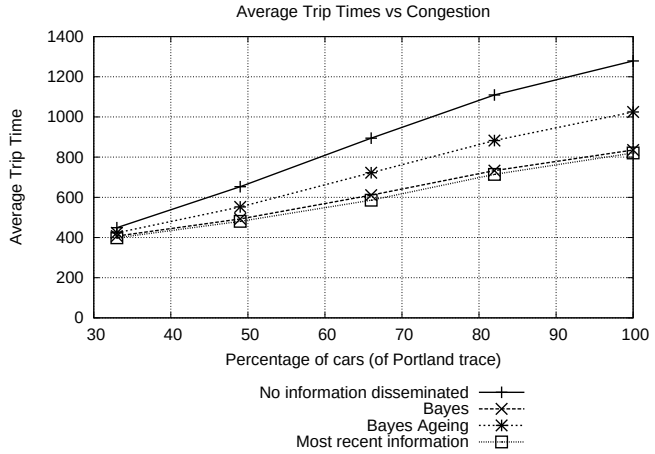


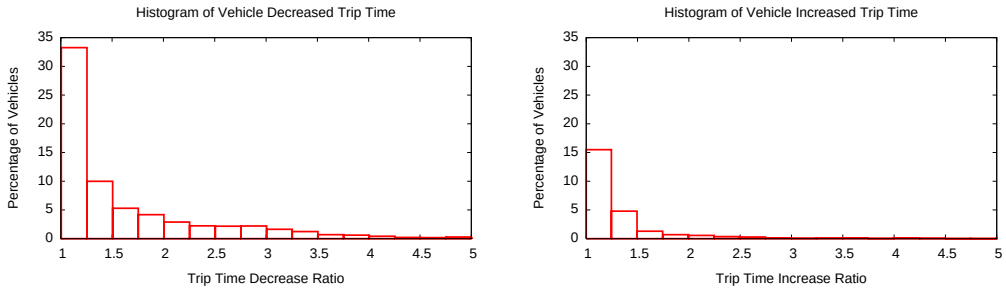
Figure 7.3: Trip times for different information handling strategies.

to reach a destination. Many vehicles could reach their destinations through alternate routes, but do not. In Figure 7.5.b we present the result of using our application, we observe many more yellow (light) links (i.e. slightly congested links) and less red (dark) links (i.e. heavily congested). In fact, visually, a red link in Figure 7.5.a is substituted by a number of yellow links in Figure 7.5.b.

The topology we are here analysing is realistic and more than one path is available to reach one destination from another. Especially in the centre of the map, in the grid-like section, we can see an increase of yellow links when using our dissemination. This leads to much better performance, under normal traffic conditions the average trip time is 29% lower, which is a significant improvement.

In terms of weight calculation algorithms, we observe that using the most recent information and Bayes strategies provide the best results for this simulated environment. This happens because if there is traffic on a link, end-to-end times will explode, while otherwise they will oscillate slightly above the free flow delay time. Bayes with ageing still improves, but is worse than the other two methods since traffic conditions are not changing rapidly.

It is important to understand how travel time improvements are distributed. Figure 7.4 presents a histogram of the trip times gain/loss for normal traffic conditions when dissemination is used. More specifically, gain ratio (Figure 7.4(a)) is defined as $ratio = \frac{oldtime}{newtime}$. For example a ratio of 2 means that the vehicle halved its trip time, a ratio of 3 that it needed one third, etc. Similarly, deterioration time (Figure 7.4(b))



(a) Improvement (less time)

(b) Deterioration (increased time)

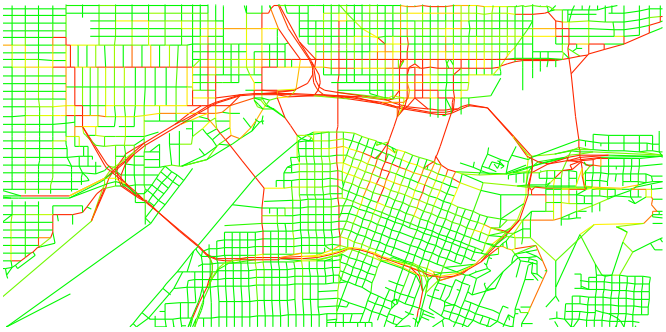
Figure 7.4: Histogram of trip-times loss/gain.

is defined as $ratio = \frac{newtime}{oldtime}$ (a ratio of 2 means the vehicle doubled its trip time). We observe that a large number of vehicles (34%) saved 20% of the time (ratio 1.25 means new time is 1/1.25 of the old time) . There were also luckier vehicles able to avoid big traffic queues and complete their journey two or three times faster (ratio 2 and 3). In total 64% of the vehicles saved time. However, at the same time, we see that some of the drivers required more time when our application was used. This is due to some of the traffic being diverted into smaller roads which, as a result, become busier. However, we can observe that far fewer drivers have their time increased rather than decreased and their trip times are no more than two times longer. Finally 23% of vehicles were not really affected (±10% trip time).

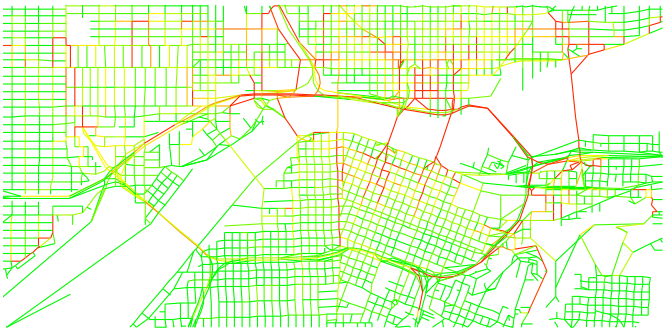
7.4.5 Traffic Information Dissemination Quality

We here want to understand how well we can disseminate traffic information, giving a close representation of real traffic conditions to each vehicle. The results we presented in Figure 7.3 may result from an unfair dissemination of information. We should remember that, in simple scenarios [Jayakrishn. 90, Arnott 89, Al-Deek 98, Kobayashi 99], it has been shown that a fully informed traffic network can deteriorate traffic performance. Random inconsistencies in distributed traffic information may be inducing a better behaviour of traffic.

To estimate the impact of the dissemination protocol on the system's performance, we compute the overall average travel time for the case that all the traffic information is immediately available at all vehicles. This is the infinite bandwidth/zero delay scenario, a full-knowledge scenario where all vehicles know everything in real-time. We then observe the variation in aggregated average traffic trip time, between a fleet of ve-



(a) No information



(b) With Dissemination

Figure 7.5: Map of speed [best viewed in colour]. Green streets are not congested. Orange areas show average speed slightly lower than the speed limit. In red streets segments the average speed is much lower than speed limit.

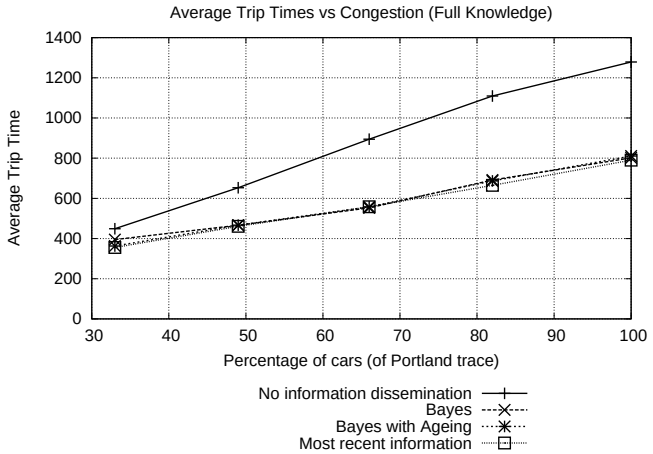


Figure 7.6: Trip times when full-knowledge is available instantly to all the vehicles (best information dissemination case).

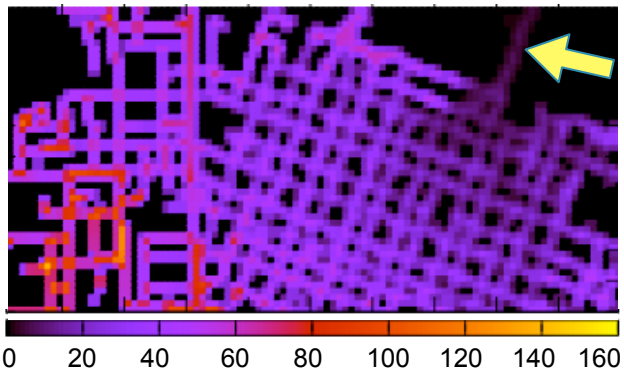


Figure 7.7: 2D Heat-map of age of received information (in seconds) about the link highlighted by the arrow (bridge) [best viewed in color]. Vehicles away from the bridge receive older traffic information.

hicles that disseminate information and a fleet of vehicles that receive all the available information immediately. This mode is obviously only possible in simulation and would not be deployable in reality.

In Figure 7.6 we show the results for the same traffic estimation methods used before, yet now the information is not collected with the dissemination protocol but all the collected information is instantly available when a vehicle re-evaluates the traffic conditions. We observe that the same trends appear as when our application is used. The comparison with Figure 7.3 reveals that trip times are slightly smaller. This result comforts us, we can conclude that informed vehicles can reduce the overall average trip time, giving an updated picture of the network to each vehicle.

To better understand how recent information is received at a vehicle, we analyse the information propagation speed on the map. Figure 7.7 shows a zoomed area. In this graph, we plot the average age of the collected information about the bridge highlighted with the (yellow) arrow. In nearby areas the information is on average less than one minute old. In areas that are about 2 km away, information is on average about 3 minutes old. In fact, in the whole simulation area we could rarely find vehicles that were using information that was more than 15 minutes old. This explains why the results shown in Figures 7.6 and 7.3 are so close: our dissemination application can perform close to full-knowledge since traffic trends (i.e. congestion) build up slower than the speed of the disseminated information, giving the vehicles enough time to react.

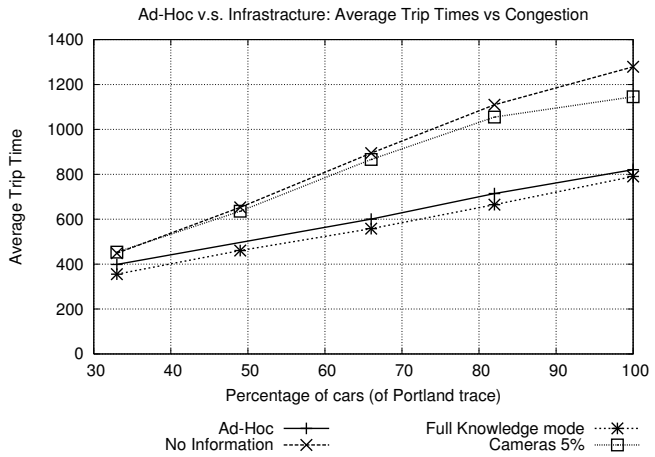


Figure 7.8: Infrastructure versus Ad-Hoc: average trip time.

7.4.6 Infrastructure v.s. Infrastructureless Probing

In this scenario we compare with other existing solutions that use cameras or induction loops in selected street segments and where information is disseminated using cellular networks (e.g., 3G) or FM radio. With such systems vehicles can access updated and accurate information for only a subset of the street segments. We have the opposite situation, information is collected by all the vehicles (and, thus, on almost all street segments) but information is not as recent due to dissemination delay. Additionally, each vehicle may have a different view of the traffic situation since it may have received different traffic updates. For our simulations, we select 5% of the streets to use these cameras and the information is then instantly propagated to all the vehicles without delay.

Figure 7.8 shows the trip times when a different amount of infrastructure is used. We are able to outperform most of the existing solutions just because we collect far more information using each vehicle as a mobile sensor. Information might be delayed as reported in previous graphs, but it is recent enough to avoid congestion hotspots.

7.5 Related work

Traffic congestion has been such an important research topic in the past decades, that it would be impossible to cite all the work carried out on the subject.

A very interesting research stream dates back to the seventies and was originated

in Japan when the Comprehensive Automobile Traffic Control System (CACS) was implemented. The CACS system was designed to test the effectiveness of providing real-time traffic information to vehicles. Tests were run on a 4 x 7 km urban area in Tokyo, area which contained 85 intersections. Traffic information was recorded with 103 roadside units, 255 loop antennas and units installed on 1,000 taxis. The impact of feeding back traffic information was observed on 330 CACS equipped vehicles, which received real-time information on the best routes to reach a destination. Following this experiment many researchers began to work on the impact of traffic information dissemination on traffic.

Assuming all vehicles were able to have perfect information on traffic conditions and that traffic would reach an equilibrium, a shortest path algorithm that finds the best route minimising travel time for each vehicle does not attain the global optimum. This result, due to Beckmann et al. [Beckmann M. 56], is one of the main arguments underlying the scepticism towards navigation systems which implement traffic guided routing. On the other hand, no proof exists that a sub-optimal solution is worse than real traffic in the majority of cases.

Results shown in [Jayakrishn. 90] point in the same direction. They show the impact of traffic information on a vehicular network, as the penetration ratio varies. Higher penetration ratios lead to poor overall performance, from their results a fully informed traffic network attains the same performance as a system with no information feedback. We should note that this and other studies [Arnott 89, Al-Deek 98] have been performed on very simple traffic networks, far from the complexity of the network that is under study in this work.

The idea of using traffic-informed navigation units finds a new stream of interest with [K. Sanwal 95], which moves one step forward on the information feedback side, suggesting the use of cellular and GPS technologies to feed back travel time estimates. This work defines more precisely, using new technologies, how a more centralised traffic information system may be implemented, but does not investigate the impact a traffic information system might have on traffic.

More recently authors of [Sommer 08] implement a bidirectionally coupled simulator, integrating a vehicular simulator and a telecommunications simulator. The analysis

of the impact of a smart navigation system is limited to a test case with 200 vehicles that leave a location and all head to the same destination.

7.6 Conclusions

In this chapter we have described a decentralised approach to road traffic management, one which takes advantage of the existence of in-vehicle connectivity and sensing. Our approach allows the recalculation of the best routes to a given destination while on the road, based on the collected information. In collaboration with UCLA, we also presented our novel testing framework, which allows realistic movement to be considered and to account for vehicle rerouting during the simulation.

The results indicate that the majority of the drivers will benefit from a dissemination system integrated with the navigation system due to the fact that traffic conditions build up much more slowly than our dissemination. This allows drivers to have a quite realistic view of the traffic conditions between their current position and their destination and select the most appropriate route.

8

Conclusions

In this thesis, we have presented the design, implementation and evaluation of a content dissemination framework for vehicular networks. We proposed that two opposing techniques are required: i) *Push-based* that allows popular information to be published proactively to a group of vehicles based on their interests, and ii) *Pull-based* that allows vehicles to explicitly request custom information. Furthermore, we have demonstrated that navigation systems provide valuable information that can be exploited in order to route and maintain information in specific geographic regions. Moreover, we showed how we can use it to automatically request and filter information which might be relevant to a vehicle.

We have implemented our framework in Microsoft .NET and MapPoint and tested it using a small number of vehicles. Furthermore, we evaluated our protocols in large-scale simulation environments. Finally, we have created a test-case application to examine whether vehicles can benefit from such a dissemination. We have successfully confirmed that in the presence of such a distributed traffic sharing scheme vehicles can actually

reduce their trip times.

We will now examine in more detail the contributions of this thesis.

8.1 Contribution of the Thesis

- `Navigation system - Mobility patterns:` In this thesis we have demonstrated how we can take advantage of the navigation system's *suggested routes* to route information to certain geographic regions. Previous approaches like [Shah 03, Zhao 04, Pentland 04, Lindgren 03, Musolesi 05, Zhao 06] exploit different mechanisms to route a message to the destination such as statistics of previous encounters, social characteristics, or even bus schedules to find the best carriers to forward messages. In our approach we take another step forward with the use of the navigation information. We designed GeOpps, a novel routing protocol that takes advantage of this interaction to route information in a delay-tolerant manner.

- `Navigation system - Matching interests:` We have also examined how we can use the navigation system to match geographic content. Compared with existing CBR and Publish/Subscribe approaches like [Carzaniga 01, Banavar 99, Cugola 01, Pietzuch 02], we treat subscriptions as general guidelines that, combined with the navigation system, are used to evaluate if the information is relevant. In our approach we use the map, GPS and navigation information (i.e., the suggested route) to evaluate whether a vehicle may be affected by the content of the disseminated information.

- `Use of Content-Based Routing (CBR):` We indicated that the CBR model is the most applicable in order to push information in vehicular networks. This decision was taken due to the fact that information usually concerns specific geographic locations (POIs) and that vehicles are not interested in receiving all the disseminated information but only information that affects them (i.e., about their route or destination). We devised a two-phase communication scheme whereby vehicles advertise their subscriptions (interests) and the neighbours (or available infrastructure) push out the matching notifications. This CBR approach enables us to automatically restrict the dissemination in areas where the information is actually required (i.e., the dissemination is self-constrained in areas

where there are a large number of subscribers). Furthermore, our overhead depends on the popularity of the disseminated information. Previous approaches like [Sormani 06, Korkmaz 04, Xu 04, Dornbush 07, Eichler 06] use geographically constrained variations of epidemic dissemination that spread the information more widely.

- `Persistence:` Moreover, we demonstrated that the dissemination should be persistently maintained: the notification also needs to be delivered to drivers arriving in the area later. This is necessary due to the fact that we concentrate the dissemination on subscribers and, therefore, the notification can fade away when the disseminated information is not popular or when there is a low density of vehicles (e.g., during the night). Existing solutions like Abiding Geocast [Maihofer 05] address this problem by employing periodic flooding or epidemic dissemination to all the vehicles. In our case we use a low number of *replicas* (i.e., master copies) that we either store in local infostations or in normal vehicles. To make sure that these replicas are maintained near their intended locations (homeZones) we used a variation of GeOpps. This persistence mechanism can also be used to develop similar decentralised spatio-temporal protocols.

- `Pull-based approach:` For our pull based approach we allowed vehicles to request custom information from local infostations, effectively using the ad-hoc network as an extension of infrastructure. We devised a novel routing protocol to route the requested information back to *moving vehicles*. Compared to previous approaches that use social theory, encounter probabilities, bus schedules, etc. ([Shah 03, Zhao 04, Pentland 04, Lindgren 03, Musolesi 05]) or localised flooding [Ko 02, Maihöfer 04], we further exploit the mobility information from the navigation system to route the information on the destination's path so as to intercept its route. This approach integrates with our push-based protocol, as it uses the same advertising model (route advertising), allowing vehicles to retrieve content that is not currently pushed (e.g., custom information).

- `Implementation of a working prototype that combines a Navigation System and Network Communication:` We designed and developed a novel system using Microsoft .NET and MapPoint. This system supports the *interaction of a navigation system* (mobility information, suggested routes, GPS, map database) *with network protocols*. For the moment, this system implements

our framework (GeOpps, Pull- and Push-based dissemination) but it is generic enough to support future network protocols and applications that could benefit from the information available in a navigation system. For example location and map-aware network applications.

- `Detailed Evaluation:` We tested our implementation using a small number of real vehicles. We deemed it necessary to run these three tests in three different locations so as to understand and demonstrate the major variations that can be found in vehicular networks (in terms of density, speed, connection time, throughput, etc). Although these tests cannot prove the correctness of our protocols, they helped us to understand the dynamics of our dissemination mechanisms in different scenarios and they provide a proof-of-concept that this kind of system is feasible. Furthermore, we used the experience gained to run large-scale simulations, using realistic synthetic traces of radically different scenarios. These simulations provide an indication that our protocols can efficiently route and disseminate information. The simulator implementation that we developed emulates the use of the navigation system and can be further used to evaluate the performance of similar future protocols.

- `Dynamic Mobility Simulator:` In collaboration with UCLA, we designed and implemented a dynamic mobility simulator: an evaluation platform that combines two separate tools i) a network simulator and ii) a mobility simulator. This platform allows the evaluation of network protocols that can affect the vehicle's mobility decisions. Although similar approaches have recently become available [Wang 07a, Sommer 08], our approach is specifically designed for wireless ad-hoc networks and provides a very accurate model of mobility and network modelling. We designed this simulation platform to investigate whether our ad-hoc applications can help the drivers to avoid problematic areas but it can be further used to evaluate any kind of network protocol and application that could influence mobility decisions.

- `Design of a simple road-traffic collection and estimation module:` We designed a simple traffic collection application. We measured end-to-end street delays and we used our dissemination framework to publish the collected information to other vehicles. We implemented a simple Bayesian estimator to evaluate the collected information and determine the current traffic conditions

on each street segment. We used Dijkstra's algorithm to select the best possible route. Although these methods are simple and well-known, it is a first step towards interpreting collected traffic information through an ad-hoc vehicular network in order to make individual decisions so as to minimise trip time.

- `Evaluation of the impact of dissemination mechanisms on drivers:` Finally, we evaluated what the impact of the disseminated information is. In our study, we showed that, although the distributed dissemination of information does not always result in full knowledge and that although the collected information might have been collected seconds or even minutes before, the majority of the vehicles will benefit from such a system. This work is the first step towards the evaluation of the impact of an ad-hoc dissemination system to the vehicular fleet.

8.2 Future work

There are multiple possible research opportunities that can spark off from the work and tools presented in this thesis. In this section we outline some of the most promising research directions.

First of all, we would like to investigate possible *data aggregation techniques*. Currently, each piece of information is individually disseminated. This may not cause additional overhead in scenarios where infrastructure is deployed, but it is clearly not efficient when only V2V communication is available. In the push-based case, we would like to aggregate notifications that are disseminated in the same areas so as to assign them at the same homeZones. Furthermore, aggregation techniques like those presented in [Caliskan 06] can be used to further optimise performance: we would like to allow vehicles to receive summarised information about distant areas (e.g., general traffic conditions in the city that they will drive through in 5 miles) and more detailed (fine-grained) information as they get closer. This will require a content-based aggregation mechanism that will correlate multiple notifications as they spread away from the POI. Similarly, notifications of the same type for adjacent POIs can be combined (e.g., we don't need to provide traffic information for each block of London's Oxford street if the whole road has the same traffic congestion levels). Similarly, in the pull-based approach we would like to use a multi-request, multi-reply model which allows aggregation of replies and content caching. In this model we would allow content that is pulled from

the same general location to be aggregated and disseminated as one notification that can be split further during the routing process.

Furthermore, we would also like to automatically push content that is requested (pulled) by a large number of vehicles. Indeed, information that might have been considered as non-popular (e.g., free parking spots in a super market) may be eventually pulled by many vehicles in an area (e.g., near a highway ramp leading to the super market). We would like to use trend-prediction techniques to identify information that is becoming popular and automatically cache (push) this information to these areas.

Currently, our protocol uses simple ways of selecting the homeZones (the key areas where the information is cached for persistence). Although our two-way dissemination process does not cause any dissemination overhead when there are no subscribers, maintaining replicas in infrastructure-less scenarios does. We would like to enhance the homeZone allocation to take advantage of areas where there is a high probability of meeting subscribers. This will avoid creating replicas that simply do not notify any vehicle. One solution (presented in [Leontiadis 07c]) would be to automatically delete replicas if no subscribers are met after a time period, but we would like to provide more efficient solutions where the map database, together with historical information, can be used to evaluate how effective a replica will be.

Finally, as we discussed earlier, this thesis did not look into security and privacy concerns. This is an important issue and, as future work, we are planning to include in our framework security, trust and privacy mechanisms in order to make sure that the disseminated information is trustworthy and that the driver's privacy is ensured.

The central inference that emerges from this study can be condensed into the fact that it is possible to disseminate information in vehicular networks by exploiting valuable tools such as the navigation system and that this information is indeed useful to the drivers. Furthermore, vehicular networks are slowly becoming a reality and we believe that they will play an important role in our future lives as they can provide a number of useful applications: increase road safety, maximise road capacity, disseminate warnings, automatic manage traffic, etc. Moreover, navigation systems are becoming more and more popular and are constantly enriched with new features and information. We believe that this work is the first direction in merging these two technologies.

Bibliography

[Adler 06] Christian Adler, Robert Eigner, Christoph Schroth & Markus Strassberger. *Context-Adaptive Information Dissemination in Vanets - Maximizing the Global Benefit.* In C. E. Palau Salvador, editeur, Communication Systems and Networks, pages 7–12. IASTED/ACTA Press, 2006.

[Al-Deek 98] Haitham M. Al-Deek, Asad J. Khattak & Paramsothy Thananjeyan. *A Combined Traveler Behavior and System Performance Model With Advanced Traveler Information Systems.* Transportation Research Part A: Policy and Practice, vol. 32, no. 7, pages 479–493, September 1998.

[Arnott 89] Richard Arnott. *Does Providing Information to Drivers Reduce Traffic Congestion?* Discussion Papers 864, Northwestern University, Center for Mathematical Studies in Economics and Management Science, June 1989.

[Banavar 99] G. Banavar, T. D. Chandra, B. Mukherjee, J. Nagara Jarao, R. E. Strom & D. C. Sturman. *An Efficient Multicast Protocol for Content-Based Publish-Subscribe Systems.* In ICDCS '99: Proceedings of the 19th IEEE International Conference on Distributed Computing Systems, page 262, Washington, DC, USA, 1999. IEEE Computer Society.

[Beckmann M. 56] C. McGuire Beckmann M. & C. Winsten. *Studies in the Economics of Transportation. Cowles Commission Monograph.* In New Haven,

CT: Yale University Press, pages 117–124, 1956.

[Bose 01] Prosenjit Bose, Pat Morin, Ivan Stojmenovic & Jorge Urrutia. *Routing with Guaranteed Delivery in Ad Hoc Wireless Networks.* ACM Journal of Wireless Networks, vol. 7, no. 6, pages 609–616, 2001.

[Burcea 03] Ioana Burcea & Hans-Arno Jacobsen. *L-ToPSS - Push-Oriented Location-Based Services.* In 4th VLDB Workshop on Technologies for E-Services (TES'03), pages 131–142, 2003.

[Burgess 06] John Burgess, Brian Gallagher, David Jensen & Brian Neil Levine. *MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks.* In Proc. IEEE INFOCOM, Barcelona Spain, pages 1–11, April 2006.

[Bychkovsky 06] V. Bychkovsky, K. Chen, M. Goraczko, H. Hu, B. Hull, A. Miu, E. Shih, Y. Zhang, H. Balakrishnan & S. Madden. *The CarTel Mobile Sensor Computing System.* In SenSys '06, pages 383–384, New York, NY, USA, 2006. ACM Press.

[Caliskan 06] Murat Caliskan, Daniel Graupner & Martin Mauve. *Decentralized Discovery of Free Parking Places.* In VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks, pages 30–39, New York, NY, USA, 2006.

[Camp 02] T. Camp, J. Boleng & V. Davies. *A Survey of Mobility Models for Ad Hoc Network Research.* Wireless Communications and Mobile Computing (WCMC), vol. 2, no. 5, pages 483–502, 2002.

[Camp 03] Tracy Camp & Yu Liu. *An Adaptive Mesh-Based Protocol for Geocast Routing.* Journal of Parallel and Distributed Computing, vol. 63, no. 2, pages 196–213, 2003.

[Caporuscio 03] Mauro Caporuscio, Antonio Carzaniga & Alexander L. Wolf. *Design and Evaluation of a Support Service for Mobile, Wireless Publish/Subscribe Applications.* IEEE Transactions on Software Engineering, vol. 29, no. 12, pages 1059–1071, Dec 2003.

[Carzaniga 01] Antonio Carzaniga, David S. Rosenblum & Alexander L. Wolf. *Design and Evaluation of a Wide-Area Event Notification Service.* ACM Transactions on Computer Systems, vol. 19, no. 3, pages 332–383, 2001.

[Chen 03] Xiaoyan Chen, Ying Chen & Fangyan Rao. *An Efficient Spatial Publish/Subscribe System for Intelligent Location-Based Services.* In DEBS '03, pages 1–6, New York, NY, USA, 2003. ACM Press.

[Cilia 03] M. Cilia, L. Fiege, C. Haul, A. Zeidler & A. P. Buchmann. *Looking Into the Past: Enhancing Mobile Publish/Subscribe Middleware.* In 2nd international workshop on Distributed event-based systems. (2003), 2003.

[Costa 03] Paolo Costa, Matteo Migliavacca, Gian Pietro Picco & Gianpaolo Cugola. *Introducing Reliability in Content-Based Publish-Subscribe Through Epidemic Algorithms.* In DEBS '03: Proceedings of the 2nd international workshop on Distributed event-based systems, pages 1–8, New York, NY, USA, 2003. ACM Press.

[Costa 04] Paolo Costa, Matteo Migliavacca, Gian Pietro Picco & Gianpaolo Cugola. *Epidemic Algorithms for Reliable Content-Based Publish-Subscribe: An Evaluation.* In ICDCS '04: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04), pages 552–561, Washington, DC, USA, 2004. IEEE Computer Society.

[Costa 05] Paolo Costa & Gian Pietro Picco. *Semi-Probabilistic Content-Based Publish-Subscribe.* In ICDCS '05: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05), pages 575–585, Washington, DC, USA, 2005. IEEE Computer Society.

[Cugola 01] Gianpaolo Cugola, Elisabetta Di Nitto & Alfonso Fuggetta. *The JEDI Event-Based Infrastructure and Its Application to the Development of the OPSS WFMS.* IEEE Trans. Softw. Eng., vol. 27, no. 9, pages 827–850, 2001.

[Cugola 02a] G. Cugola, G. Picco & A. Murphy. *Towards Dynamic Reconfiguration of Distributed Publish-Subscribe Middleware.* In Third International Workshop on Software Engineering and Middleware, 2002.

[Cugola 02b] Gianpaolo Cugola & H.-Arno Jacobsen. *Using Publish/Subscribe Middleware for Mobile Systems.* SIGMOBILE Mobile Computing and Communication Review, vol. 6, no. 4, pages 25–33, 2002.

[Cugola 05] Gianpaolo Cugola & Jose Enrique Munoz de Cote. *On Introducing Location Awareness in Publish-Subscribe Middleware.* In ICDCSW '05: Proceedings of the Fourth International Workshop on Distributed Event-Based Systems (DEBS) (ICDCSW'05), pages 377–382, Washington, DC, USA, 2005. IEEE Computer Society.

[Dash 09] Dash. *Dash Navigation, Inc.* In `http://www.dash.net`, 2009.

[Dornbush 07] Sandor Dornbush & Anupam Joshi. *StreetSmart Traffic: Discovering and Disseminating Automobile Congestion Using VANET's.* In Proceedings of the 65^{th} Vehicular Technology Conference, Dublin, Ireland, April 2007.

[Drytkiewicz 03] W. Drytkiewicz, S. Sroka, V. Handziski, A. Koepke & H. Karl. *A Mobility Framework for OMNeT++.* 3rd International OMNeT++ Workshop, at Budapest University of Technology and Economics, Department of Telecommunications Budapest, Hungary, January 2003.

[Eichler 06] Stephan Eichler, Christoph Schroth, Timo Kosch & Markus Strassberger. *Strategies for Context-Adaptive Message Dissemination in Vehicular Ad Hoc Networks.* In Proceedings of the Second International Workshop on Vehicle-to-Vehicle Communications (V2VCOM), July 2006.

[Eriksson 08] Jakob Eriksson, Hari Balakrishnan & Samuel Madden. *Cabernet: Vehicular Content Delivery Using WiFi.* In Proc. of ACM International Conference on Mobile Computing and Networking (MobiCom), 2008.

[Eugster 05] Patrick Th. Eugster, Benoy't Garbinato & Adrian Holzer. *Location-based Publish/Subscribe.* In Fourth IEEE International Symposium on Network Computing and Applications, pages 279–282, 2005.

[Fall 03] Kevin Fall. *A Delay-Tolerant Network Architecture for Challenged Internets.* In SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, pages 27–34, New York, NY, USA, 2003. ACM Press.

[Fiege 03] Ludger Fiege, Felix C. Gärtner, Oliver Kasten & Andreas Zeidler. *Supporting Mobility in Content-Based Publish/Subscribe Middleware.* In Middleware'03, pages 103–122, 2003.

[Fubler 03] Holger Fubler, Hannes Hartenstein, Dieter Vollmer, Martin Mauve & Michael Kasemann. *MobiCom poster: Location-Based Routing for Vehicular Ad-Hoc Networks.* SIGMOBILE, vol. 7, no. 1, pages 47–49, 2003.

[G. Marfia 07] G. Pau G. Marfia P. Lutterotti. *MobiTools: An Integrated Toolchain for Mobile Ad Hoc Networks.* In UCLA Technical Report #070019, 2007.

[Garcia-Aceves 99] J. J. Garcia-Aceves & Marcelo Spohn. *Source-Tree Routing in Wireless Networks.* In ICNP '99: Proceedings of the Seventh Annual International Conference on Network Protocols, page 273, Washington, DC, USA, 1999. IEEE Computer Society.

[Gerlach 06] Matthias Gerlach. *Assessing and Improving Privacy in VANETs.* In in Embedded Security in Cars (ESCAR, 2006.

[Gerlach 07a] Matthias Gerlach. *Trust for Vehicular Applications.* In ISADS '07: Proceedings of the Eighth International Symposium on Autonomous Decentralized Systems, pages 295–304, Washington, DC, USA, 2007. IEEE Computer Society.

[Gerlach 07b] Matthias Gerlach & Felix Gttler. *Privacy in VANETs using Changing Pseudonyms - Ideal and Real.* In VTC Spring, pages 2521–2525. IEEE, 2007.

[Giordano 01] Silvia Giordano, Ivan Stojmenovic & Ljubica Blazevic. *Position Based Routing Algorithms For Ad Hoc Networks: A Taxonomy.* In Ad Hoc Wireless Networking, pages 103–136. Kluwer, 2001.

[GMSF 09] GMSF. *The GMSF Trace Generator:* `http://gmsf.hypert.net`. 2009.

[Haas 06] Zygmunt J. Haas, Joseph Y. Halpern & Li Li. *Gossip-based Ad-Hoc Routing.* IEEE/ACM Transactions in Networking, vol. 14, no. 3, pages 479–491, 2006.

[Haas 09] Jason J. Haas, Yih-Chun Hu & Kenneth P. Laberteaux. *Design and Analysis of a Lightweight Certificate Revocation Mechanism for VANET.* In VANET '09: Proceedings of the sixth ACM international workshop on Vehicular Internetworking, pages 89–98, New York, NY, USA, 2009. ACM.

[Herrtwich 05] Ralf Guido Herrtwich. *Communicating Vehicles - Communicating Roadways: New Approaches to Driver Information and Road Safety.* In Invited Talk at The Eleventh Annual International Conference on Mobile Computing and Networking (MobiCom'05), Cologne Germany, 2005.

[Huang 03] Yongqiang Huang & Hector Garcia-Molina. *Publish/Subscribe Tree Construction in Wireless Ad-Hoc Networks.* In Mobile Data Management, pages 122–140, 2003.

[Huang 04] Yongqiang Huang & Hector Garcia-Molina. *Publish/Subscribe in a Mobile Environment.* Wireless Networks'04, vol. 10, no. 6, pages 643–652, 2004.

[IEEE 03] IEEE. *Wireless LAN Medium Access Control (MAC) and Physical*

Layer (PHY) specifications: Higher-speed Physical Layer Extension in the 2.4 GHz Band. Rapport technique IEEE Std 802.11b-1999 (R2003), IEEE, 2003.

[IEEE 09] IEEE. *IEEE P802.11-Task Group P: Wireless Access for the Vehicular Environment (WAVE).* 2009.

[Imielinski 99] Tomasz Imielinski & Julio C. Navas. *GPS-based Geographic Addressing, Routing, and Resource Discovery.* ACM Journal of Communications, vol. 42, no. 4, pages 86–92, 1999.

[Jacquet 01] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum & L. Viennot. *Optimized Link State Routing Protocol for Ad-Hoc Networks.* In Proceedings of the 5th IEEE Multi Topic Conference (INMIC 2001), 2001.

[Jain 04] Sushant Jain, Kevin Fall & Rabin Patra. *Routing in a Delay Tolerant Network.* In SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications, pages 145–158, New York, NY, USA, 2004. ACM Press.

[Jayakrishn. 90] R. Jayakrishn. & Hani S. Mahmassani. *Dynamic simulation-Assignment Methodology to Evaluate In-Vehicle Information Strategies in Urban Traffic Networks.* In WSC' 90: Proceedings of the 22nd conference on Winter simulation, pages 763–769, Piscataway, NJ, USA, 1990. IEEE Press.

[K. Sanwal 95] J. Walrand K. Sanwal. *Vehicles As Probes.* In University of California, Berkeley, Technical Report, UCB-ITS-PWP-95-11, 1995.

[Karp 00] Brad Karp & H. T. Kung. *GPSR: Greedy Perimeter Stateless Routing for Wireless Networks.* In Mobile Computing and Networking, pages 243–254, 2000.

[Killijian 01] M. Killijian, R. Cunningham, R. Meier, L. Mazare & V. Cahill. *To-*

wards Group Communication for Mobile Participants. In Proceedings of the 1st ACM Workshop on Principles of Mobile Computing (POMC 2001), pages 75–82, 2001.

[Ko 98] Young-Bae Ko & Nitin H. Vaidya. *Location-Aided Routing (LAR) in Mobile Ad Hoc Networks.* In Mobile Computing and Networking, pages 66–75, 1998.

[Ko 00] Young-Bae Ko & N.H. Vaidya. *GeoTORA: a Protocol for Geocasting in Mobile Ad-Hoc Networks.* IEEE International Conference on Network Protocols, vol. 1, page 240, 2000.

[Ko 02] Young-Bae Ko & Nitin H. Vaidya. *Flooding-Based Geocasting Protocols for Mobile Ad Hoc Networks.* MONET, vol. 7, no. 6, pages 471–480, 2002.

[Kobayashi 99] K Kobayashi & H Tatano. *Information and Rational Expectations in Modeling Driver Information Systems: A Welfare Measurement.* In Behavioural and Network Impacts of Driver Information Systems, Ashgate Publishing Limited, 1999.

[Korkmaz 04] Gökhan Korkmaz, Eylem Ekici, Füsun Özgüner & Ümit Özgüner. *Urban Multi-hop Broadcast Protocol for Inter-Vehicle Communication Systems.* In VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks, pages 76–85, New York, NY, USA, 2004. ACM.

[Kosch 02] Timo Kosch, Christian Schwingenschlgl & Li Ai. *Information Dissemination in Multihop Inter-Vehicle Networks - Adapting the Ad-hoc On-demand Distance Vector Routing Protocol (AODV).* In The IEEE 5th International Conference on Intelligent Transportation Systems, pages 685– 690, 2002.

[Krumm 07] John Krumm & Eric Horvitz. *Predestination: Where Do You Want to Go Today?* Computer, vol. 40, no. 4, pages 105–107, 2007.

[Lebrun 05] J. Lebrun, Chen-Nee Chuah, D. Ghosal & Michael Zhang. *Knowledge-Based Opportunistic Forwarding in Vehicular Wireless Ad Hoc Networks.* Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st, vol. 4, pages 2289–2293, 2005.

[Leontiadis 07a] Ilias Leontiadis. *Publish/Subscribe Notification Middleware for Vehicular Networks.* In MDS' 07. Collocated with Middleware 2007, Newport Beach, CA, USA, November 2007. IEEE Press.

[Leontiadis 07b] Ilias Leontiadis & Cecilia Mascolo. *GeOpps: Opportunistic Geographical Routing for Vehicular Networks.* In Proceedings of the IEEE Workshop on Autonomic and Opportunistic Communications. (Colocated with WOWMOM07), Helsinki, Finland, June 2007. IEEE Press.

[Leontiadis 07c] Ilias Leontiadis & Cecilia Mascolo. *Opportunistic Spatio-Temporal Dissemination System for Vehicular Networks.* In In Proceedings of the First International Workshop on Mobile Opportunistic Networking (ACM/SIGMOBILE MobiOpp 2007). Co-located with MobiSys 2007, Puerto Rico, USA, June 2007.

[Leontiadis 09a] Ilias Leontiadis, Paolo Costa & Cecilia Mascolo. *A hybrid approach for content-based publish/subscribe in vehicular networks.* In Journal of Pervasive and Mobile Computing, Elsevier. (in press, doi:10.1016/j.pmcj.2009.07.016), 2009.

[Leontiadis 09b] Ilias Leontiadis, Paolo Costa & Cecilia Mascolo. *Persistent Content-based Information Dissemination in Hybrid Vehicular Networks.* In Proceedings of the 7^{th} IEEE International Conference on Pervasive Computing and Communications (PERCOM), 2009.

[Leontiadis 09c] Ilias Leontiadis, Gustavo Marfia, David Mack, Giovanni Pau, Cecilia Mascolo & Mario Gerla. *On The Effectiveness of an Opportunistic Traffic Management System for Vehicular Networks.* In Journal of Intelligent Transportation Systems, IEEE (under submission, T-ITS-09-07-0180), 2009.

[Leontiadis 10a] Ilias Leontiadis, Paolo Costa & Cecilia Mascolo. *Extending Access Point Connectivity through Opportunistic Routing in Vehicular Networks.* In In the 29th IEEE International Conference on Computer Communications (INFOCOM'10), mini-track. San Diego, USA, 2010.

[Leontiadis 10b] Ilias Leontiadis, Paolo Costa & Cecilia Mascolo. *GREAT: Geographic Routing for Extending Access-Point Transmissions in Vehicular Networks.* In IEEE Journal on Selected Areas in Communications, under review, 2010.

[Liao 00] W.-H. Liao. *GeoGRID: A Geocasting Protocol for Mobile Ad Hoc Networks Based on GRID.* Journal of Internet Technologies, vol. 1, no. 2, pages 23–32, December 2000.

[Lindgren 03] Anders Lindgren, Avri Doria & Olov Schelén. *Probabilistic Routing in Intermittently Connected Networks.* SIGMOBILE Mobile Computing and Communications Review, vol. 7, no. 3, pages 19–20, 2003.

[Lochert 05a] Christian Lochert, Andreas Barthels, Alfonso Cervantes, Martin Mauve & Murat Caliskan. *Multiple Simulator Interlinking Environment for IVC.* In VANET '05: Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks, pages 87–88, New York, NY, USA, 2005. ACM Press.

[Lochert 05b] Christian Lochert, H. Hartenstein, J. Tian, Holger Fuessler, D. Hermann & M. Mauve. *A Routing Strategy for Vehicular Ad Hoc Networks in City Environments.* pages 156–161, 2005.

[Luna 09] Lynnette Luna. *AT&T chief: Operators aren't prepared for onslaught of data traffic.* In `http://tinyurl.com/at-t-chief`, 2009.

[Maihöfer 04] Christian Maihöfer. *A Survey of Geocast Routing Protocols.* IEEE Communications Surveys & Tutorials, vol. 6, pages 32–42, 2004.

[Maihofer 05] Christian Maihofer, Tim Leinmuller & Elmar Schoch. *Abiding Geocast: Time–Stable Geocast for Ad-Hoc Networks.* In VANET '05:

Proceedings of the 2nd ACM International Workshop on Vehicular Ad-Hoc Networks, pages 20–29, New York, NY, USA, 2005. ACM Press.

[Mauve 01] Martin Mauve, Juorg Widmer & Hannes Hartenstein. *A Survey on Position-based Routing in Mobile Ad-Hoc Networks.* IEEE Network, vol. 15, no. 6, pages 30–39, 2001.

[Meier 02] R. Meier & V. Cahill. *STEAM: Event-Based Middleware for Wireless Ad-Hoc Networks.* In 1st International Workshop on Distributed Event-Based Systems (DEBS '02), Vienna, Austria, pages 639–644, 2002.

[Mohapatra 04] Prasant Mohapatra, Chao Gui & Jian Li. *Group Communications in Mobile Ad Hoc Networks.* Computer, vol. 37, no. 2, pages 52–59, 2004.

[Mottola 05] Luca Mottola, Gianpaolo Cugola & Gian Pietro Picco. *Tree Overlays for Publish-Subscribe in Mobile Ad Hoc Networks.* Rapport technique, Politecnico di Milano, 2005.

[Murthy 95] Shree Murthy & J. J. Garcia-Luna-Aceves. *A Routing Protocol for Packet Radio Networks.* In Mobile Computing and Networking, pages 86–95, 1995.

[Musolesi 05] M. Musolesi, S. Hailes & C. Mascolo. *Adaptive Routing for Intermittently Connected Mobile Ad-Hoc Networks.* pages 183–189, 2005.

[Naumov 06] Valery Naumov, Rainer Baumann & Thomas Gross. *An Evaluation of Inter-Vehicle Ad-Hoc Networks Based on Realistic Vehicular Traces.* In MobiHoc '06, pages 108–119, New York, NY, USA, 2006. ACM Press.

[Nissan Mot. 09] Nissan Mot. *Nissan Motors to Test Intelligent Transportation System In Kanagawa.* http://www.nissan-global.com/EN/NEWS/2006/_STORY/060915-02-e.html, May

2009.

[Ogier 04] R. Ogier. *Topology Dissemination Based on ReversePath Forwarding (TBRPF).* In RFC 3684, 2004.

[OMNET++ 09] OMNET++. *The OMNET++ Community Website: http://www.omnetpp.org/.* 2009.

[Ott 05] J. Ott & D. Kutscher. *A Disconnection-Tolerant Transport for Drive-Thru Internet Environments.* INFOCOM 2005, vol. 3, pages 1849–1862 vol. 3, 2005.

[Papadim. 08] Panos Papadim., Levente Buttyan, Tamas Holczer, Elmar Schoch, Julien Freudiger, Maxim Raya, Zhendong Ma, Frank Kargl, Antonio Kung & Jean-Pierre Hubaux. *Secure Vehicular Communications: Design and Architecture.* EEE Communications Magazine, vol. 46, no. 11, page 2–8, 11/2008 2008.

[Pentland 04] A. Pentland, R. Fletcher & A. Hasson. *DakNet: Rethinking Connectivity in Developing Nations.* Computer, vol. 37, no. 1, pages 78–83, 2004.

[Perkins 94] Charles E. Perkins & Pravin Bhagwat. *Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers.* In ACM Conference on Communications Architectures, Protocols and Applications, SIGCOMM '94, London, UK, pages 234–244. ACM, ACM, August 1994.

[Pietzuch 02] Peter Pietzuch & Jean Bacon. *Hermes: A Distributed Event-Based Middleware Architecture.* In J. Bacon, L. Fiege, R. Guerraoui, A. Jacobsen & G. Mühl, editeurs, In Proceedings of the 1st International Workshop on Distributed Event-Based Systems (DEBS'02), pages 611–618, July 2002.

[Resendes 08] Raymond Resendes. *The New Grand Challenge - Deploying Vehicle Communications.* In The Fifth ACM International Workshop on

VehiculAr Inter-NETworking (VANET 2008), 2008.

[Robinson 06] C. L. Robinson, L. Caminiti, D. Caveney & K. Laberteaux. *Efficient Coordination and Transmission of Data for Cooperative Vehicular Safety Applications.* In VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks, pages 10–19, New York, NY, USA, 2006. ACM Press.

[Sampiget. 05] Krishna Sampiget., Leping Huang, Mingyan Li, Radha Poovendran, Kanta Matsuura & Kaoru Sezaki. *Caravan: Providing Location Privacy for Vanet.* In in Embedded Security in Cars (ESCAR, 2005.

[Sedgewick 84] R. Sedgewick & J. S. Vitter. *Shortest Paths In Euclidean Graphs.* In SFCS '84: Proceedings of the 25th Annual Symposium onFoundations of Computer Science, 1984, pages 417–424, Washington, DC, USA, 1984. IEEE Computer Society.

[Serna 08] Jetzabel Serna, Jesus Luna & Manel Medina. *Geolocation-Based Trust for Vanet's Privacy.* Information Assurance and Security, International Symposium on, vol. 0, pages 287–290, 2008.

[Shah 03] R. C. Shah, S. Roy, S. Jain & W. Brunette. *Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks.* In Sensor Network Protocols and Applications, pages 30–41, 2003.

[Shirshanka 05] Alok Nandan Shirshanka, Alok N, Shirshanka Das, Biao Zhou, Giovanni Pau & Mario Gerla. *AdTorrent: Digital Billboards for Vehicular Networks.* In in Proc. of IEEE/ACM International Workshop on Vehicleto-Vehicle Communications (V2VCOM), pages 77–82, 2005.

[Sivaharan 05] Thirunavukkarasu Sivaharan, Gordon S. Blair & Geoff Coulson. *GREEN: A Configurable and Re-configurable Publish-Subscribe Middleware for Pervasive Computing.* In OTM Conferences (1), pages 732–749, 2005.

[Skordylis 08] Antonios Skordylis & Niki Trigoni. *Delay-bounded Routing in Vehic-*

ular Ad-hoc Networks. In ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 2008.

[SNT 09] SNT. *Scalable Network Technologies, Qualnet Network Simulator.* In `http://www.scalable-networks.com/`, 2009.

[Sommer 08] C. Sommer, Zheng Yao, R. German & F. Dressier. *Simulating the Influence of IVC on Road Traffic Using Bidirectionally Coupled Simulators.* In INFOCOM 2008: IEEE Conference on Computer Communications Workshops, pages 1–6, 2008.

[Sormani 06] Davide Sormani, Gabriele Turconi, Paolo Costa, Davide Frey, Matteo Migliavacca & Luca Mottola. *Towards Lightweight Information Dissemination in InterVehicular Networks.* In Proceedings of the 3^{rd} ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2006), Los Angeles, California, USA, 2006.

[TomTom 09] TomTom. *TomTom International BV.* In `www.tomtom.com`, 2009.

[Traces 09] ETH Vehicular Traces. *http://lst.inf.ethz.ch/ad-hoc/car-traces*, 2009.

[U.S.C.B. 09] U.S.C.B. *U.S. Census Bureau's Mapping and Cartographic Resources.* In `http://tiger.census.gov/`, 2009.

[Vahdat 00] A. Vahdat & D. Becker. *Epidemic Routing for Partially Connected Ad Hoc Networks.* In Technical Report CS 200006, Duke University, 2000.

[Varga 01] András Varga. *The OMNET++ Discrete Event Simulation System.* Proceedings of the European Simulation Multiconference (ESM'2001), 2001.

[Wang 07a] S. Y. Wang, C. L. Chou, Y. H. Chiu, Y. S. Tzeng, M. S. Hsu, Y. W. Cheng, W. L. Liu & T. W. Ho. *NCTUns 4.0: An Integrated Simulation Platform for Vehicular Traffic, Communication, and Network Researches.* In Vehicular Technology Conference, 2007. VTC-2007

Fall. 2007 IEEE 66th, pages 2081–2085, 2007.

[Wang 07b] Zhou Wang & Chunxiao Chigan. *Cooperation Enhancement for Message Transmission in VANETs.* Journal of Wireless Personal Communication, vol. 43, no. 1, pages 141–156, 2007.

[Xiangchuan 01] Chen Xiangchuan & Murphy Amy L. *Enabling Disconnected Transitive Communication in Mobile Ad Hoc Networks.* In Workshop on Principles of Mobile Computing, colocated with PODC'01, pages 21–27, 2001.

[Xu 04] Bo Xu, A. Ouksel & O. Wolfson. *Opportunistic Resource Exchange in Inter-Vehicle Ad-Hoc Networks.* In Proceedings of the IEEE International Conference on Mobile Data Management (MDM 2004), pages 4–12, 2004.

[Xu 05] Zhengdao Xu & Hans-Arno Jacobsen. *Efficient Constraint Processing for Location-Aware Computing.* In MDM '05: Proceedings of the 6th international conference on Mobile data management, pages 3–12, New York, NY, USA, 2005. ACM Press.

[Zeidler 03] Andreas Zeidler & Ludger Fiege. *Mobility Support with REBECA*. In ICDCSW '03: Proceedings of the 23rd International Conference on Distributed Computing Systems, page 354, Washington, DC, USA, 2003. IEEE Computer Society.

[Zhao 04] W. Zhao, M. Ammar & E. Zegura. *A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks.* In MobiHoc '04, pages 187–198, New York, NY, USA, 2004. ACM Press.

[Zhao 06] J. Zhao & G. Cao. *VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks.* In Proceedings of the 25^{th} Conference on Computer Communications (INFOCOM06), 2006.