Design and implementation of simulation tools, protocols and architectures to support service platforms on vehicular networks

Miguel Báguena Albaladejo

Ph.D. advisors:
Dr. Pietro Manzoni
Dr. Carlos Tavares Calafate

June 5, 2017

Acknowledgements

Many people have supported me during this long, tough, hard and heavy process. This is the time to acknowledge and return all the support and the kindliness that I got from them.

In the first place I want to thank my parents. They always give me any help and support I could need.

Professors Carlos Tavares Calafate and Pietro Manzoni were my advisors in this thesis. They always tried to make a researcher out of me.

When I first met Carlos, he was the lecturer of the lab sessions of one of my subjects. I decided to do the final project of my bachelor's degree under his guidance and, after that, I decided to stay with him until the end. He taught me the basic principles of research from the very beginning.

I was collaborating with Pietro more closely during the last years of my PhD. I easily realized that he was a quick-witted researcher with strong management skills and a classic elegance, which were only a few of their natural gifts. There is only one word that can describe him: Italian.

The third leg of this stool that support my research during these years was Juan Carlos Cano Escribá. He did his best to review my work to give me very valuable advice, and he was my tripmate to the conferences where I showed my work.

There was a lot of people that shared lab with me during the years I was doing my PhD. Some of them were Oscar Alvear, Danilo Vargas, Jorge Herrera, Jorge Luzuriaga, William Zamora, etc, but there are three of them that I must highlight. They are (in alphabetical order) Johann Marcelo Márquez Barja, Sergio Martínez Tornell and Álvaro Torres Cortés. We stayed together for almost the whole period of my PhD, and they created the perfect working environment. Without them, reaching the end of this long distance run would have been impossible.

I also want to thank my friends in Valencia, who kept me sane during the time I was living there. Ignacio Domingo was my flatmate during my first years

in Valencia. Throughout my PhD, we keep into contact for films and football. I always could count on Ana Roig and Abigail Casañ, whose house was my base camp most of the times. I want also thank the members of the *GUbik project*, the *Floral Pretty Bonets* cabal, and *Maldito Lag!*: José López, who was my labmate during our master's; Juan Martinez, who was my flatmate for a brief period of time; and Alfonso Pérez and Estíbaliz Parcero, who I went out with many times.

As Javier Sierra would say, during my many trips I have met many people. At Imperial College I met Peter Pietzuch. He showed me two very important things: how research in a first level university works, and what being a British gentleman means. I had skilled and friendly colleagues there, such as Andreas Pamboris, Lukas Ruppretch, and Raúl Castro.

At North Carolina State University I worked with Mihail Sichitiu. He made my stay there a wonderful experience and gave me the opportunity of participating in my first programming contest. I also want to thank Lee Jones, who gave Federico, Marta and myself the real taste of being an American person.

At Trinity College, I worked with Douglas Leith. It was an intense experience and I learned a lot about research under the supervision of a great researcher. My colleagues there where also very helpful and welcoming. Andrés, Cristina, Victor and Bahar were great people that I was lucky to meet.

At Google Switzerland, I worked with Luis Quesada Torres. He became my manager during the internship and he did his best to develop me professionally and personally. I must thank him too many things to be listed here. Xavier Martínez Palau was also a key piece in my time there, helping me to meet very interesting people. I spent most of my time there with Joan Pastor, who I team up with to go through the hiring process. I also had a great time with Marcos Calvo, Jorge Gorbe and Kevin Montalvá, with whom I shared lunch, dinner and board games.

A special place in my heart is reserved for the iNiT research group and its "initliness". They show me an alternative way of research that can be productive, too. I want to thank Francisco Martínez, the head of the group, Julio Sangüesa and Vicente Torres, who helped me during the tedious process of writing this document.

There is an endless list of people that I should include here (Arnoldo Díaz, José Cano, Sabela Ramos, etc.) but it is very difficult for me to fit an endless list in a finite text. However, I want to thank all the people that I did not include in this section, and also did their best to help me to finish this PhD.

Resumen

Los productos relacionados con los Sistemas Inteligentes de Transporte (ITS) se están transformando en una realidad en nuestras carreteras. Todos los fabricantes de coches comienzan a incluir acceso a internet en sus vehículos y a facilitar su integración con los teléfonos móviles, pero más y más servicios se introducirán en el futuro. La conectividad usando las "redes vehiculares" se convertirá en la piedra angular de cada nueva propuesta, y ofrecer una calidad de servicio adecuada será, obviamente, deseable. Sin embargo, se necesita una gran cantidad de trabajo para que las redes vehiculares ofrezcan un rendimiento similar al de las redes cableadas.

Las redes vehiculares quedan definidas por sus dos características básicas: alto dinamismo, pues los nodos pueden alcanzar una velocidad relativa de más de 250 km/h; y heterogeneidad, por la gran cantidad de propuestas diferentes que los fabricantes están lanzando al mercado. Por ello, para hacer posible el despliegue de servicios sobre ellas, se impone la necesidad de hacer un estudio en profundidad de este entorno, y deben de proponerse y desarrollarse las herramientas adecuadas.

Esta tesis ataca la problemática del despliegue de servicios en estas redes a tres niveles diferentes: (i) el nivel físico y de enlace, mostrando varios análisis en profundidad del medio físico y modelos derivados para su simulación; (ii) el nivel de red, proponiendo un protocolo de difusión de la información para los paquetes IP; y (iii) el nivel de transporte, donde otros protocolos son propuestos para mejorar el rendimiento del transporte de datos.

En primer lugar, se han estudiado y modelado las dos principales tecnologías inalámbricas que se utilizan para la comunicación en redes vehiculares, la rama de estándares 802.11, en concreto 802.11p; y la comunicación celular, en particular LTE. Dado que el estándar 802.11p es un estándar bastante maduro, nos centramos en crear (i) un modelo de propagación y atenuación capaz de replicar el rango de transmisión de dispositivos 802.11p reales, en condiciones de visión directa y obstrucción por pequeños obstáculos, y (ii) un

modelo de visibilidad capaz de simular el efecto de grandes obstáculos, como son los edificios, de una manera realista. Además, proponemos un modelo basado en indicadores de rendimiento de alto nivel (ancho de banda y retardo) para LTE, que facilita la validación y evaluación de aplicaciones.

En el plano de red, se propone un protocolo híbrido, llamado AVE, para el encaminamiento y reenvío de paquetes usando un conjunto de estrategias estándar de enrutamiento. Dependiendo del escenario, AVE elige entre cuatro estrategias diferentes: a) entrega directa a dos saltos, b) Dynamic MANET On-demand (DYMO) c) georouting voraz, y d) una técnica store-carry-and-forward, para adaptar su comportamiento dinámicamente a cada situación.

En el plano de transporte, se propone un protocolo bidireccional de distribución de contenidos en canales con pérdidas que mejora la entrega de contenidos en situaciones en las que la red es un cuello de botella, como las redes inalámbricas. Ha sido diseñado, validado, optimizado, y su rendimiento ha sido analizado en términos de productividad y eficiencia en la utilización de recursos.

Finalmente, a nivel de sistema, proponemos un modelo de computación asistida que permite reducir la latencia en la respuesta a muchas consultas colocando una unidad de computación en el borde de la red, i.e., la red de acceso. Este esquema podría ser usado en redes basadas en 802.11p y en redes celulares, si bien en esta tesis decidimos centrarnos en su evaluación usando redes LTE.

La plataforma presentada en esta tesis combina todos los esfuerzos individuales para crear una plataforma única y eficiente. Este nuevo entorno puede ser usado por cualquier proveedor para mejorar la calidad de la experiencia de usuario en los servicios desplegados sobre redes vehiculares.

Abstract

Products related with Intelligent Transportation Systems (ITS) are becoming a reality on our roads. All car manufacturers are starting to include Internet access in their vehicles and to integrate smartphones directly from the dashboard, but more and more services will be introduced in the near future. Connectivity through "vehicular networks" will become a cornerstone of every new proposal, and offering an adequate quality of service is obviously desirable. However, a lot of work is needed for vehicular networks to offer performances similar to those of the wired networks.

Vehicular networks can be characterized by two main features: high variability due to mobility levels that can reach up to 250 kilometers per hour, and heterogeneity, being that various competing versions from different vendors have and will be released. Therefore, to make the deployment of efficient services possible, an extensive study must be carried out and adequate tools must be proposed and developed. This PhD thesis addresses the service deployment problem in these networks at three different levels: (i) the physical and link layer, showing an exhaustive analysis of the physical channel and models; (ii) the network layer, proposing a forwarding protocol for IP packets; and (iii) the transport layer, where protocols are proposed to improve data delivery.

First of all, the two main wireless technologies used in vehicular networks where studied and modeled, namely the 802.11 family of standards, particularly 802.11p, and the cellular networks focusing on LTE. Since 802.11p is a quite mature standard, we defined (i) a propagation and attenuation model capable of replicating the transmission range and the fading behavior of real 802.11p devices, both in line-of-sight conditions and when obstructed by small obstacles, and (ii) a visibility model able to deal with large obstacles, such as buildings and houses, in a realistic manner. Additionally, we proposed a model based on high-level performance indicators (bandwidth and delay) for LTE, which makes application validation and evaluation easier.

At the network layer, a hybrid protocol called AVE is proposed for packet forwarding by switching among a set of standard routing strategies. Depending on the specific scenario, AVE selects one out of four different routing solutions: a) two-hop direct delivery, b) Dynamic MANET On-demand (DYMO), c) greedy georouting, and d) store-carry-and-forward technique, to dynamically adapt its behavior to the specific situation.

At the transport layer, we proposed a content delivery protocol for reliable and bidirectional unicast communication in lossy links that improves content delivery in situations where the wireless network is the bottleneck. It has been designed, validated, optimized, and its performance has been analyzed in terms of throughput and resource efficiency.

Finally, at system level, we propose an edge-assisted computing model that allows reducing the response latency of several queries by placing a computing unit at the network edge. This way, traffic traversal through the Internet is avoided when not needed. This scheme could be used in both 802.11p and cellular networks, and in this thesis we decided to focus on its evaluation using LTE networks.

The platform presented in this thesis combines all the individual efforts to create a single efficient platform. This new environment could be used by any provider to improve the quality of the user experience obtainable through the proposed vehicular network-based services.

Resum

Els productes relacionats amb els sistemes intel·ligents de transport (ITS) s'estan transformant en una realitat en les nostres carreteres. Tots els fabricants de cotxes comencen a incloure accés a internet en els vehicles i a facilitarne la integració amb els telèfons mòbils, però en el futur més i més serveis s'hi introduiran. La connectivitat usant les xarxes vehicular esdevindrà la pedra angular de cada nova proposta, i oferir una qualitat de servei adequada serà, òbviament, desitjable. No obstant això, es necessita una gran quantitat de treball perquè les xarxes vehiculars oferisquen un rendiment similar al de les xarxes cablejades.

Les xarxes vehiculars queden definides per dues característiques bàsiques: alt dinamisme, ja que els nodes poden arribar a una velocitat relativa de més de 250 km/h; i heterogeneïtat, per la gran quantitat de propostes diferents que els fabricants estan llançant al mercat. Per això, per a fer possible el desplegament de serveis sobre aquestes xarxes, s'imposa la necessitat de fer un estudi en profunditat d'aquest entorn, i cal proposar i desenvolupar les eines adequades.

Aquesta tesi ataca la problemàtica del desplegament de serveis en aquestes xarxes a tres nivells diferents: (i) el nivell físic i d'enllaç , mostrant diverses anàlisis en profunditat del medi físic i models derivats per simular-lo; (ii) el nivell de xarxa, proposant un protocol de difusió de la informació per als paquets IP; i (iii) el nivell de transport, on es proposen altres protocols per a millorar el rendiment del transport de dades.

En primer lloc, s'han estudiat i modelat les dues principals tecnologies sense fils que s'utilitzen per a la comunicació en xarxes vehiculars, la branca d'estàndards 802.11, en concret 802.11p; i la comunicació cel·lular, en particular LTE. Atès que l'estàndard 802.11p és un estàndard bastant madur, ens centrem a crear (i) un model de propagació i atenuació capaç de replicar el rang de transmissió de dispositius 802.11p reals, en condicions de visió directa i obstrucció per petits obstacles, i (ii) un model de visibilitat capaç de simular

l'efecte de grans obstacles, com són els edificis, d'una manera realista. A més, proposem un model basat en indicadors de rendiment d'alt nivell (ample de banda i retard) per a LTE, que facilita la validació i l'avaluació d'aplicacions.

En el pla de xarxa, es proposa un protocol híbrid, anomenat AVE, per a l'encaminament i el reenviament de paquets usant un conjunt d'estratègies estàndard d'encaminament. Depenent de l'escenari , AVE tria entre quatre estratègies diferents: a) lliurament directe a dos salts, b) Dynamic MANET On-demand (DYMO) c) georouting voraç, i d) una tècnica store-carry-and-forward, per a adaptar-ne el comportament dinàmicament a cada situació.

En el pla de transport, es proposa un protocol bidireccional de distribució de continguts en canals amb pèrdues que millora el lliurament de continguts en situacions en què la xarxa és un coll de botella, com les xarxes sense fils. Ha sigut dissenyat, validat, optimitzat, i el seu rendiment ha sigut analitzat en termes de productivitat i eficiència en la utilització de recursos.

Finalment, a nivell de sistema, proposem un model de computació assistida que permet reduir la latència en la resposta a moltes consultes col·locant una unitat de computació a la vora de la xarxa, és a dir, la xarxa d'accés. Aquest esquema podria ser usat en xarxes basades en 802.11p i en xarxes cel·lulars, si bé en aquesta tesi decidim centrar-nos en la seua avaluació usant xarxes LTE.

La plataforma presentada en aquesta tesi combina tots els esforços individuals per a crear una plataforma única i eficient. Aquest nou entorn pot ser usat per qualsevol proveïdor per a millorar la qualitat de l'experiència d'usuari en els serveis desplegats sobre xarxes vehiculars.

Contents

1 Introduction **15**
1.1 Motivation 15
1.2 Objectives 16
1.3 Structure 18

2 Background **19**
2.1 Intelligent transportation systems 19
2.2 Vehicular networks 21
2.3 Wireless Local Area Networks 22
2.4 Wireless Metropolitan Area Networks 24
2.5 Raptor codes 27
2.6 Summary 28

3 Raptor-based Reliable Unicast Content Delivery in Wireless Network Environments **31**

4 Analysis and Evaluation of a Raptor-based Content Delivery Protocol and its High-performance Extensions **53**

5 Assessing the impact of obstacle modeling accuracy on IEEE 802.11p based message dissemination **83**

6 VACaMobil: VANET Car Mobility Manager for OMNeT++ **99**

7 An Adaptive Anycasting Solution for Crowd Sensing in Vehicular Environments **113**

8 Towards Enabling Hyper-Responsive Mobile Apps at Scale Using Edge Assistance **137**

9 Measurement-Based Modelling of LTE Performance in Dublin City **155**

10 Discussion **173**
10.1 General overview of contributions 173
10.2 Raptor-based Content Delivery Protocol (RCDP) 175
10.3 Adaptive Anycasting solution for Vehicular Environments (AVE) 178
10.4 Edge computing 179
10.5 Support tools 180
10.6 Summary 183

11 Conclusions **185**
11.1 Final remarks 185
11.2 Publications related with this thesis 186
11.3 Future work 189

Bibliography **191**

Acronyms glossary

- AVE - Anycasting solution for Vehicular Environments
- DSRC - Dedicated short-range communications
- DTN - Delay Tolerant Network
- DYMO - Dynamic Manet On-Demand
- EPC - Evolved Packet Core
- ETSI - European Telecommunications Standard Institute
- E-UTRAN - Evolved Terrestrial Radio Access Network
- FDMA - Frequency Division Multiple Access
- FEC - Forward Error Correction
- GPRS - General Packet Radio Service
- GTP-U - GPRS Tunneling Protocol for User Data
- HSS - Home Subscriber Server
- IEEE - Institute of Electrical and Electronics Engineers
- IoT - Internet of Things
- IP - Internet Protocol
- ISO - International Organization for Standardization
- ITS - Intelligent Transportation Systems
- LTE - Long Term Evolution

- MAC - Medium Access Control
- MANET - Mobile Ad-hoc Network
- MME - Mobility Management Entity
- NAS - Non-Access Stratum
- OBU - On Board Unit
- OFDM - Orthogonal Frequency-Division Multiplexing
- OFDMA - Orthogonal Frequency Division Multiple Access
- OLSR - Optimized Link State Routing
- PDCP - Packet Data Convergence Protocol
- PDN - Packet Data Network
- PGW - PDN Gateway
- RCDP - Raptor-based Content Delivery Protocol
- RLC - Radio Link Control
- RRC - Radio Resource Control
- RSU - Road Side Unit
- SC-FDMA - Single Carrier - Frequency Division Multiple Access
- SGW - Serving Gateway
- TCP - Transport Control Protocol
- UDP - User Datagram Protocol
- UE - User Entity
- V2I - Vehicle to Infrastructure
- V2V - Vehicle to Vehicle
- VACaMobil - VANET Car Mobility
- VANET - Vehicular Ad-hoc Network
- WAVE - Wireless Access in Vehicular Environments
- WLAN - Wireless Local Area Networks

Chapter 1

Introduction

1.1 Motivation

People are used to get frequent aid from technology in their daily lives. In the past, they only used computers at their homes or on other locations through desktop devices. Later, laptops allowed users to travel with their own computing devices. Nowadays, smartphones provide Internet access to users almost everywhere. Additionally, since networking and communication with other computers enhances the user experience, every incremental step was associated to a network improvement: wired networks for desktop computers, WLAN for laptops, and 3G/4G networks for smartphone-based data access. Nowadays, new steps will be taken to further integrate technology in our lives through novel paradigms such as the Internet of Things (IoT), Smart Cities and Intelligent Transportation Systems (ITS).

In this particular thesis, we focus on vehicular networks, the key component of ITS allowing vehicles to communicate between them and with the infrastructure. This kind of networks aims at interconnecting vehicles in order to feed its On Board Units (OBUs) with different kinds of data, and provide both drivers and passengers with new services. There are several kinds of services that are useful for drivers. The range goes from safety messages to infotainment, and from advertisements to social content distribution.

Nowadays, there are several companies that supply their vehicles with built-in vendor-specific devices to provide on-route services. Some of these solutions are BMW Connected Drive, Opel OnStar, Renault R-Link, Citroën Connect Box, Nissan Connect and Audi connect. Unfortunately, these solutions are

quite heterogeneous: some of them rely on cellular infrastructure (3G/4G), others use the widespread 802.11 standards, and yet others take advantage of the novel 802.11p standard (with or without WAVE and DSRC). Thus, applications for different technologies must be evaluated in the right environment and using the adequate protocol stack.

A second problem related to vehicular networks is performance. Current systems implement the common protocols used in wired and ad-hoc networks, whose performance is not as good as it should in vehicular networks. In particular, performance is downgraded because of the special characteristics of vehicular networks: fast node mobility, variable distances between nodes, access points are not always available, etc. Since services offered for vehicular networks are intended to be as good as the ones for desktop computers, and since some of them are even more time critical, new protocols must be designed to transmit information in an effective way.

In order to improve performance and reduce the problems created by heterogeneity, there are several issues that must be addressed. Since these issues concern many different tasks, they must be handled by the appropriate layer inside the vehicular network protocol stack. Therefore, in order to effectively improve performance of vehicular networks, research efforts must go through most of these layers to handle the targeted problems where required. This approach attempt to be the most adequate way to fix the problems found in vehicular networks, forcing us to distribute our research efforts throughout the entire vehicular network protocol stack.

1.2 Objectives

The main objective of this thesis is developing some of the necessary protocols and simulation tools that will help in the *evaluation* and *deployment* of services for vehicular networks. In this section we summarize the contributions of this thesis. These contributions are stated sequentially in this section, but they are classified in layers along the document for an easier analysis.

The first contribution of the thesis is the RCDP protocol. This content delivery protocol is able to use Raptor codes to create a loss-resilient channel. An adaptive control system is included to regulate the transmission rate. This protocol is able to significantly improve TCP performance on error-prone channels, like the ones in vehicular networks.

The second contribution of this thesis is a hybrid forwarding protocol that dynamically adapt its behavior by switching among four different routing approaches: the two-hop direct delivery, the Dynamic MANET On-demand

(DYMO), the greedy georouting, and the store-carry-and-forward technique. This protocol attempts to overcome the different drawbacks that each technique shows separately by wisely selecting the most suitable one or each situation. This is particularly important for vehicular networks because you can find very different situations along time, so the protocol must be able to switch to the best technique easily.

The third contribution of this thesis is a model of the LTE channel defined in terms of its main performance indicators (bandwidth, delay, packet losses). It is important because LTE is a commonly used communication technology in vehicular networks. This model was generated based on real measurements taken in Dublin (Ireland) and Valencia (Spain). These measurements were taken from 2 operators in each country in order to make comparisons. Several static placements were chosen to get samples from places with a different user density. A statistical model was derived from the measurements to allow performing channel simulations.

The fourth contribution of this thesis is the study of an edge-computing architecture to improve the performance of highly-responsive applications over LTE networks. This study finds out that a local server can be deployed at the PDN gateway to handle some of the requests sent to a central server, thereby improving response latency. Additionally, it shows a trade off between propagation delay from the Internet and queuing delay due to resource constraints at the edge server. Vehicular network performance can be highly improved with it, since they are networks with a big latency.

The fifth contribution of this thesis is a hybrid propagation model to simulate vehicular networks in the OMNeT++ simulator. This model in able to handle line-of-sight and non-line-of-sight transmissions by using the Nakagami model with a specific set of parameters, and a visibility model for signal fading due to buildings. The Nakagami model was tunned to show a similar behavior to measurements made with real 802.11p devices. The visibility model was taken from the works of Sommer et. al., and it can be used for simulating vehicular networks more accurately.

Additionally, the making of this thesis included the collaboration on the design and implementation of the VACaMobil tool. It is a mobility manager for OMNeT++ and SUMO that helps to carry out realistic simulations of vehicular networks with a controlled number of vehicles in any map. This tool allows the user to define the number of vehicles in the simulation, how this value will vary along time, and the RSU placement and distribution, among others.

1.3 Structure

In the first step, we provide the reader with some background in Chapter 2 to explain the context where the thesis was developed. We introduce the most relevant standards and technologies the work is based on.

From Chapter 3 to Chapter 9 we present the papers that are associated to this thesis. The main topics included in this section are our Raptor-based Content Delivery protocol (RCDP), our attenuation and visibility model, our adaptive Anycasting solution for Vehicular Environments (AVE), our VANET Car Mobility manager (VACaMobil), our system for edge computing, and our LTE lightweight model.

In Chapter 10 we give a brief overview of all the research carried out in our thesis, reviewing and discussing the whole set of results that we obtained. We specially highlight the role of each component in the general system that we present, and explain how the results improve the performance of the final system.

Finally, Chapter 11 concludes the thesis also summarizing the research contributions and discussing future works.

Chapter 2

Background

As stated in Section 1.1, there are many technologies that play an important role in the vehicular networks area and Intelligent Transportation Systems in general. This chapter focuses on the main protocols, systems and solutions that are related to this thesis.

2.1 Intelligent transportation systems

Creating a way to communicate vehicles between them and with other networks is a necessity that emerged a long time ago. It started as a primitive system that was only able to create unidirectional links between nodes. Later, Intelligent Transportation Systems (ITS) evolved until they enabled stations to establish peer-to-peer connections with any kind of node, such as road side units, other vehicles, etc.

Many factors shaped the current ITS situation. First, the different efforts of many different companies to introduce their own protocol stacks created a high heterogeneity. The second element that influenced the current ITS situation is the growth of smartphones in the customers market. It puts a device with multiple network interfaces in almost any vehicle, so developers can use different kinds of links in order to create a vehicular network. The third element is a joint work of several task forces to create a set of standards. Among these contributors we highlight the European Telecommunications Standard Institute (ETSI), and the International Organization for Standardization (ISO).

Figure 2.1 shows how an ITS protocol stack could be created by putting together all the pieces that we have now. A important thing is the duality

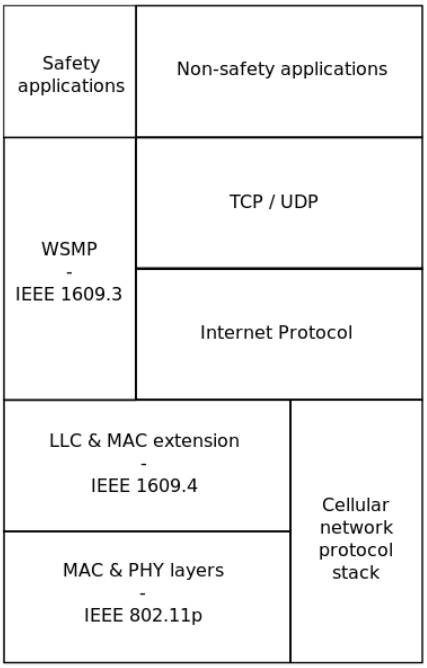


Figure 2.1: Layered architecture for Intelligent Transportation Systems.

between safety and non-safety applications. This difference motivated the standardization organizations to allow the use of two kinds of upper layer protocols: the general usage ones, and some specific ones in order to improve the performance of safety messages.

Vehicular networks open a wide range of communication scenarios. The first ones are the safety applications. In such scenarios, an alert is disseminated to as many vehicles as possible in order to let them know that an accident occurred, or to avoid future collisions by predicting them. The second one falls into the resource efficiency domain. If information regarding the traffic conditions is sent to the vehicles, they can be rerouted through better routes, avoiding unnecessary delays. The third one is infotainment. Entertainment is a field that is highly demanded by users. Therefore, including it into the ITS framework is an intelligent approach in order to offer the services that people seek.

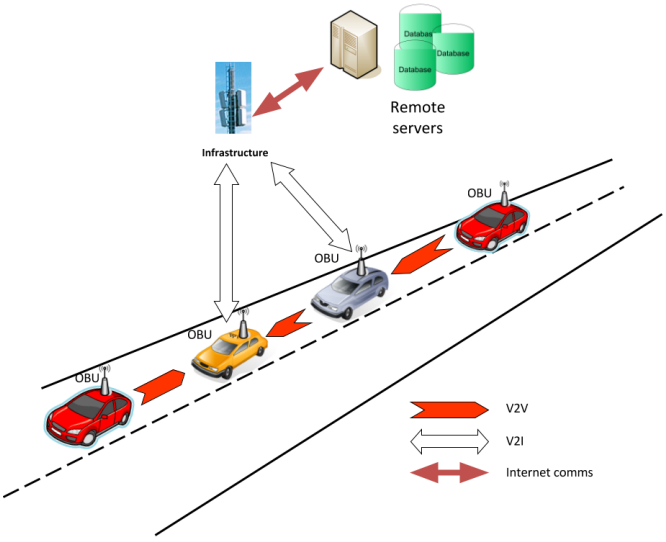


Figure 2.2: Example of V2V and V2I communications. © iNiT 2016.

2.2 Vehicular networks

Vehicular network is a generic term that comprehends all those kinds of networks when some of the network nodes are vehicles. Each vehicle usually has an On-Board Unit (OBU), which is a small computer that can act as a router, if needed. Since it is a very generic concept, it has lead to heterogeneous implementations. In this section, we will try to provide a global overview of the very different shapes that this concept can take in the real world.

A common implementation of vehicular networks is using standards of the 802.11 family (see Section 2.3). These standards provide the vehicles with local area connectivity, that can be used to connect to other nearby nodes. Depending on the presence of infrastructure elements, vehicular networks can be classified as Vehicle-to-Vehicle (V2V) or Vehicle-to-Infrastructure (V2I) networks. Both of them can be seen in Figure 2.2.

In V2V networks, the communication can be performed between vehicles only. In V2I networks, there are additional fixed nodes, called Road Side Units (RSU), that can be the source or destination of traffic, and that can be used as a link to bigger networks, such as the Internet, or to nodes that are far away.

The fact that vehicles are usually moving introduces important challenges in routing. The main difference is that, when vehicles have very variable mobility, topologic-based routing approaches (such as OLSR or DYMO) highly downgrade their performance. However, the geographic location of a node can be used as a heuristic for getting closer to the destination, i.e. if the next hop is geographically closer to the destination, it is probably topologically closer. Since vehicles can be equipped with positioning systems, such as GPS, geographic routing approaches were introduced in vehicular networks. Additionally, it is quite common that a node finds a hole in the network that prevents the packet destination from being reached. In order to solve this problem, protocols that are able to store and carry the packet for a while, until the destination or a better node is found, must be adopted. This technique, usually known as Delay Tolerant Networking (DTN), increases the packet delivery rate by increasing delay.

Introducing LTE into vehicular networks avoids the dependency on RSUs to reach external networks. This makes it easier for the vehicles to reach the service providers on the Internet. Unfortunately, easy tasks that could be performed faster by using direct node-to-node communications increase their latency by traversing the cellular access network. However, combining 802.11 and LTE interfaces can lead to more complex systems that can take advantage of both approaches to get a better performance for some tasks.

Finally, we want to highlight the differences between Vehicular Ad-hoc NETworks (VANET) and Mobile Ad-hoc NETworks (MANET). In both of them, networks are created ad-hoc, and nodes can move out and into the network. However, node mobility in VANETs is higher. Network topology can change more often. Additionally, nodes follow well known paths (roads), so their route can be traced. All of these characteristics are new, and make these networks a very different environment with their own challenges and risks.

2.3 Wireless Local Area Networks

The 802.11 standards family

The Institute of Electrical and Electronics Engineers (IEEE) started to create standards for Wireless Local Area Networks in 1996. The working group IEEE 802.11 was then created. The first standard was out in 1997, and it got its first revision in 1999. Since then, a series of task groups were created to handle different topics:

- 802.11a/b/g/n/ac/ad: Increasing the speed rates and Radio Frequency

	Year	Frequency band	Max speed
802.11	1997	2.4GHz	2 Mbps
802.11a	1999	5GHz (300 MHz)	54 Mbps
802.11b	1999	2.4GHz	11 Mbps
802.11g	2003	2.4GHz	54 Mbps
802.11n	2009	2.4 GHz / 5 GHz	540 Mbps
802.11ac	2014	5 GHz	6,240 Mbps
802.11ad	2009	60 GHz	6,756.75 Mbps

Table 2.1: 802.11 family summary

optimization.

- 802.11e: Quality of Service issues
- 802.11i: Security
- 802.11c/f: Interconnectivity and mobility protocols.
- 802.11d/h: Regulations.
- 802.11p: ITS connectivity

The annexes to the standard that advance faster and attract more customer attention are those focused on increasing the speed rates. Table 2.1 shows a summary of the performance improvements that these standard annexes brought.

The standard annex that is most related to this thesis' topic is 802.11p. As the rest of the standards in the 802.11 family, it defines the protocols to interconnect devices at the physical and MAC layer. The allocated spectrum for Intelligent Transportation Systems is in the 5.9 GHz band. There are minor variations depending on the country, but we will focus on the US case as our example, and we will later discuss the differences towards some proposals for Europe. In the US case, 75 MHz (from 5.850 GHz to 5.925 GHz) are allocated, i.e. seven 10MHz channels and a 5MHz guard band. The central channel is chosen as the control channel (used for alert dissemination and service advertisement), and the remaining channels are used as service channels. The minimum number of radio devices is one, meaning that a protocol for channel switching will be required in those cases where the number of radio interfaces is lower than the number of channels.

The maximum allowed transmission power is also regulated. Devices are classified in different classes, as occurs for cellular communications, with different maximum transmission power. They go from class A, with 0 dBm of transmission power and a range of 15 meters, to class D, with 28.8 dBm of transmission power and a range of 1 Km.

In Europe, some alternative deployments are taken into account. An alternative includes the allocation of just 30 MHZ of the spectrum, divided in just two channels (service and control). Additionally, the use of devices with two antennas is getting more relevance than in the US. However, there are many different organizations (ETSI, OSI, the EU itself) that are now working on these standards.

Orthogonal Frequency Division Multiplexing (OFDM) was set as the Physical layer protocol for each 10 MHz channel. Additionally, four modulation techniques are allowed and Forward Error Correction (FEC) codes are also included in three different codification rates. The combination of all these elements allows a maximum data rate of 27 Mbps.

The main change at the MAC layer in this release of the protocol is that a node does not need to belong to a service set (BSS, EBSS, etc) to communicate with a peer. In this case, a new type of communication, the *outside of the context of a BSS* (OCB) communication, is allowed. This mode is intended to reduce the delay in communication by removing the necessity of association between stations. It has several new features:

- A station cannot engage in OCB communication while it belongs to a BSS.
- The beacons that announce a BSS are not issued in this mode.
- No synchronization is required before stations can communicate.
- No authentication is carried out in the MAC sublayer.
- No association is required between stations before communication starts.

All the features that have been removed, must now be handled by upper-layer protocols.

2.4 Wireless Metropolitan Area Networks

Fourth-generation cellular networks

Figure 2.3 shows the main components of an access network of a 4G/LTE cellular network, which is named Evolved Packet Core. The connection between

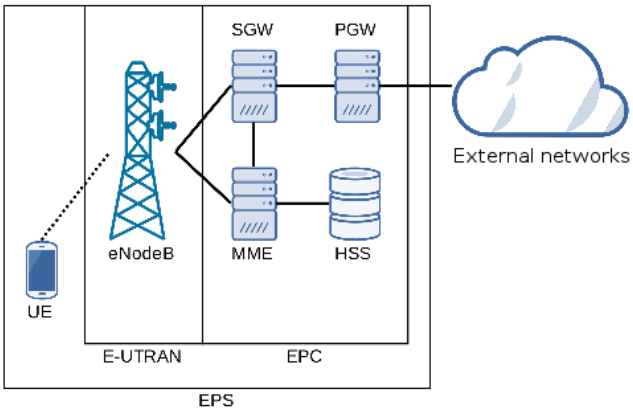
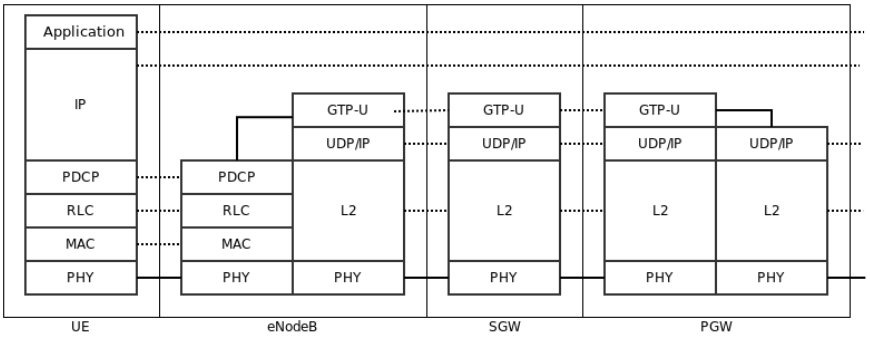


Figure 2.3: 4G/LTE cellular network

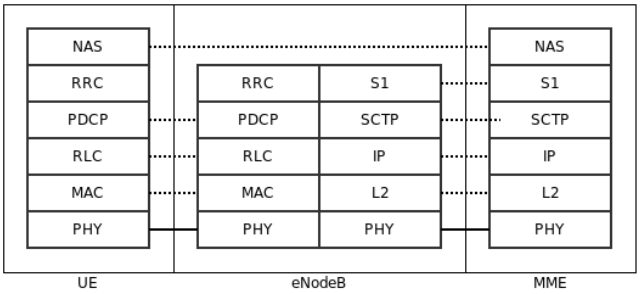
the clients, also known as User Entities (UE), and the network is achieved through the E-UTRAN devices. The E-UTRAN is a set of base stations (eNodeB), which are usually connected between them. This is the first step for a user to reach the external networks, i.e. the Internet, through a cellular network. The base stations inside the same eNodeB are interconnected in order to improve performance by handling some events as soon as possible. However, much information must be updated in control nodes, so they cannot operate in a totally independent fashion.

Communication between UEs and E-UTRANs is based on frequency division (FDMA). However, uplink and downlink use slightly different approaches. Downlink uses Orthogonal Frequency Division Multiple Access (OFDMA). This technology would have made user devices too expensive, so uplink implements a Single Carrier - Frequency Division Multiple Access (SC-FDMA). This diversity makes an asymmetric performance in uplink and downlink, which must be taken into account for evaluating and modeling this kind of networks.

The next big set of devices is the Evolved Packet Core (EPC). This group is composed by the Serving Gateway (SGW), the PDN Gateway (PGW), the Mobility Management Entity (MME) and the Home Subscriber Server (HSS). The SGW and the PGW are gateways that handle user traffic to and from the access network. The MME manages mobility and security signals generated in the network. The HHS is a database that keeps information of users and



(a) User data plane



(b) Control data plane

Figure 2.4: LTE software architecture

subscribers, and methods to retrieve and change it.

Figure 2.4 shows the software architecture implemented in those networks. There are three layers in the protocol stack that are common for control and data: the Packet Data Convergence Protocol (PDCP), the Radio Link Control (RLC) and the Medium Access Layer (MAC). The PDCP handles header compression, security, packet reordering and retransmission during handover. The RLC handles packet segmentation and assembly to adapt the packet size to the interface maximum transmission size. Finally, the MAC handles medium access and data multiplexing. This layer is special because of the scheduler, which handles the radio resource allocation.

In the data plane, IP and UDP packets are generated, but they are usually

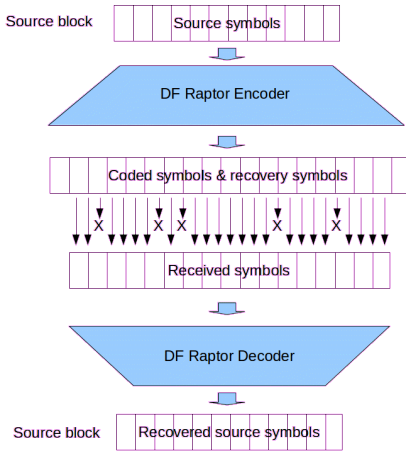


Figure 2.5: Raptor codes creation and recovery procedure.

encapsulated into the GPRS Tunneling Protocol for User Data (GTP-U). Each data connection is linked to a bearer, and it is identified by a Connection ID. Additionally, each endpoint is identified with its own Terminal ID. These identifiers are used for the tunneling process. In the control plane, the Non-Access Stratum (NAS) and the Radio Resource Control (RRC) protocols are implemented. NAS manages the EPC from the highest level (bearer creation and destruction, authentication, etc.), and RRC is a transport protocol for these messages.

There are two main issues that make it difficult to study such network from the outside: (i) the GTP-U protocol and (ii) the scheduler. Since the GTP-U protocol is a tunneling protocol, it hides information related to the nodes that are in the access cellular network. That makes it impossible for the user to know the components within the access network, and so the delay introduced by each of them. The scheduler, which manages the packet queues in the base station, is defined by the operator. Therefore, the behavior may be different from one operator (or even from one deployment) to others.

2.5 Raptor codes

Raptor Codes are Forward Error Correction (FEC) codes that operate at the application layer, and that allow recovering the transmitted information even when some of the packets are lost. FEC codes add redundant information to

the transmitted data to allow the receiver, through mathematical transformations, to recover the whole message sent. This coding system starts from a finite piece of information called a block, which is divided into smaller pieces of information called source symbols. Then, starting from these source symbols, recovery symbols are generated. Recovery symbols add the redundant information required to mitigate information loss, making the recovery process independent of which symbols are actually lost. In fact, the only requirement is that the amount of symbols received must be equal to, or slightly higher than, the original amount.

Raptor symbols are encoded in two phases: first, a pre-coding phase creates pre-coded symbols through linear combinations of original symbols; afterward, the coding phase creates a near-unlimited number of recovery symbols to be sent to the receiver. Coded symbols also result of linear combinations, in this case of combinations of pre-coded symbols.

The decoding process is just the opposite: received symbols, independently on whether they are source or recovery symbols, are combined to generate pre-decoded symbols; afterward, these pre-decoded symbols are also combined to obtain the source symbols.

The whole process can be seen in Figure 2.5.

2.6 Summary

Vehicular networks are growing at a steady pace. A connected car is a valuable element for the user, which drives companies into including connectivity features in their new models. This user interest for connected cars is also pushing governments to install Road Side Units at the roads, and to fill every cellular coverage gap in the roads.

However, communication in this environment is still quite challenging: the channel quickly degrades, different types of obstacles block signals quite often, and mobility makes it difficult for relayed packets to reach their destination efficiently. Thus, Research is still in progress to solve all these issues.

Nowadays, there are two main trends that the companies are following. The first one is using cellular-based (4/5G) devices to offer connectivity. The main advantage of this approach is that all the required infrastructure is already there, and so connectivity is possible at this time. The second one is based on the IEEE 802.11p standard. The performance achieved by this standard is expected to be better than the performance of the previous one. However, the car density and the available infrastructure are far meeting basic requirements.

Finally, Forward Error Correction techniques, like Raptor Codes, shield

the channel to improve performance on these networks. Using them, TCP can be avoided and data delivery can be substantially accelerated.

Chapter 3

Raptor-based Reliable Unicast Content Delivery in Wireless Network Environments

Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. *Raptor-based reliable unicast content delivery in wireless network environments.* In Chun Tung Chou, Tom Pfeifer, and Anura P. Jayasumana, editors, IEEE 36th Conference on Local Computer Networks, LCN 2011, Bonn, Germany, October 4-7, 2011 , pages 841-849. IEEE Computer Society, 2011.

Abstract

Delivering contents over wireless networks can be a challenging task due to the intrinsic limitations of these environments. In this chapter we propose RCDP, a solution which adopts an application layer FEC scheme based on Raptor codes to avoid retransmissions, optimizing content delivery for the unicast case. The proposed solution relies on end-to-end bandwidth estimations to perform rate control, achieving high throughput levels in lossy wireless environments. We have designed and implemented RCDP for the GNU/Linux platform to validate our approach under different channel conditions, varying packet loss

ratio, end-to-end delay, and channel capacity. Experimental results show that the proposed solution is very efficient when facing high loss ratios and end-to-end delays, allowing for an efficient usage of channel resources in wireless scenarios

3.1 Introduction

Reliable delivery of data over unreliable and bandwidth-constraint networks is a challenging task. When attempting to deliver any kind of content over wireless networks, the list of problems faced in traditional networks - i.e. lack of QoS support, losses and delay caused by congestion - is extended to include new problems specific of wireless environments, such as errors associated with low signal levels, fading signal and multipath interference effects, interference caused by other wireless networks or even contention among nearby devices. For this reason, the adoption of new mechanisms to mitigate the effect of losses becomes of utmost importance.

Achieving application layer reliability in an efficient manner is an issue that has recently drawn much attention from the research community [1]. In particular, researchers attempt to solve the packet-loss problem without resorting to retransmission protocols because such strategy generally does not scale well in large deployments, and the delay introduced by retransmission adversely affects performance. In wireless environments, retransmission-based protocols face additional problems [2, 3] since they must differentiate between congestion and channel losses. In the particular case of wireless multi-hop environments, such task becomes even more difficult [4, 5].

Several of the previously mentioned drawbacks can be overcome by the use of Forward Error Correction (FEC) schemes at the application layer, a technique also known as AL-FEC. In contrast to link and physical layer FEC, AL-FEC operates transparently above the IP layer, and is applied end-to-end to particular application flows.

Up to now, most research in this area has focused mostly on IP-based content broadcasting, where the goal was to distribute location-dependent information in highly populated areas such as airports and sports stadiums. The most effective schemes currently available [6] achieve this goal by combining the use of AL-FEC schemes, such as Raptor Codes [7], with the File Delivery over Unidirectional Transport (FLUTE) protocol [8]. Although such solution is very efficient for multicast/broadcast based content delivery to high numbers of users, it is not appropriate for unicast content delivery since: i) there is no mechanism defined for communication between transmitter and receiver

to provide feedback about performance, or to signal the end of the reception; ii) there is no mechanism to perform flow control; and iii) there is no rate control regulation scheme to adapt the transmission rate to variable network conditions. Additionally, when the content's size becomes very large, the adoption of FLUTE can incur in excessive memory requirements, meaning that a simpler and more efficient alternative to FLUTE should be adopted.

In this chapter we propose a novel content delivery scheme for wireless environments that targets reliable unicast content delivery. Our solution uses end-to-end bandwidth estimation techniques to perform rate control, and adopts an AL-FEC solution based on Raptor Codes [7] to avoid packet retransmissions. The proposed strategy performs flow control by acknowledging blocks of data instead of individual packets, which significantly reduces bandwidth requirements in the return path. Experimental testbed results validate the effectiveness of the proposed solution, showing error resilience levels significantly higher than the standard TCP-based solutions.

This chapter is organized as follows: in the next section we present an overview of some related works. Section 3.3 briefly introduces the Raptor codes technology. The core of the chapter is section 3.4, which introduces our Raptor-based Content Delivery Protocol (RCDP), including both architectural and implementation details. Performance results are then presented in section 3.6. Finally, section 3.7 concludes the chapter.

3.2 Background

In the literature we can find different solutions that attempt to improve content delivery performance in wireless environments. Among these we have: link-layer solutions, split- connection solutions, TCP-enhancements, MANET-specific proposals and FEC-based solutions.

In terms of link layer solutions, the AIRMAIL protocol [9] combines both retransmission and error correction to improve performance; the snoop protocol [10] relies on an agent to detect channel losses; additionally, Tulip [11] enforces retransmission acceleration at the MAC level.

With respect to split-connection approaches, which divide each TCP connection into two separated connections, Mobile TCP [12] has a three-layered structure which routes, reconnects and controls the transmission rate; Wireless-TCP [13] adopts a different approach, avoiding the use of a window-based flow control.

Concerning those solutions that enhance the original TCP implementation, TCP SACK [14] informs the sender node about packet loss providing more

details than TCP; SMART [15] combines the Go-Back-N approach and the selective ACK; Caceres and Iftode [16] propose a fast retransmission solution focused on mobile communications.

In terms of MANET-specific proposals, TCP-F [4] uses RFN and RNN packets to stop and start packet transmission, while Ad-hoc TCP [5] defines states in the sender; both are an example of research specifically focused on improving TCP performance in MANETs.

Finally, focusing on FEC-based solutions, Luby et al. [6] propose a solution for reliable file delivery over mobile broadcast networks, concentrating on Raptor codes for Multimedia Broadcast and Multicast Services (MBMS) within the scope of the 3GPP specification. Authors emphasize on the goodness of the solution achieved, and consider that Raptor codes are applicable to other scenarios such as video broadcasting over the Internet and peer-to-peer distribution. Overall, authors predict that, with the availability of powerful and low-complexity Raptor codes, many innovative applications and services are enabled in a very efficient and reliable manner. In this chapter we adopt these guidelines, although relying on Raptor codes for unicast content delivery instead.

Our proposal differs from the previous solutions by addressing reliable two-way communications following a completely novel approach. In particular, our solution is based on a novel transport protocol which relies on Forward Error Correction to completely avoid retransmission, along with an end-to-end bandwidth estimation technique to perform rate control. The solution we offer avoids that intermediate nodes participate actively in the process, as well as introducing any hardware changes. In terms of implementation, no windowing or re- transmission control has to be performed, which simplifies the tasks on both transmitter and receiver sides. To the best of our knowledge, no similar solution was proposed in the literature offering efficient and robust content delivery while supporting reliable and bidirectional communications.

3.3 Raptor Codes Overview

Raptor Codes [7] are Forward Error Correction (FEC) codes that operate at the application layer, and that allow recovering the transmitted information even when some of the packets are lost. FEC codes add redundant information to the transmitted data to allow the receiver, through mathematical transformations, to recover the whole message sent. This coding system starts from a finite piece of information called a block, which is divided into smaller pieces of information called *source symbols*. Then, starting from these source sym-

bols, recovery symbols are generated. Recovery symbols add the redundant information required to mitigate information loss, making the recovery process independent of which symbols are actually lost. In fact, the only requirement is that the amount of symbols received must be equal to, or slightly higher than the original amount.

Raptor symbols are encoded in two phases: first, a pre-coding phase creates pre-coded symbols through linear combinations of original symbols; afterwards, the coding phase creates a near-unlimited number of recovery symbols to be sent to the receiver. Coded symbols also result of linear combinations, in this case of combinations of pre-coded symbols.

The decoding process is just the opposite: received symbols, independently on whether they are source or recovery symbols, are combined to generate pre-decoded symbols; afterwards, these pre-decoded symbols are also combined to obtain the source symbols.

Finally, we must distinguish between two types of codes: systematic codes and non-systematic codes. A code is systematic when source symbols are always reproduced among output symbols, and it is non-systematic if source symbols are not necessarily reproduced among the output ones. For our solution we adopted a systematic version of Raptor Codes, thus avoiding wasting time in the decoding process when no losses occur.

3.4 Our Raptor-based Content Delivery Protocol (RCDP)

Nowadays, the most widely adopted content delivery strategy over unicast is based on the well known HTTP protocol [17], which operates at the application layer. This protocol must be combined with TCP at the transport layer to achieve the goal of providing a reliable end-to-end content delivery. When attempting to deliver contents over wireless networks, the HTTP/TCP tandem suffers from low performance since TCP is unable to distinguish whether the packet losses detected are due to network congestion or to channel-related problems. Thus, efficient content delivery solutions should be sought to optimize performance in wireless environments.

To achieve this goal, we introduce our novel Raptor-based Content Delivery Protocol (RCDP). The basic idea of RCDP is to combine an AL-FEC strategy with the UDP protocol to create a content delivery solution which is nearly immune to packet losses. In particular, RCDP will rely on Raptor codes to recover the original data even when part of the information is lost in-transit.

This behavior enabled by the Raptor encoding scheme, which allows ignoring packet losses. Therefore, the source information is recovered only through newly incoming packets, not requiring any information to be retransmitted by the sender. This characteristic is common to all fountain codes [18], being Raptor codes a particularly efficient FEC scheme within this group.

RCDP uses an end-to-end bandwidth measurement strategy to determine the available bandwidth on the channel. This strategy allows the receiver to determine the channel's current bandwidth simply by measuring the time between consecutive packet arrivals. The receiver can then send this information back to the sender, allowing it to adjust its sending rate to the most appropriate value in order to maximize throughput. Notice that this strategy is not applicable to any broadcast/multicast based content delivery scheme previously proposed [6].

We split the implementation of RCDP into four sublayers, where at the top we have the application interface sublayer, below which we have the Raptor sublayer followed by the end-to-end management sublayer; at the bottom we have the channel communications sublayer. The purpose of the application interface sublayer is to offer the developer the typical sockets interface. The Raptor sublayer offers Raptor encoding/decoding of blocks of data. The end-to-end management sublayer deals with protocol control tasks such as flow control. Finally, the channel communications sublayer interfaces with UDP and abstracts all the operations related to network communication.

We now detail the most relevant design and implementation issues.

3.4.1 Raptor coding and decoding process

In the RCDP architecture, the Raptor sublayer encodes the information received by the application interface sublayer at the sender side, handing packets over to the end-to-end management sublayer buffers. The Raptor sublayer will also be responsible for decoding the encoded information received by the management sublayer at the receiver, handling it to the application sublayer accordingly.

The Raptor sublayer introduces a packet header that consists of four integer fields: (i) the number of source symbols in the original message, (ii) the first symbol identifier, (iii) the block identifier, and (iv) the block size. This header will be appended to both source and recovery symbols to create a packet.

The sender, as described in algorithm 3.1, starts by retrieving the information to be sent from the application interface sublayer. It then splits this information into one or more source blocks. Each source block is subdivided into pieces called source symbols. We add the header previously described to

Algorithm 3.1 Raptor data processing at the sender.

1. **While** (there is information to send) **do**
   (a) Read next data block from upper sublayer
   (b) Split data block into source symbols
   (c) **For** (all symbols created) **do**
      i. Packetize symbol
      ii. Send packet
   (d) Perform Raptor encoding of data block
   (e) **While not** (received *successfully recovered block* message) **do**
      i. Generate recovery symbol
      ii. Packetize recovery symbol
      iii. Send packet

each symbol to create an RCDP packet, which will be handled by the sublayer below. Notice that, since we rely on systematic Raptor codes, source symbols match the original information block, meaning that they can be sent immediately, without waiting for the encoding process to complete. In fact, the encoding process is started in parallel to maximize performance. The Raptor encoding output includes both source and recovery symbols; the latter are also packed and handled o the end-to-end management sublayer for delivery. The latter task goes on uninterruptedly until a *successfully recovered block* notification is received. The sender repeats this process with the following block until the information flow from the top sublayer ends, or the connection is lost.

With regard to the receiver, it takes the sequence of steps described in algorithm 3.2. Thus, it is continually waiting to receive the symbols of a block (both source and recovery symbols), which are stored in memory. When it has enough symbols to recover the source block, it proceeds with the recovery process and sends a *successfully recovered block* notification back to the sender; such notification also serves flow control purposes. The recovered block is then handed over to the top sublayer, and this entity goes back to the symbol reception state. Notice that, in case the channel is lossless, successful decoding takes place immediately after all the source symbols are received, meaning that recovery symbols are not necessary. In those situations the Raptor-related delays are reduced to a minimum.

Algorithm 3.2 Raptor data processing at the receiver.

1. **While** (information is coming) **do**
   (a) Receive symbol
   (b) **If** (belongs to current block) **do**
      i. Store it in memory
      ii. **If** (received enough symbols to recover the block) **do**
         A. Recover the block
         B. Handle it to the upper sublayer
         C. Generate *successfully recovered block* message

3.4.2 The rate control scheme

In wireless networks, channel bandwidth is continuously changing due to variable link quality, variable congestion states, or even variable paths. Therefore, we have to create a rate control system that can easily adapt to highly variable network states.

The rate control algorithm we propose is based on rate-control, and it relies on channel bandwidth estimations made by the receiver. These estimations are calculated based on packet arrival patterns, and are returned to the sender as soon as they are obtained.

Following the strategy we proposed in a previous work [19], the sender will operate by grouping packets in packet trains, and sending them at a speed higher than the one estimated at the receiver. This way, when there is more bandwidth on the channel than the one estimated, packets arrive at the receiver faster than expected. In such cases the receiver sends back a new bandwidth estimation reporting that transmission conditions have improved, which allows the sender to increase the sending rate to take advantage of that situation. Otherwise, if channel conditions become worse, packets will arrive at a rate lower than expected because of the higher delays experienced. Likewise, the receiver sends back a report so that the sender proceeds to decrease the delivery rate.

To implement this idea we have devised the following algorithm: the sender initially receives the bandwidth reports (C_i) and applies them a correction parameter (β), as shown in equation 3.1, to obtain a target data rate R_i. Parameter β varies between 0 and 1, and its purpose is to slightly reduce the target data rate to avoid saturating the channel, thus offering some room for

best-effort traffic. The target data rate will be the one we expect to measure at the receiver side.

A correction factor (α) allows obtaining the train rate (Ω_i) from the target data rate (see equation 3.3), where α is a value between 0 and 1. The train rate will be the actual rate used to send the packets which are part of a burst.

$$R_i = \beta \times C_i \tag{3.1}$$

$$\Omega_i = \frac{1}{\alpha} \times R_i \tag{3.2}$$

When the packet burst is sent, it is followed by a pause period so that the data rate over one period (T_k) matches with the target data rate (R_i), which is the data rate we expect to find in the channel. Notice that T_k is calculated based on the target data rate, the number of packets in a burst (N) and the packet size expressed in bytes (P_{size}) as follows:

$$T_k = \frac{8 \times N \times P_{size}}{R_i} \tag{3.3}$$

Notice that the Raptor encoder generates symbols of a same size, which must be initially defined. This means that making P_{size} a constant value becomes a trivial matter. In our solution, P_{size} is optimized according to the layer-2 MTU, similarly to the approach followed by TCP.

3.4.3 Implementation details

We have implemented our RCDP solution following the architecture shown in figure 3.1. As shown, we adopted a multi-threaded approach where different modules are combined to achieve an efficient and robust solution.

To accelerate the development of the proposed RCDP protocol we relied on the UDT library [20], which is a communications library written in C/C++ that is available for Linux, Solaris and Windows. This library offers all the features required to implement RCDP, including socket creation and configuration for communication with applications, connection startup and closing, information delivery and reception, etc. Thus, we took advantage of the basic pillars of the UDT library as a starting point to develop RCDP.

Concerning the Raptor modules, they were developed using the libraries provided by Digital Fountain Inc.[1], released under an academic research agreement. In particular, we relied on version 11 of the Raptor libraries for Linux to perform coding/decoding tasks.

[1] Licensed by Qualcomm Inc.

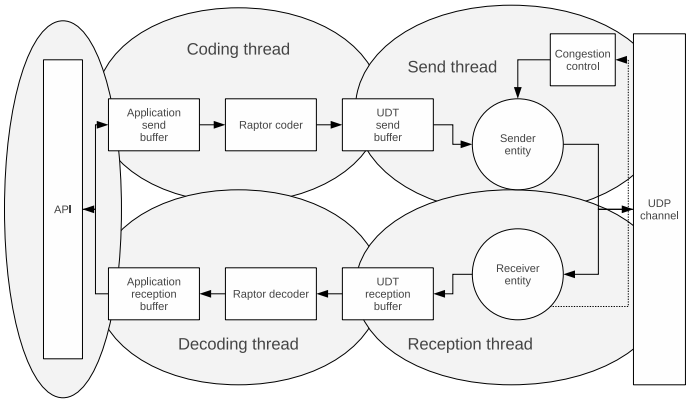


Figure 3.1: Overview of the RCDP architecture.

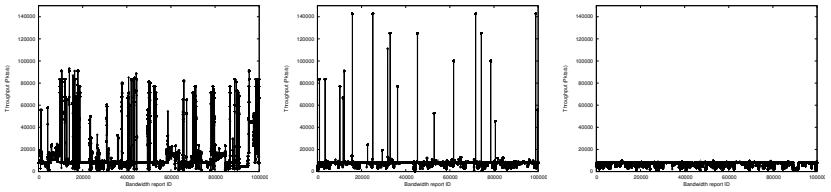
The Raptor Coding Library offers the programmer an interface similar to other coding libraries, allowing us to encapsulate all the functionality related to Raptor coding and decoding in an independent sublayer by creating a pair of C classes, coder and decoder, that are totally independent of the other classes in our architecture.

The Raptor coder and decoder classes constitute a sublayer between the application interface sublayer and the management sublayer. As shown in figure 3.1, they are created as independent threads which communicate with other sublayers through a pair of buffers to provide and retrieve information.

Notice that all these modules are executed at the user level, being the interaction with the kernel limited to the UDP exchanges.

3.5 Improvements to the bandwidth estimation process

Achieving accurate bandwidth measurements is of utmost importance for the rate control scheme to be successful. In this section we describe the steps taken and the different strategies evaluated when improving bandwidth estimation accuracy at both sender and receiver sides.

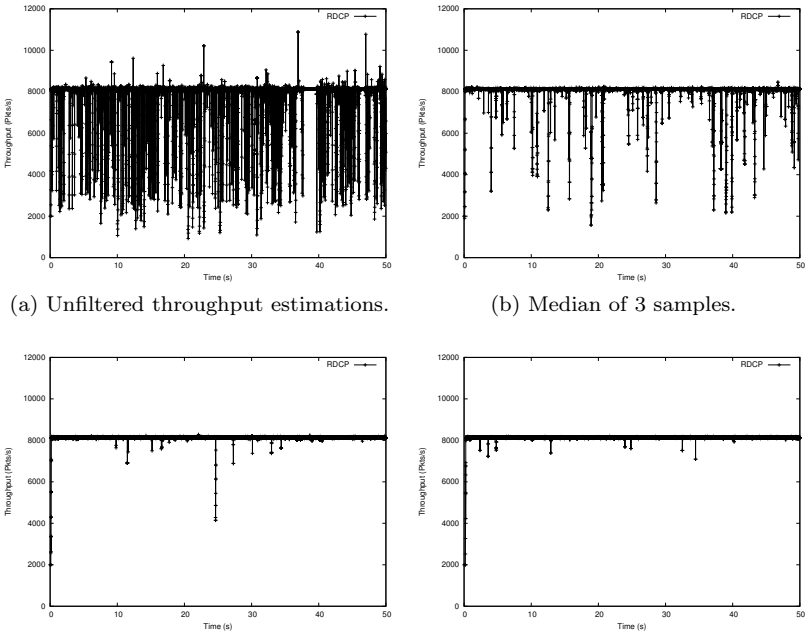


(a) Strategy #1 (b) Strategy #2 (c) Strategy #3

Figure 3.2: Bandwidth report values (C_i) when using different measurement strategies.

3.5.1 Bandwidth estimation accuracy at the receiver

To study the accuracy of the bandwidth estimation process at the receiver we connected two terminals using a Fast Ethernet link and measured the estimated throughput values. When using packets of maximum size, such channel supports about 8000 packets per second. However, when analyzing the bandwidth report values returned, we found that the values measured following the methodology presented in section IV-B were far from this reference value, being often much higher (see figure 3.2a). We found that such high variability was due to the inter-packet measurement strategy. Our initial measurement strategy (strategy #1) relied on retrieving the reception timestamp of each packet just after it was recovered from the UDP buffer in the kernel. Since variable buffering times were expected to cause unpredictable delays, we considered that this first strategy was not adequate. Thus, our second strategy relied on kernel reception timestamps instead. This improved the inter-packet measurement accuracy by substantially reducing the number of peaks, as can be observed in figure 3.2b. Still, several peaks remained where the estimated throughput was higher than the maximum theoretical value. We found that these remaining peaks were associated with situations were two factors were combined: i) the network interface presented a bursty behavior, introducing very small inter-packet values for certain packet pairs, and ii) bandwidth reports were returned earlier than expected, and considered only two or three packets in the bandwidth estimation calculations. By forcing that at least half the packets in a train had to arrive before a bandwidth report was returned (strategy #3), we achieved very significant improvements (see figure 3.2c), with overestimations taking place less than once every 260.000 times. Such rare cases can be easily filtered at the sender, as shown next.



(a) Unfiltered throughput estimations.

(b) Median of 3 samples.

(c) Median of 5 samples.

(d) Median of 7 samples.

Figure 3.3: Throughput estimations made at the receiver using different degrees of filtering.

3.5.2 Filtering strategies at the sender

Despite enforcing the bandwidth estimation optimizations referred above for the receiver, the sender must also introduce additional filtering strategies to avoid, as shown in figure 3.3a, overestimating or underestimation bandwidth values. In particular, the underestimation problem is of more concern to the sender as can be observed. To address this problem we tested with two different unidimensional smoothing filters: a median filter and an exponential filter.

A median filter including a window of k elements can be expressed as follows:

$$M_n(k) = median(C_n, C_{n-1}, ..., C_{n-k+1}), \quad n > k-1 \tag{3.4}$$

It means that the bandwidth estimation used by the source will be the

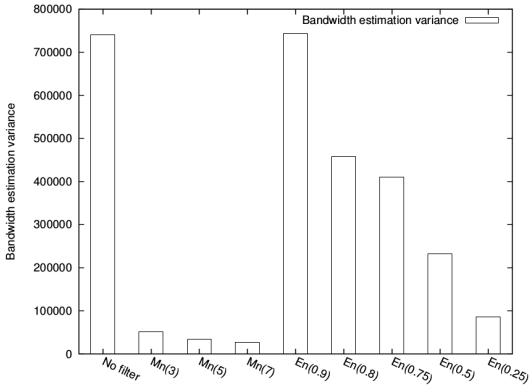


Figure 3.4: Bandwidth estimation variance for different filtering strategies.

median value of the set of measurements including the current bandwidth report (C_n) and the $k-1$ previous reports. In our experiments we vary the median filter window size (k) to achieve different degrees of smoothing.

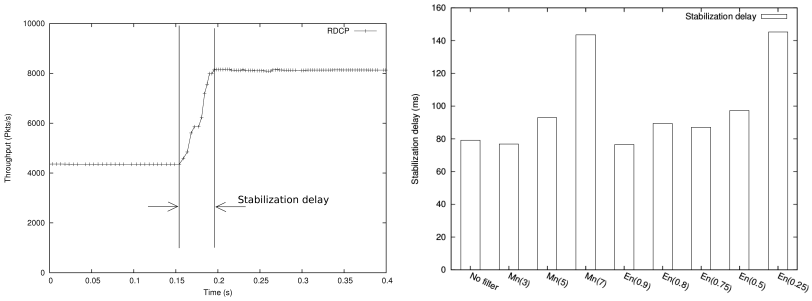
Concerning the exponential filter, it can be expressed as follows:

$$E_n(\gamma) = \gamma \cdot C_n + (1-\gamma) \cdot E_{n-1}, \quad n > 1 \tag{3.5}$$

where C_n is the nth bandwidth report returned by the receiver, E_{n-1} is the previous smoothed bandwidth value, and E_n is the new smoothed bandwidth value. In our experiments we tested different values for the γ parameter to achieve different degrees of smoothing.

Figure 3.4 shows the statistical variance associated with bandwidth estimation when applying no filtering, and when applying either median and exponential smoothing filters with different values of k and γ, respectively. These results show that median filters are much more effective than exponential filters, even for a set of values as low as $k = 3$.

In figure 3.3 we show the results of the median filtering process in visual terms. We can clearly see that, despite the differences in terms of variance are not too significant, median filters using $k = 5$ or $k = 7$ are quite more efficient at removing unwanted values than using the median of just 3 samples. In detail, we find that unfiltered estimations showing very low or very high values occur about 13.15% of the times. This value goes down to 3.48% when



(a) Illustration of stabilization delay upon a channel capacity shift.

(b) Comparison between different filtering strategies.

Figure 3.5: Stabilization delay

introducing a median filter with $k = 3$. For $k = 5$ and $k = 7$ such peaks become residual, with values as low as 0.54% and 0.44%, respectively.

Although the aforementioned results indicate that using median filters with a lot of samples if the best choice in terms of steady-state accuracy, the end-to-end capacity of a connection is prone to experience frequent fluctuations due to signal-to-noise ratio fluctuations, route changes and variable congestion states. Thus, besides aiming at high accuracy in the bandwidth estimations, the proposed solution should also be able to adapt quickly to sudden capacity changes. In figure 3.5a we introduce the concept of stabilization delay, which is the time elapsed from the instant the capacity of the channel changes to the instant when the rate control element achieves that new value. To optimize network resources, this delay should be as low as possible.

Figure 3.5b shows the average stabilization delay values achieved by the different filtering strategies evaluated. We find that excessive smoothing has a negative impact on delay for both filter types. In particular, median filters with $k = 7$ or greater introduce an excessive delay, as well as exponential filters with $\gamma = 0.25$ or lower. Overall, we consider that the best strategy in terms of bandwidth estimation accuracy and stabilization delay is achieved by using a median filter with $k = 5$. Thus, we will adopt such filtering approach for the experiments that follow.

3.6 Performance evaluation

We now proceed to validate the proposed RCDP protocol, assessing its performance under different channel conditions. For comparison purposes, we also present performance results obtained with the most widely used content delivery solution in the Internet, which is based on the combination of HTTP and TCP.

We devise a set of tests where we compared both solutions under the same conditions of error, delay and channel bandwidth. Since our experiments were made with real working software for the GNU/Linux platform, and to make sure that the test sequences were reproducible, we created a controlled environment for testing where we emulated different channel conditions. With this purpose we created a network black box that emulates a configurable point-to-point connection. The proposed network black box was initially validated to make sure that test results were reliable, and that the comparisons made through it were fair.

For evaluation we created a very simple content provider (server) and content requester (client) that can operate with any of the two content delivery solutions under evaluation: RCDP over UDP, or HTTP over TCP. We emulated the channel conditions of different wireless network environments by defining different values for delay, packet loss ratio and channel bandwidth. In our tests the client requests a large-sized content from the server, and calculates the productivity (in Mbps) associated with the transmission of that content. For each configuration we averaged five samples to obtain each point represented. Concerning both α and β parameters, we set $\alpha = 0.5$ to be able to adapt to bandwidth fluctuations of up to 100%, and we provide a resource reservation of 10% to avoid saturating the channel by setting $\beta = 0.9$. The block size for the Raptor encoder was of 10 Mbytes, since this size was several times smaller than the file size used for testing (about 100 Mbytes). Notice that, if delivering small contents, the block size can and should be made equal to that size, thereby optimizing performance. For large sized contents, though, performance gains are achieved by splitting the contents into blocks of smaller size, thereby avoiding high encoding delays per block.

Figure 3.6 shows the results obtained when varying the packet loss rate from 0 to 10%. In this first set of experiments the channel bandwidth is limited to 2Mbps, and the end-to-end delay is of 10 ms. When the packet loss ratio is very low, we find that the throughput attained by the RCDP/UDP solution is about 85% of the HTTP/TCP solution. This difference is due both to the idle times introduced during the Raptor coding and decoding processes, and to the fact that we have left a 10% capacity margin as a preventive measure

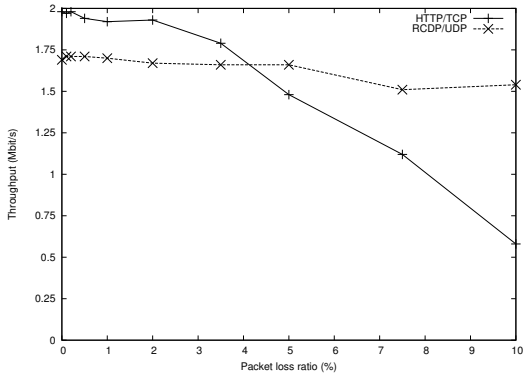


Figure 3.6: Throughput vs. packet loss ratio in a 2Mbps channel with a 10 ms delay.

to avoid channel collapse. However, when the packet loss ratio is greater than 4%, RCDP/UDP achieves a much better performance. In particular, for a packet loss ratio of 10%, RCDP/UDP improves the HTTP/TCP solution by 122%. Adopting RCDP/UDP as a content delivery strategy also achieves a much more stable performance than using HTTP/TCP. We can see that the HTTP/TCP throughput decreases by up to 71% compared to the no-loss situation, while the RCDP/UDP performance never decays by more than 12%.

Figure 3.7 shows the results obtained when varying the end-to-end delay from 0 to 10 ms, being channel bandwidth limited to 2Mbps, and packet loss ratio fixed at 2%.

We find that for very low delay values (less than 20 ms) the HTTP/TCP delivery solution performs slightly better than the RCDP/UDP, being the reasons for this difference the same as explained above. However, for delay values greater than 20 ms, RCDP/UDP performs significantly better, offering improvements over HTTP/TCP of up to 64%. Again we find that the RCDP/UDP solution achieves a much more stable performance, being that, in the worst case, the throughput differences are of only 8%, while with HTTP/TCP we can observe differences of up to 50%. To understand why delay has such a strong impact on performance for HTTP/TCP, notice that packet losses cause TCP's congestion window size to be reduced, and that is grows with a speed that is proportional to the routing-trip time (RTT). Thus, for a same packet loss rate, higher delays are associated with lower throughput values when using TCP.

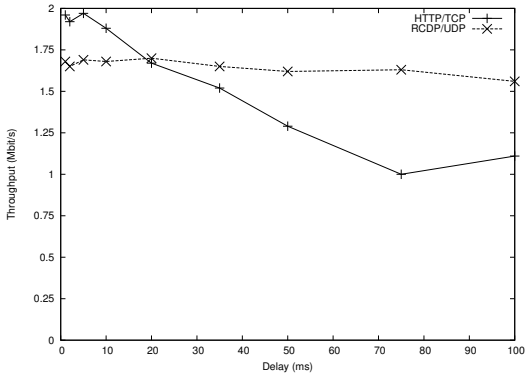


Figure 3.7: Throughput vs. delay in a 2Mbps channel with a 2% packet loss rate.

Finally, figure 3.8 shows the results obtained when varying channel bandwidth between 0.5 and 30 Mbit/s. In this third set of experiments the end-to-end delay is of 10 ms, and packet loss is fixed at 2%. We find that the delay and loss conditions of the channel have a strong impact on HTTP/TCP, which becomes unable to increase its throughput beyond the 5.44 Mbit/s threshold. On the contrary, RCDP/UDP is able to scale much more efficiently despite of the poor channel conditions, achieving significant productivity improvements for channel bandwidth values above 5 Mbit/s. In particular, for a channel bandwidth of 30 Mbit/s, the gain achieved under the conditions defined is of 207%.

Overall, the results show that both content delivery schemes experience a performance degradation as the channel conditions become worse. However, the performance reduction for the HTTP/TCP solution is, in all cases, much more pronounced than the one for RCDP/UDP. In particular, the transmission rate reduction experienced with HTTP/TCP can be considered excessive when the packet loss ratio or the delay are significant. Such results clearly emphasize the lack of effectiveness of solutions based on the HTTP and TCP protocols to handle scenarios where losses are not congestion related. On the contrary, we find that the combination of RCDP and UDP allows maintaining its transmission rate much more stable despite of packet losses, clearly outperforming HTTP/TCP under poor channel conditions. The differences between both content delivery strategies become much more pronounced if the channel bandwidth increases beyond 5 Mbit/s, where we can see that the HTTP/TCP

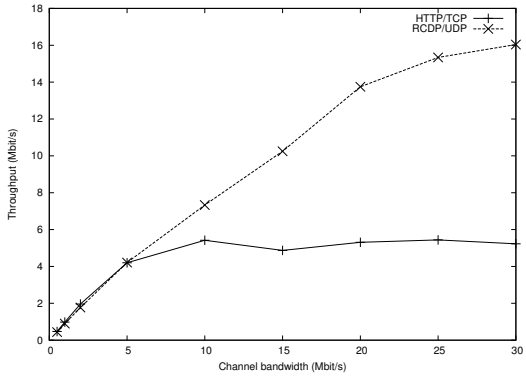


Figure 3.8: Throughput vs. bandwidth in a channel with a 10 ms delay and a 2% packet loss rate.

strategy is unable to make an efficient use of the available bandwidth.

3.7 Conclusions and future work

Content delivery solutions for wireless environments require devising novel strategies that mitigate the impact of channel losses. The adoption of FEC schemes at the application layer has been proposed by different authors as an efficient solution for content delivery in broadcast and multicast IP networks. However, using this strategy for unicast content delivery is a research field that has remained mostly untackled.

In this chapter we propose RCDP, a novel solution that is able to improve the performance of content delivery over wireless networks, especially in very loss-prone environments. RCDP deals with packet losses by relying on application-layer FEC techniques. In particular, it adopts Raptor codes to send an encoded data stream instead of plain data, like other solutions do. This behavior allows RCDP to recover the original contents without the need for retransmissions, window management, per-packet acknowledgments, or timeouts. In fact, with our solution, the receiver only has to keep track of the amount of data received, checking whether this amount is enough to allow decoding a data block. Flow control is performed at the receiver by notifying the sender about successful decoding processes, and then both sender and receiver may move forward to the next data block.

Our sender RCDP agent performs rate control by relying on channel band-

width estimations made at the receiver agent. We proposed enhancements to the bandwidth estimation process by improving the accuracy of such estimations at the receiver, and adopting median-based filtering techniques at the sender. The proposed solution allows the protocol to quickly adapt the transmission rate to channel congestion and other time-varying factors, while offering accurate end-to-end bandwidth estimations.

By relying on an RCDP implementation for the Linux platform, we performed a set of experiments to determine the performance of RCDP over UDP compared to HTTP over TCP, under different channel conditions. Results evidence the superiority of the RCDP/UDP solution when dealing with high delays and high packet loss rates in the channel, including significant improvements in terms of channel usage for bandwidths above 5 Mbit/s.

As future work we will focus on optimizations to the encoding process, and on scalability issues using simulation. Additionally, we will compare our solution against other variants of TCP to further assess the goodness of our proposal.

Acknowledgments

This work was partially supported by the *Ministerio de Ciencia e Innovación*, Spain, under Grant TIN2008-06441-C02-01.

Bibliography

[1] M. Luby, T. Stockhammer, and M. Watson. IPTV Systems, Standards and Architectures: Part II - Application Layer FEC In IPTV Services,. *IEEE Communications Magazine*, 46(5):94–101, 2008.

[2] A. Natani, J. Jakilinki, M. Mohsin, and V. Sharma. TCP for Wireless Networks. *Project Report, Univ. of Texas at Dallas, USA*, 2001.

[3] K. Pentikousis. TCP in wired-cum-wireless environments. *IEEE Communications Surveys*, 3(4):2–14, 2000.

[4] K. Chandran, S. Ragbunathan, S. Venkatesan, and R. Prakash. A feedback based scheme for improving TCP performance in ad-hoc wireless networks. In *Distributed Computing Systems, 1998. Proceedings. 18th International Conference on*, pages 472–479. IEEE, 2002.

[5] J. Liu and S. Singh. ATCP: TCP for mobile ad hoc networks. *Selected Areas in Communications, IEEE Journal on*, 19(7):1300–1315, 2002.

[6] M. Luby, M. Watson, T. Gasiba, T. Stockhammer, and Wen Xu. Raptor codes for reliable download delivery in wireless broadcast systems. In *Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE*, volume 1, pages 192 – 197, January 2006.

[7] A. Shokrollahi. Raptor codes. *Information Theory, IEEE Transactions on*, 52(6):2551–2567, 2006.

[8] T. Paila, M. Luby, R. Lehtonen, V. Roca, and R. Walsh. FLUTE - File Delivery over Unidirectional Transport. RFC 3926, October 2004.

[9] E. Ayanoglu, S. Paul, T.F. LaPorta, K.K. Sabnani, and R.D. Gitlin. AIRMAIL: A link-layer protocol for wireless networks. *Wireless Networks*, 1(1):47–60, 1995.

[10] S. Vangala and M.A. Labrador. Performance of TCP over wireless networks with the Snoop protocol. In *CONFERENCE ON LOCAL COMPUTER NETWORKS*, volume 27, pages 600–604. Citeseer, 2002.

[11] C. Parsa. TULIP: A link-level protocol for improving TCP over wireless links. In *Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE*, pages 1253–1257. IEEE, 2002.

[12] K. Brown and S. Singh. M-TCP: TCP for mobile cellular networks. *ACM SIGCOMM Computer Communication Review*, 27(5):19–43, 1997.

[13] P. Sinha, T. Nandagopal, N. Venkitaraman, R. Sivakumar, and V. Bharghavan. WTCP: A reliable transport protocol for wireless wide-area networks. *Wireless Networks*, 8(2/3):301–316, 2002.

[14] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. RFC2018: TCP Selective Acknowledgement Options. *RFC Editor United States*, 1996.

[15] S. Keshav and SP Morgan. SMART retransmission: Performance with overload and random losses. In *INFOCOM'97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pages 1131–1138. IEEE, 2002.

[16] R. Caceres and L. Iftode. Improving the performance of reliable transport protocols in mobile computing environments. *Selected Areas in Communications, IEEE Journal on*, 13(5):850–857, 2002.

[17] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, June 1999.

[18] John W. Byers, Michael Luby, Michael Mitzenmacher, and Ashutosh Rege. A digital fountain approach to reliable distribution of bulk data. *SIGCOMM Comput. Commun. Rev.*, 28:56–67, October 1998.

[19] Carlos T. Calafate, Pietro Manzoni, and Manuel P. Malumbres. Supporting soft real-time services in MANETs using distributed admission control and IEEE 802.11e technology. In *The 10th IEEE Symposium on Computers and Communications*, La Manga del Mar Menor, Cartagena, Spain, June 2005.

[20] Y. Gu and R.L. Grossman. Supporting configurable congestion control in data transport services. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*. IEEE Computer Society, 2005.

Chapter 4

Analysis and Evaluation of a Raptor-based Content Delivery Protocol and its High-performance Extensions

Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. *Analysis and evaluation of a raptor-based content delivery protocol and its high-performance extensions.* Ad Hoc & Sensor Wireless Networks , 26(1-4):125-149, 2015.

Abstract

Forward Error Correction (FEC) has proved to be a very effective solution in the presence of challenging network environments. Its usage is focused mainly on multicast and broadcast delivery, but its impact on research is much wider. For example, FEC can also be used for unicast content delivery to avoid known TCP issues in wireless network environments. In this chapter we propose RCDP, a solution that exploits the error resilience properties of Raptor Codes to offer reliable and bidirectional unicast communication in lossy links. Based on an initial approach, different architectural choices are proposed to seek a trade-off between complexity and efficiency, and offer suitable instances of the

protocol for real systems. Experimental results show that RCDP is a highly efficient solution for environments characterized by high delays and packet losses (e.g. ad-hoc networks), achieving significant performance improvements compared to traditional transport-layer protocols.

4.1 Introduction

Several research efforts have been made in order to solve the TCP performance issues in lossy links. However, most of the proposed alternatives to TCP lack the flexibility required to allow their implementation on current hardware, or they require extra nodes to run. There are many specific-purpose systems which must achieve a higher performance level, even sacrificing the TCP compatibility, like sensor networks, point-to-point satellite connections or vehicular ad-hoc networks. These systems attempt to efficiently use the available network resources, no matter whether it is wired, wireless, or a combination of both. To achieve this goal, the protocols involved have throughput maximization as a primary objective. Nevertheless, protocol design must also take into account other characteristics such as flexibility, scalability and reliability, seeking an optimal network experience in all sorts of devices [7].

Efficient Application-Layer Forward Error Correction (AL-FEC) algorithms have been proposed recently. Most authors adopted such algorithms to address challenges associated with multicasting/broadcasting to a high number of users. Nevertheless, the applicability of AL-FEC solutions is much wider, and several fields of application remain unexplored. One of them is unicast content delivery over wireless networks, where lossy transmission channels reduce the throughput achieved by conventional TCP-based delivery protocols. Most of the TCP issues in wireless networks are described in [21].

In this chapter, we propose a novel protocol for end-to-end content delivery in wireless networks that adopts the aforementioned characteristics. Our solution, named Raptor-based Content Delivery Protocol (RCDP), uses an Application-Layer Forward Error Correction (AL-FEC) scheme based on Raptor codes [33] to achieve efficient, reliable and bidirectional unicast data transmission in loss-prone environments. In particular, RCDP is oriented to wireless networks and wired/wireless mixed environments. It is able to provide a reliable content delivery solution achieving a higher degree of efficiency than TCP while avoiding specific hardware or additional nodes. In terms of implementation, we propose different architectural and design alternatives at both client and server that seek an optimal trade-off between throughput and resource consumption. Experimental testbed results show that our application-layer

implementation of RCDP is able to achieve significant performance improvements compared to existing transport layer protocols.

The rest of this chapter is organized as follows: Section 4.2 reviews different protocols available in the literature that attempt to optimize content delivery in wireless environments. Section 4.3 explains how the proposed RCDP protocol works, including its main tasks and components. Its behavior is tested in Section 4.4. Section 4.5 details the different improvements that were implemented, discussing the complexity of these changes, and how they could affect the overall performance. In Section 4.6, we thoroughly evaluate the different implementation alternatives to assess their performance benefits; afterwards, we compare our solution against other existing transport protocols. Finally, Section 4.7 concludes the chapter.

4.2 Related Work

Developing efficient content delivery protocols for wireless environments has received much attention from the research community. Several performance studies have been done to evaluate such solutions in terms of both design and implementation. The works available in the literature can be split into four different groups [5]: link-layer solutions, split-connection solutions, TCP-enhancements, and FEC based solutions.

In terms of link layer solutions, protocols like Snoop [6] and Tulip [29] attempt to improve the performance of higher layer protocols by making the link-layer aware of on-going connections. Another important protocol is Rate-More [17], which uses a coding scheme to predict the number of symbols needed to recover the message. This group tends to move towards cross-layer solutions, like [11] and [27], that also focus on modifying the transport layer to enhance the global performance.

Split-connection approaches, like Mobile TCP [8], Wireless-TCP [34] and MWTCP [26], attempt to improve TCP performance in wireless environments by dividing the TCP connection into two separated connections. Both link layer and split-connection approaches require performing changes in some of the network's intermediate elements, which can be a drawback when attempting to deploy them.

Concerning those solutions that enhance the TCP protocol, TCP Westwood [9] offers substantial performance improvements in wireless networks with lossy links compared to legacy protocols, such as TCP SACK and TCP Reno. Mo-TCP [28] is another solution addressing heterogeneous wireless networks by adjusting its operations according to underlying link and network

conditions. TCP-AP [14] adaptively sets the transmission rate using the current delay and coefficient of variation of recent RTTs. There are a lot of techniques to enhance TCP, as discussed in [15], and most of them are end-to-end solutions, avoiding any changes at intermediate network elements.

Concerning content delivery solutions that explicitly adopt Raptor-based FEC, Luby et al. [22] propose a solution for reliable file delivery over 3GPP mobile broadcast networks, using Raptor codes to offer Multimedia Broadcast and Multicast Services (MBMS). Chiao et al. [12] describe the experience of using the FLUTE protocol, which is a multicast protocol for unidirectional communication, for file delivery over a WiMAX unicast network. However, notice that these Raptor based approaches rely on unidirectional file pushing to clients, being applicable to multicast and broadcast scenarios without bidirectional communication requirements.

Recently, an explosion of Raptor-based schemes has occurred. The main target area of these codes is broadcasting, where they have been used in a wide range of applications, such as generic information dissemination [20], infotainment [3], video streaming [10], and safety [2]. However, Raptor codes are becoming a basic tool to carry out innovative research in other fields related to computer networks. Starting from P2P schemes [13], they are moving to vehicular environments [19] and starting to offer other schemes in this kind of networks. New applications of this coding technique have been presented for distributed storage [32], infrastructure to vehicle communication [4], and DTN networks [31]. Moreover, going a step further, some researchers have started to present new distributed coding schemes [35].

Our proposal differs from the previous ones by taking a completely novel approach. In particular, we rely on Raptor-based FEC to provide a unicast content delivery solution that offers reliable bidirectional communications (like TCP), while completely avoiding packet retransmissions (unlike TCP). Thus, no windowing or retransmission control has to be performed. Instead, the proposed solution relies on bandwidth estimations at the endpoints to perform rate control based on the end-to-end congestion state.

4.3 The RCDP Protocol

RCDP is a content delivery protocol that was developed focusing on wireless network scenarios. Thus, it focuses on environments characterized by low end-to-end throughput, high error rates and high retransmission delays. To avoid poor performance under these conditions, it relies on a Forward Error Correction strategy, known as Raptor encoding [33], to ensure that any piece

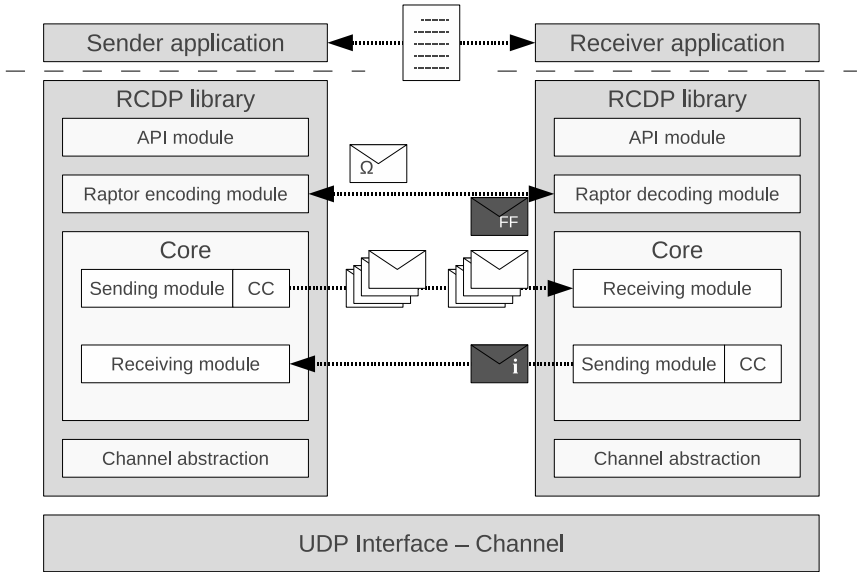


Figure 4.1: Simple RCDP implementation diagram.

of information sent can be successfully recovered at the final destination, even when part of that information has been lost.

4.3.1 Raptor codes

Our Raptor codes technique send encoded data by splitting it into blocks, typically bigger than 1 MByte, which are coded separately. Each block is split into symbols whose size is typically made equal to the maximum size that fits into a single packet; these symbols are then used as input to a two stage encoding process. In the first stage, where input data are referred to as *source symbols*, pre-coded symbols are generated. Then, through an arithmetic combination of these pre-coded symbols, an infinite number of encoded symbols can be generated. To recover information at the destination, any combination of the original symbols and the recovery symbols allows retrieving the original information. Although theoretically this task can be done in linear cost time [33], the final performance can vary depending on the implementation of the coding system. An example of a hardware and software implementation of this coding technique has been provided and analyzed in [25]. For the most recent

version of the Raptor libraries [1], the probability of successfully decoding a total of r received symbols is:

$$P_{dec} > 1 - 10^{-2(r-k+1)},\ r \geq k \tag{4.1}$$

This means that, to recover a source block with symbol size k, the probability of a successful decoding is greater than 99% if k encoded symbols are received, greater than 99.99% if $k+1$ encoded symbols are received, and greater than 99.9999% if $k+2$ encoded symbols are received.

The RCDP protocol is full-duplex, encompassing both sending and receiving processes. At the sender side, the contents requested are partitioned and encoded as stated above. Afterward, each symbol is placed in a single UDP packet to be sent to the receiver. Notice that, in order to help the receiver in the decoding tasks, all the information required to recover a block is available in each packet's header, namely the number of source symbols, the symbol identifier, the block identifier, and the real block size. In order to take advantage of forward recovery capabilities, lost symbols are not sent again, but a continuous flow of symbols are delivered until the receiver is able to recover the original information; such event is notified to the sender with a control packet, named *successful block recovery* packet. When this control packet is received by the sender, delivery of the next data block begins. Since control packets are generated periodically, the loss of such a packet will be recovered by the next control packet successfully received.

The sequence of actions taken by the receiver are complementary to those of the sender: the receiver is continuously listening to incoming symbols, storing them in memory upon arrival. When enough symbols to recover a block have been received, the receiver sends back the *successful block recovery* notification and starts the decoding procedures; this strategy allows the sender to switch to the following block as soon as possible.

4.3.2 Flow and rate control

Since the proposed strategy does not adopt the *sliding windows* approach, flow and rate control tasks can easily be made independent. In our solution, flow control operates on a block basis, as described above, while rate control is made on a packet basis. In particular, rate control relies on end-to-end estimations of available bandwidth based on packet arrival patterns. The sender continuously adjusts the transmission rate according to these estimations, generating packet trains [30] towards the receiver. This strategy requires all packets to be marked with a train identifier, as well as recording the arrival time for

each packet at the receiver. Bandwidth estimations are made based on information about arrival times for the first and the last packet of a train, along with the number of packets received per train. This estimation is sent back to the sender for every train ID in a control message, allowing it to dynamically adapt its transmission rate. Notice that this technique seamlessly adapts to network congestion since congestion will cause the time between consecutive packets in a train to increase, thereby decreasing the bandwidth value in the estimations made.

Packet trains are generated as follows: starting from the bandwidth estimation described above, the sender applies a β correction to this value, obtaining a target data rate ($\bar{R}$) that will usually be slightly lower than the estimated bandwidth (BW_e):

$$\bar{R} = \beta \times BW_e,\ 0 < \beta \leq 1 \tag{4.2}$$

The available bandwidth value must be maintained to avoid saturating the channel. Notice that higher β values imply that the network will be working close to saturation. In order to detect when the bandwidth available increases, packets are grouped in a train, and the train rate (T_r) is defined as:

$$T_r = \frac{1}{\alpha} \times \bar{R},\ 0 < \alpha \leq 1 \tag{4.3}$$

In this Equation 4.3, parameter α allows adjusting the degree of burstiness. In particular, lower values for α are synonym of lower inter-packet times. Notice that T_r will be always higher than, or equal to, the target data rate ($\bar{R}$), being the latter the average data rate for each train period. To make sure that $\bar{R}$ is maintained, all packet trains are followed by a pause period (inter-train time).

4.3.3 Protocol implementation

We have designed a solution offering the basic functionality required for the RCDP protocol to be operative. Using the UDT library [16] as a starting point, RCDP was created following the architecture shown in Figure 4.1. As shown in the figure, we followed a layered approach to implement our solution. The library has four layers: (i) the API module, (ii) the Raptor coding modules, (iii) the core, and (iv) the channel abstraction.

The first module (API) acts as an interface between top level applications and the services offered by the library. It is the responsible for receiving and delivering data from and to the application. The second module acts as a data

Table 4.1: Summary of Raptor codes implementation and usage information.

Parameter	Value
Raptor codes implementation	DF Raptor™ R11 Encoder/Decoder 2.2
Vendor	Digital Fountain, Inc.
Block size (B)	10000000
Symbol size (B)	1440
Number of repair symbols	As many as required to recover the block

encoder and decoder. At the sender side, blocks are encoded into symbols to be sent. At the receiver side, symbols are decoded to recover blocks. The sending and receiving modules are the main modules of the core, which are responsible for rate control. The control packet management is also performed by this layer. Finally, RCDP includes a channel abstraction module which simply sends and receives packets to and from an UDP pipe.

Table 4.1 specifies all data related to the Raptor codes implementation and usage information.

Figure 4.1 also summarizes the different kinds of packets implemented by our protocol: packets that carry symbols (labeled as Ω), *successful block recovery* control packets (labeled as *FF*), and *available bandwidth estimation* control packets (labeled as i).

4.4 Preliminary results

We performed a set of basic tests in order to validate the proposed RCDP protocol. We aimed at checking whether properties, such as loss resilience, have been achieved, and assessing its performance under different channel conditions. For comparison purposes, we also present performance results obtained with the most widely used content delivery solution, which is based on the combination of HTTP and TCP. Obviously, since HTTP is a very mature standard, it offers a lot more functionalities than RCDP. To achieve a fair comparison, we have compared them in terms of performance, carrying out the most basic service: request and delivery of a single file.

We devised a set of tests comparing both solutions under the same conditions of error, delay and channel bandwidth. Our experiments were made with real working software for the GNU/Linux platform. In order to make sure that the test sequences were reproducible, we created a controlled environment for testing where we emulated different channel conditions. It is a network black

box that emulates a configurable point-to-point connection. This emulation is developed using two computers (client and server) connected through an Ethernet switch. The wireless conditions are emulated using the linux *tc* tool, which ensures a controlled packet loss and delay, among other parameters. The proposed network black box was initially validated to make sure that all the test results were reliable, and that the comparisons made were fair.

For evaluation we created a very simple content provider (server) and content requester (client) that can operate with any of the two content delivery solutions under evaluation: RCDP over UDP, or HTTP over TCP. We emulated the channel conditions of different wireless network environments by defining different values for delay, packet loss ratio, and channel bandwidth. In our tests the client requests a large-sized content from the server, and calculates the productivity (in Mbps) associated with the transmission of that content. For each configuration we averaged five samples to obtain each of the points represented. Concerning both α and β parameters, we set $\alpha = 0.5$ to be able to adapt to bandwidth fluctuations of up to 100%, and we provide a resource reservation of 10% to avoid saturating the channel by setting $\beta = 0.9$. Configuration related to Raptor coding is shown in Table 4.1. Notice that, when delivering small-sized contents, the block size can and should be made equal to the content size, thereby optimizing performance. For large sized contents, though, performance gains are achieved by splitting the contents into blocks of smaller size, thereby avoiding high encoding delays per block.

Figure 4.2 shows the results obtained when varying the packet loss rate from 0 to 10%. In this first set of experiments the channel bandwidth is limited to 2 Mbps, and the end-to-end delay is of 10 ms. When the packet loss ratio is under 3%, we find that the throughput attained by the RCDP/UDP solution is about 85% of the HTTP/TCP solution. This difference is due to both the idle times introduced during the Raptor coding and decoding processes, and to the fact that we have left a 10% capacity margin as a preventive measure to avoid channel collapse. However, when the packet loss ratio is greater than 4%, RCDP/UDP achieves a much better performance. In particular, for a packet loss ratio of 10%, RCDP/UDP improves the HTTP/TCP solution by 122%. Adopting RCDP/UDP as a content delivery strategy also achieves a much more stable performance than using HTTP/TCP. We can see that the HTTP/TCP throughput decreases by up to 71% compared to the no-loss situation, while the RCDP/UDP performance never drops by more than 12%.

Figure 4.3 shows the results obtained when varying the end-to-end delay from 0 to 100 ms, being channel bandwidth limited to 2 Mbps, and packet loss ratio fixed at 2%.

We find that, for very low delay values (less than 20 ms), the HTTP/TCP

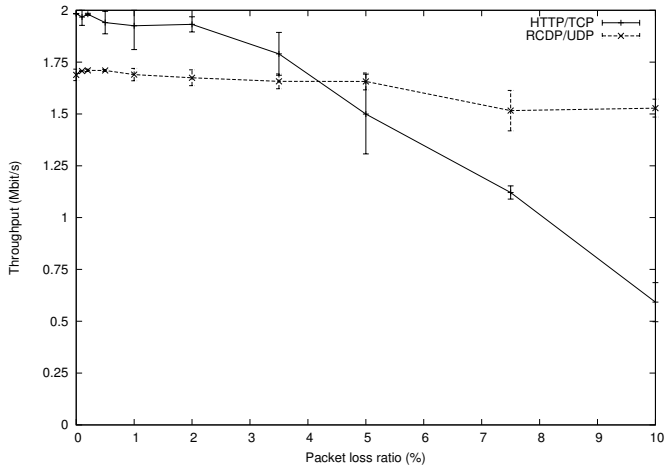


Figure 4.2: Throughput vs. packet loss ratio in a 2Mbps channel with a 10 ms delay.

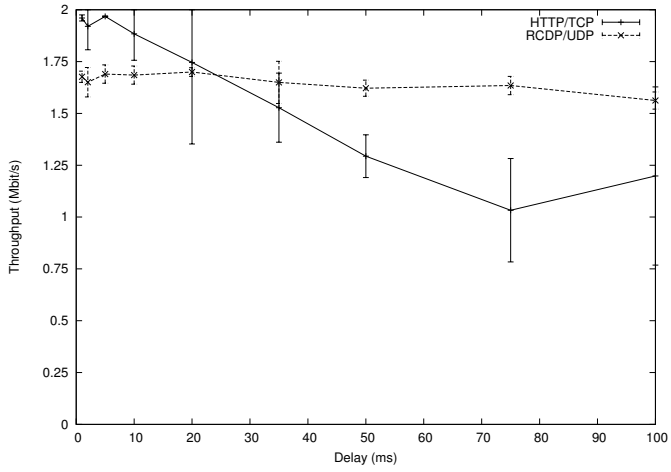


Figure 4.3: Throughput vs. delay in a 2 Mbps channel with a 2% packet loss rate.

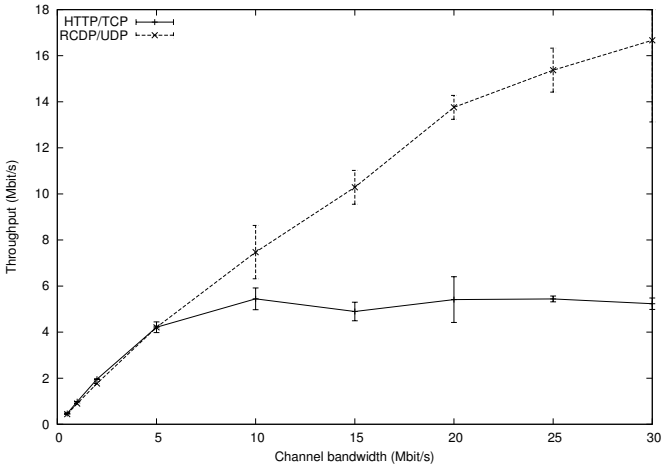


Figure 4.4: Throughput vs. bandwidth in a channel with a 10 ms delay and a 2% packet loss rate.

delivery solution performs slightly better than RCDP/UDP, being the reasons for this difference the same as explained above. However, for delay values greater than 20 ms, RCDP/UDP performs significantly better, offering improvements over HTTP/TCP of up to 64%. Again, we find that the RCDP/UDP solution achieves a much more stable performance, being that, in the worst case, the throughput differences only are of 8%, while with HTTP/TCP we can observe differences up to 50%. To understand why delay has such a strong impact on the performance of HTTP/TCP, notice that packet losses cause TCP's congestion window size, which grows with a speed that is proportional to the round-trip time (RTT), to be reduced. Thus, for a same packet loss rate, higher delays are associated with lower throughput values when using TCP.

Finally, Figure 4.4 shows the results obtained when varying channel bandwidth between 0.5 and 30 Mbit/s. In this third set of experiments, the end-to-end delay is set equal to 10 ms, and packet loss is fixed at 2%. We find that the delay and loss conditions of the channel have a strong impact on HTTP/TCP, which becomes unable to increase its throughput beyond the 5.44 Mbit/s threshold. This occurs because, beyond this channel capacity, the bottleneck is TCP performance itself, and not the network anymore. On the

contrary, RCDP/UDP is able to scale much more efficiently despite of the poor channel conditions, achieving significant productivity improvements for channel bandwidth values above 5 Mbit/s. In particular, for a channel bandwidth of 30 Mbit/s, the gain achieved under the conditions defined is of 207%.

Overall, the results show that both content delivery schemes experience a performance degradation as the channel conditions become worse. However, the performance reduction for the HTTP/TCP solution is, in all cases, much more pronounced than the one for RCDP/UDP. In particular, the transmission rate reduction experienced with HTTP/TCP can be considered excessive when the packet loss ratio or the delay are significant. Such results clearly emphasize the lack of effectiveness of solutions based on the HTTP and TCP protocols to handle scenarios where losses are not congestion related, such as what happens in wireless scenarios. On the contrary, we find that the combination of RCDP and UDP allows maintaining its transmission rate much more stable despite of packet losses, clearly outperforming HTTP/TCP under poor channel conditions. The differences between both content delivery strategies become much more pronounced if the channel bandwidth increases beyond 5 Mbit/s, where we can observe that the HTTP/TCP strategy is unable to make an efficient use of the available bandwidth.

However, an interesting behavior is observed in scenarios with low packet loss ratios and low delays, being that HTTP/TCP outperforms RCDP in terms of throughput. We find that this difference is associated with different factors; one of them is the overhead of context switching from user mode to kernel mode. Since RCDP is executed at the application level, every UDP packet sent requires switching to kernel mode. TCP can handle data in a more coarse grain fashion, thus reducing the number of context switches dramatically. Another factor that contributes to this difference is the CPU and memory overhead introduced by the coding scheme during the transmission process. The coding process carried out by the RCDP protocol must hold in memory the entire block to be encoded or decoded, and must spend a block-size linear-dependent time to perform any of these tasks in cases where the Raptor Codes implementation can achieve its theoretical limits. All this extra time wastage is also avoided in TCP. The last factor that explains this behavior is the bandwidth measurement precision. TCP only cares about packet losses to detect a congestion process; this is a discrete event that may happen or not, avoiding any actual bandwidth calculation. Differently from TCP, RCDP must be highly accurate at creating the packet trains at the source, and at measuring the inter-packet arrivals at the destination. However, issues like task sharing at the CPU level, or thread management and wake up precision at the user plane, add noise to the bandwidth measurement process as well.

In the following section we will address these problems, attempting to reduce, mask or remove them whenever possible, so improving the performance of our RCDP approach.

4.5 High-performance extensions to RCDP

AL-FEC techniques are directly connected to the high computational cost of the coding and decoding process. RCDP uses Raptor codes which, despite offering a theoretical linear cost in their coding and decoding algorithms, have implementation-depended issues which complicate achieving such linear cost increase. Thus, we still require a very efficient implementation for the proposed solution to be efficient as a whole. Moreover, since we adopt a user-level development approach, there are additional delays associated with the switching between kernel and user modes that do not appear in kernel level approaches, and whose effects should be mitigated.

4.5.1 Baseline optimization

To optimize the performance of our RCDP implementation, we have to tackle several issues. The first one is related to the coding module. Since we are using systematic Raptor codes [33], the first output symbols from the encoder are the source symbols themselves, and so no pre-processing for this first set of symbols is required: they can be sent without actually requiring any encoding to take place. However, the Raptor encoding process is mandatory to create the recovery symbols. Therefore, a delay between the first (source) and the second (recovery) set of symbols is introduced. To optimize this sequence of tasks, we reconfigure the coding process so as to eliminate the delay between the two sets of symbols. This strategy avoids introducing periods when no packets are sent.

The first approach to implement this optimization is to split the coding process into smaller slices, interleaving them with the delivery of source symbols. However, this approach could introduce additional problems related to the regularity with which the system is able to deliver packets due to the high CPU usage and low granularity level at this point. Another approach is to optimize the buffer's size. If we consider the encoding process as an irregular injection of symbols to be sent, instead of using two periods of symbol generation followed by an intermediate pause, we can use a buffer to regulate symbol

generation to the lower layers. In this case, the only parameter that must be correctly tuned is the buffer's size. We must ensure that the transmission time of the buffered packets will be greater than the coding time to avoid starvation at the queue level, as shown in Equation 4.4:

$$B_s \geq \frac{T_c \cdot BW}{P_s} \tag{4.4}$$

where B_s is the minimum size that the queue buffer should have, in number of packets, T_c is the block coding time, BW is the maximum bandwidth that the channel can achieve (in bits per second), and P_s is the packet size (in bits).

A second issue that must be considered is related to the timing accuracy for the packet generation process. The proposed solution to this problem relies on a two phase approach. In the first phase, a timed wait corresponding to a fraction of the sleeping time takes place. In a second phase, the last part of the idle period is a busy waiting.

A third element prone to optimization is the instant when the source is warned about the correct decoding of the current block. By default, this occurs only when the block decoding procedure is successfully completed. However, since the Raptor libraries provide feedback about the viability of the decoding process even before this process has started, the receiver can warn the sender about it much earlier, thus avoiding wasting time and network resources by preventing the generation of additional recovery symbols when they are no longer required.

In summary, the baseline optimizations introduced are the following: (i) buffer size tuning to regulate symbol generation, (ii) increased packet injection time accuracy, and (iii) early feedback from the receiver about successful decoding. A prototype of RCDP encompassing these optimizations, tagged as *RCDP+BO*, will be compared against other alternative solutions in Section 4.6.

4.5.2 Multithreading support

In this section we propose a performance improvement strategy that exploits parallel processing.

We will use two independent threads to code data blocks in parallel to overlap two different coding processes, thereby avoiding idle periods in the network. To achieve this goal, when a first block is being sent, the next block starts being coded. Due to this early load of the next block, the sending process will be improved by parallelizing all the management structures. The ability of decoding up to two different blocks is also introduced, and this can

be used, similarly to parallel coding, to get a decoder ready to work without delay while the previous block is being decoded.

When several threads are working cooperatively, as in the aforementioned cases, processing overhead associated with thread management can become a problem. As in all processes whose complexity is incremented, additional software overhead must be included. This introduces processing delays, which could downgrade performance compared to simpler, sequential implementations. Therefore, it is important to determine the optimal trade-off between parallelization and overhead to achieve maximum performance.

Several prototypes were made to evaluate each of the improvements separately. Both RCDP+PE and RCDP+PE+BO include the capability of parallel encoding up to two blocks. The only difference is that the latter also adopts the optimizations proposed in 4.5.1. RCDP+PED upgrades RCDP+PE+BO to parallel decoding up to two blocks. A solution where encoding process is handled by an independent thread, RCDP+PEIT, is implemented for comparison purposes. All these versions are compared in Section 4.6.

4.5.3 Design optimizations

Handling block acknowledgements is a main issue for all coding-based protocols. For example, link-layer approaches usually rely on "per packet" acknowledgements, which may lead them to have an undesirable channel overhead. In application-layer approaches we can limit the acknowledgements to application-layer messages. Also, acknowledgement messages can be combined with rate control control packets to further reduce this overhead.

In RCDP, acknowledgments refer to Raptor source blocks. In order to notify the sender that it may proceed to deliver symbols of the following block, a successfully recovered block notification is sent by the receiver. In high delay networks, a lot of unnecessary recovery symbols will be sent and too much time will be wasted until the sender realizes that the current block has already been decoded successfully. In this section we propose a protocol enhancement that can alleviate this problem.

The parallel coding design introduced in the previous section allows generating symbols from two different data blocks, if needed. Such a combination of symbols could be particularly interesting for the period that begins when all source symbols are sent, and recovery symbols start being generated, since we do not exactly know how many recovery symbols are actually required. Thus, to prevent occupying the whole transmission medium with redundant information for a period equivalent to, at least, one round-trip time, we propose mixing symbols of two consecutive data blocks in order to perform a gradual

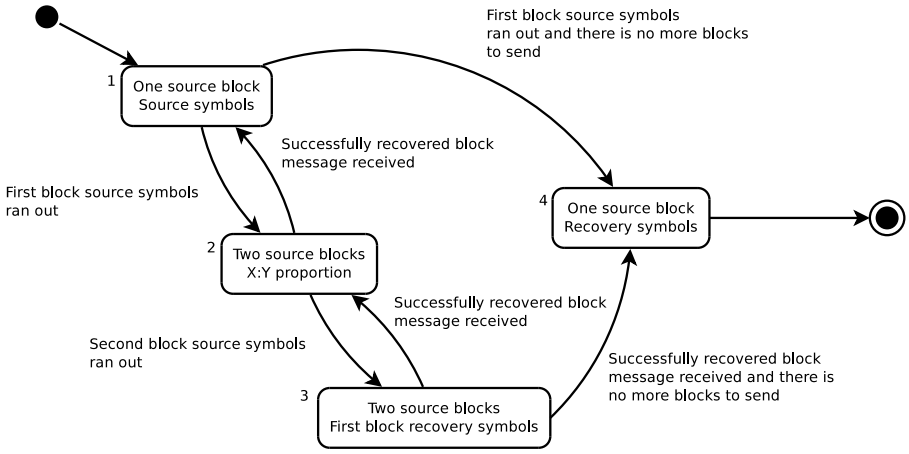


Figure 4.5: Design optimization state diagram

transition from the first to the second block. Thus, by mixing recovery symbols from the current block with source symbols from the next block, we ensure that at least part of the information sent during such period is useful to the receiver.

Figure 4.5 shows the state diagram for RCDP when adopting the proposed enhancement. Each state shows the number of coding blocks, and which symbols are being transmitted. The state that introduces the proposed enhancement is state two. There, the protocol is able to send symbols from different blocks following an X:Y ratio, where X is the number of source symbols for one block and Y is the number of recovery symbols for the other block. As seen in the diagram, the protocol switches between states when an appropriate message arrives, or when running out of source symbols. The first and fourth states represent the start and the end of a transmission when there is only a single block to send, and state three is achieved by the protocol in environments characterized by an extremely high loss rate. In that case, priority is given to block recovery symbols.

For evaluation purposes we developed the RCDP Mixed Blocks implementation (RCDP+PED+MB), a solution able to mix recovery symbols of the current block and source symbols of the next block following the strategy described above. In particular, the recovery vs. source symbols ratio can be set to any value (X:Y). The ratio can be optimized to different packet loss ratios,

and it should be lower when channel losses are higher. Finally, in the unlikely event that transmission of source symbols of the second block is completed before the first block is recovered, only recovery symbols for the first blocked are generated to promote an ordered recovery of transmitted blocks.

4.6 Performance evaluation

In this section we will quantify the difference between the original implementation, described in Section 4.3, and the optimized ones described in Section 4.5. The seven RCDP versions under analysis are the following:

- Initial implementation:
  - RCDP: This implementation corresponds with the initial one described in Section 4.3. It implements the four layered approach without any other optimization.
- Implementation-based high performance optimizations:
  - RCDP+BO: This implementation contains a buffer size large enough to avoid interruptions during the sending process, as explained in Section 4.5.1. This implementation also includes the two phased process described in that section, and anticipates the delivery of *successfully recovered block* notifications.
  - RCDP+PE: This implementation includes multithreading support, offering the parallel encoding of two different blocks as described in Section 4.5.2.
  - RCDP+PE+BO: A prototype which combines the two previous optimizations.
  - RCDP+PED: An upgrade of the RCDP+PE+BO solution where the decoder is initialized earlier so as to prepare a slot for decoding a block.
  - RCDP+PEIT: A solution that is similar to the previous one, but where the encoding process is handled by an independent thread.
- Design-based high performance optimizations:
  - RCDP+PED+MB: Includes the enhancements described for solution RCDP+PED, as well as the ability of sending, receiving, and

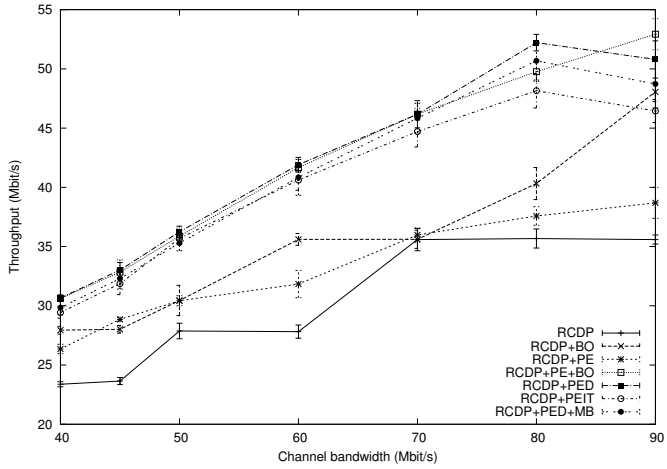


Figure 4.6: Throughput vs. available bandwidth in a network with a 10 ms end-to-end delay (null error rate).

storing mixed symbols from two different blocks, as described in Section 4.5.3. The selected ratio of source vs. recovery symbols is 1:4.

To test the effectiveness of the different RCDP optimizations, we use the same network black box environment described in Section 4.4. Also, for each RCDP version, we implement both server and client applications, where the latter requests a very large file to the former. We took ten samples for all the selected configurations.

4.6.1 Performance evaluation results

In this section we study the performance of the different RCDP versions when varying the available channel bandwidth, the end-to-end packet loss ratio and the size of the contents to be delivered.

Figure 4.6 shows the throughput achieved when varying the available bandwidth. We see that, in general, the different RCDP versions tested behave as expected, experiencing an almost linear throughput increase as the available bandwidth increases. In these results, two clear groups can be observed: those that include both baseline and multithreading optimizations, and the rest. We

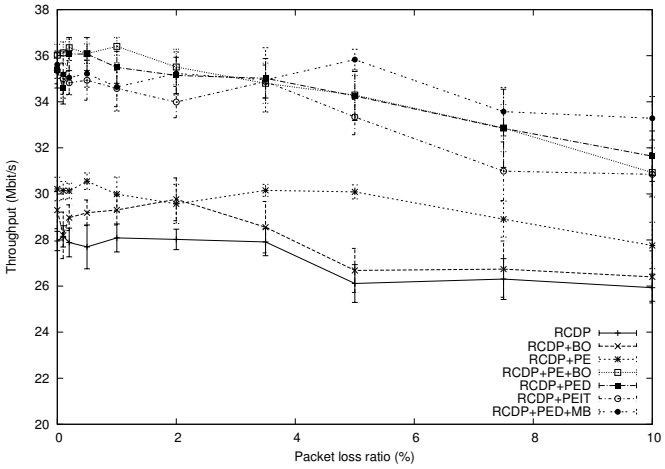


Figure 4.7: Throughput vs. packet loss rate in a 50 Mbps channel with a 10 ms delay.

find that the former are able to achieve a higher degree of productivity compared to the latter ones. In particular, when compared to the original RCDP implementation, combining baseline optimizations with parallel encoding allows increasing throughput by about 45% in the best case, which is a very substantial improvement. Besides throughput enhancements, the block precharge (PED) technique described in Section 4.5.2, along with the buffer size optimization described in Section 4.5.1, enables the generation of a continuous symbol flow which contributes to a more regular transmission rate compared to both RCDP+BO and the original RCDP.

In Figure 4.7 we can see the throughput performance when varying the packet loss rate in the network. The desired behavior would show a linear throughput decrease as packet loss ratio increase. We find that, in general, all the RCDP versions approximately follow this trend. Similarly to the previous results, two well defined groups can be identified. Results show that, in general, error immunity remains similar to the original RCDP implementation, although actual throughput values basically depend on the different enhancements proposed.

In Figure 4.8 we can see the throughput performance when varying the size of the delivered content. Notice that, when the file size is small, the

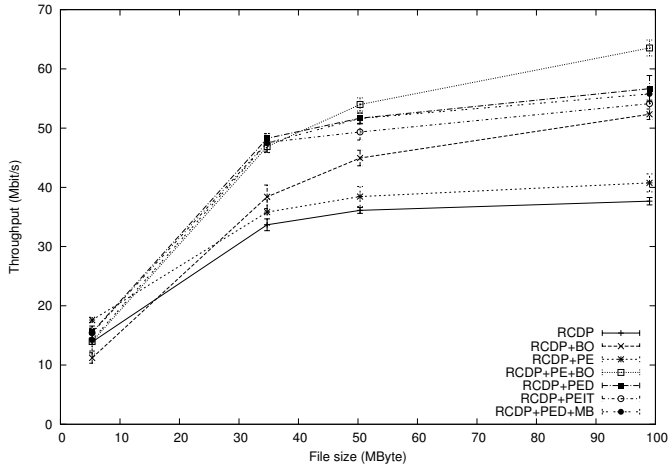


Figure 4.8: Throughput vs. file size in a network environment with 100 Mbps bandwidth and a 10 ms delay (null error rate).

average throughput is low because the initial startup time overhead is similar to the transmission time. A similar effect occurs with TCP as well. For greater file sizes, the impact of the startup times on throughput becomes negligible. We find that the two groups of implementations detected earlier can still be identified, although the two simplest implementations (RCDP+PE and RCDP+BO) are more difficult to classify than the others because they show a different trend. This difference is due to the software complexity of the different solutions. In particular, the block management strategy for these two implementations is more lightweight (fewer threads, fewer program instructions to execute, lower memory usage) than for the other implementations; therefore, for a high memory and CPU demanding software, such as Raptor coding, it has an impact in terms of achieved throughput.

The impact of adding additional threads can be seen by comparing RCDP+PED and RCDP+PEIT. The only difference between these two implementations is that a new thread is added to the coding process, as explained in Section 4.5.2. By adding this new thread, delays increase due to scheduling, dispatching and coordination tasks, which slightly decrease the throughput of this implementation (up to 5%). Also, this approach needs more RAM to be carried out, meaning that the performance of this implementation is affected by memory

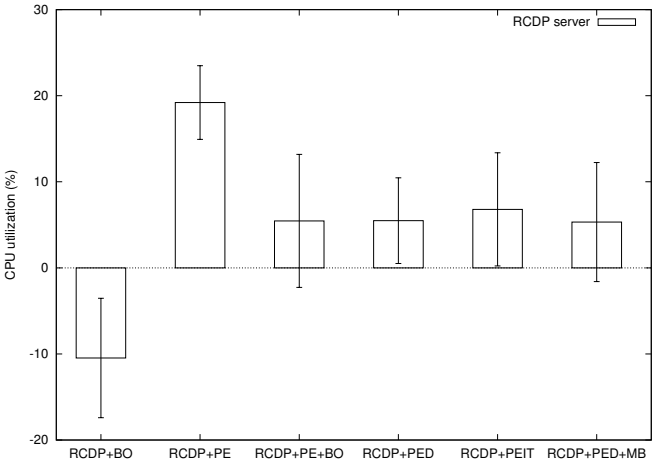


Figure 4.9: Additional CPU overhead at the server compared to the original RCDP implementation.

access delays.

4.6.2 Resource consumption analysis

We now focus on the computational resources required by RCDP and the proposed optimizations. The results were obtained using the Linux "ps" tool as a background process, which was in charge of periodically measuring the CPU utilization (CPU time used divided by the time the process has been running) and RAM utilization at both client and server. Notice that, despite the software running on both client and server is similar, and despite communication is bidirectional, data is being transferred from server to client, meaning that the server will be mostly performing data encoding tasks, while the client will be performing decoding tasks instead.

Figures 4.9 and 4.10 show the additional CPU overhead of the different RCDP enhancements compared to the original implementation. At the server side, notice that the CPU utilization increases significantly when parallel coding is adopted, although this growth is significantly mitigated by using the set of baseline optimizations proposed (see Section 4.5.1). When baseline optimizations are introduced, the CPU usage is further reduced. At the

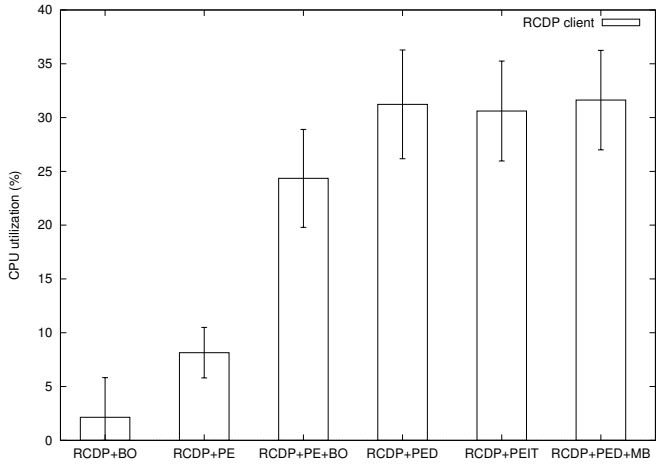


Figure 4.10: Additional CPU overhead at the client compared to the original RCDP implementation.

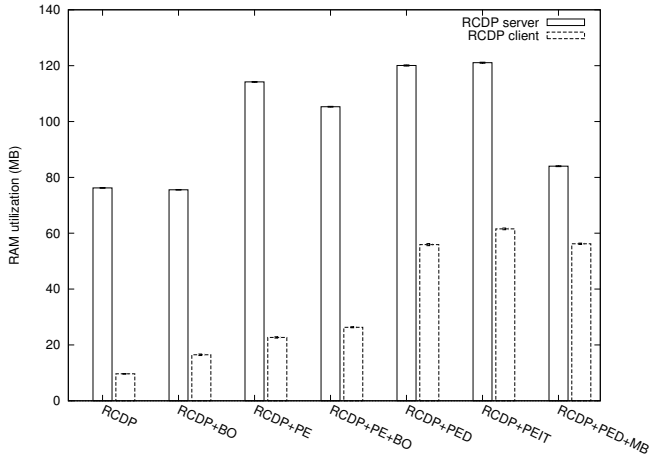


Figure 4.11: RAM utilization at the server and the client.

client side, CPU load is usually quite lower than the server load in absolute terms. Focusing now on the differential analysis, the different RCDP versions shows that the CPU overhead increases significantly when some of the proposed improvements are adopted. In particular, load becomes higher for the family of solutions adopting parallel decoding (RCDP+PED, RCDP+PEIT, and RCDP+PED+MB) due to extra threads introduced to support parallel decoding.

As shown in figure 4.9, there are important performance differences between RCDP+PE and RCDP+PE+BO. Notice that a partial reimplementation of some of the components of our library was required for '+BO' prototypes to include the new optimizations introduced in RCDP+BO, to improve the sending process, and to offer adequate support for additional optimizations; all these changes have a direct impact on the final CPU utilization.

We also evaluated the CPU usage of HTTP/TCP clients in our testbed in order to obtain reference values. We found that HTTP/TCP clients achieved a 30.67% of CPU usage, approximately. This value is only a bit lower than our basic RCDP client, which achieves a CPU usage of 41.72%. Moreover, we also found that software agents, such as *wget* only need around a 7% of CPU to perform their tasks for typical download speeds (up to 2 Mbps).

Focusing on RAM usage, Figure 4.11 shows that the data structures required to support parallel coding or decoding cause memory consumption to increase. In particular, RCDP+PED and RCDP+PEIT present the highest memory usage since both parallel coding and decoding are supported. RCDP+PEIT has the highest RAM requirements since it introduces an additional thread for coding functions, which has a clear impact on the memory resources consumed. Notice that, no matter which of the endpoints we focus on (client or server), both must perform coding and decoding tasks to fully support bidirectional communication.

Overall, results show that, despite both client and server share the same protocol architecture, the emphasis on either encoding or decoding tasks results in different behaviors. In particular, the impact of the different RCDP enhancements is quite heterogeneous in terms of both CPU and RAM utilization, being RCDP+PE and RCDP+PED the most resource consuming solutions both at the server and the client sides, respectively.

4.6.3 Performance comparison against different transport layer solutions

To complete our analysis, in this section we assess the effectiveness of the proposed design and implementation optimizations for RCDP against different

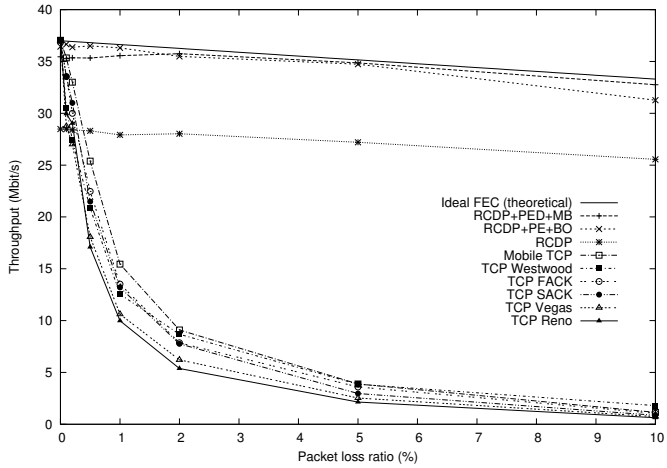


Figure 4.12: Throughput performance of RCDP compared to different TCP-based solutions.

versions of TCP.

In our tests we use well-known TCP variants (TCP Reno [36], TCP Vegas [18], TCP SACK [23], TCP FACK [24]) for reference, as well as other solutions like Mobile TCP [28] and TCP Westwood [9], which specifically address wireless channels by discriminating channel-related losses from congestion-related losses.

Figure 4.12 shows the results obtained in our testbed. The available network bandwidth is of 50 Mbps, and delay is set to 10ms. The α and β parameters for all RCDP versions are set to 0.7 and 0.9, respectively. A set of experiments were performed in order to select these α and β values, according to the configuration of RCDP. The ideal FEC line is calculated by assuming that the throughput is the product between the traffic injected and the packet arrival ratio, since that would be the best-case situation.

From Figure 4.12 we observe that, although offering different levels of error resilience, the different TCP-based solutions are quite sensitive to packet losses, mostly due to (i) the need of retransmitting lost packets, and (ii) using sliding windows of variable size to adjust the data rate in the presence of loss. Thus, when packet loss increases, the throughput associated with the different TCP variants drops quickly, experiencing a significant decrease compared to the *no loss* situation. Notice that the most significant differences between the

different TCP versions occur in the loss range from 0% to 1%, as expected. On the contrary to TCP, the throughput values remain mostly immune to loss for all RCDP versions, being sustained near the maximum value, and it only decreases by a 15% in the worst case. In fact, we find that both RCDP+PED+MB and RCDP+PE+BO present performance levels that are comparable to those of an ideal (theoretical) FEC solution. Figure 4.12 also shows that, compared to the original RCDP implementation, the chosen RCDP optimizations allow improving throughput by about 25%.

4.7 Conclusions

The delivery of large contents in wireless environments can be a complex task due to the different sources of loss inherent to these networks. In this chapter we proposed RCDP, a novel solution offering reliable unicast content delivery that operates at the application layer. To achieve high reliability and efficiency while avoiding packet retransmissions, RCDP relies on Raptor codes, an AL-FEC solution that is highly effective at recovering missing data, and that allows generating as many recovery packets as required.

When compared to transport layer solutions, we find that RCDP achieves much more stable throughput values in the presence of loss, approaching the performance of an ideal solution. In particular we find that, when the packet loss rate increases up to 10%, RCDP only experiences a throughput decrease of 15% in the worst case, while TCP based alternatives experience a throughput decrease of up to 95%.

Experimental results showed that parallelization of the encoding and decoding stages, along with the fine tuning of buffer sizes, represents different design alternatives, which tackle the different trade-offs between complexity and efficiency in order to maximize end-to-end throughput.

Overall, we consider that RCDP introduces a new paradigm in the field of wireless communications, being a very attractive solution for reliable content delivery in environments such as ad-hoc networks due to its highly efficient use of available bandwidth resources. Additionally, since it operates in an end-to-end basis, no changes to the network infrastructure are required.

As future work we plan to develop a version of RCDP for the OMNeT++ simulator in order to test RCDP's performance in vehicular network environments.

Acknowledgments

This work was partially supported by the *Ministerio de Ciencia e Innovación*, Spain, under Grant TIN2011-27543-C03-01 and by the *Ministerio de Educación*, Spain, under the FPU program, AP2010-4397.

Bibliography

[1] M. Luby, A. Shokrollahi, M. Watson, and T. Stockhammer. RaptorQ Forward Error Correction Scheme for Object Delivery. Internet Engineering Task Force, RFC6330, August 2011.

[2] N.F. Abdullah, A. Doufexi, and R.J. Piechocki. Car-to-car safety broadcast with interference using raptor codes. In *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, pages 1 –5, may 2011.

[3] N.F. Abdullah, A. Doufexi, and R.J. Piechocki. Raptor codes for infrastructure-to-vehicular broadcast services. In *Vehicular Technology Conference (VTC Fall), 2011 IEEE*, pages 1 –5, sept. 2011.

[4] N.F. Abdullah, A. Doufexi, and R.J. Piechocki. Multi-rate vehicular communications with systematic raptor codes in urban scenarios. In *Communications (ICC), 2012 IEEE International Conference on*, pages 7141 –7145, june 2012.

[5] H. Balakrishnan, V.N. Padmanabhan, S. Seshan, and R.H. Katz. A comparison of mechanisms for improving TCP performance over wireless links. *Networking, IEEE/ACM Transactions on*, 5(6):756–769, 1997.

[6] H. Balakrishnan, S. Seshan, E. Amir, and R.H. Katz. Improving TCP/IP performance over wireless networks. In *Proceedings of the 1st annual international conference on Mobile computing and networking*, pages 2–11. ACM, 1995.

[7] R.A.F. Bhoedjang, T. Ruhl, and H.E. Bal. User-level network interface protocols. *Computer*, 31(11):53–60, November 1998.

[8] K. Brown and S. Singh. M-TCP: TCP for mobile cellular networks. *ACM SIGCOMM Computer Communication Review*, 27(5):19–43, 1997.

[9] Claudio Casetti, Mario Gerla, Saverio Mascolo, M. Y. Sanadidi, and Ren Wang. TCP westwood: end-to-end congestion control for wired/wireless networks. *Wireless Networks*, 8:467–479, September 2002.

[10] P. Cataldi, M. Grangetto, T. Tillo, E. Magli, and G. Olmo. Sliding-window raptor codes for efficient scalable wireless video broadcasting with unequal loss protection. *Image Processing, IEEE Transactions on*, 19(6):1491 –1503, june 2010.

[11] R.S. Cheng, D.J. Deng, Y.M. Huang, L. Huang, and H.C. Chao. Cross-layer tcp with bitmap error recovery scheme in wireless ad hoc networks. *Telecommunication Systems*, 44(1):69–78, 2010.

[12] Hsin-Ta Chiao, Kuan-Ming Li, Hung-Min Sun, Shih-Ying Chang, and Hsin-An Hou. Application-Layer FEC for file delivery over the WiMAX unicast networks. In *Communication Technology (ICCT), 2010 12th IEEE International Conference on*, pages 685 –688, nov. 2010.

[13] Philipp M. Eittenberger. Raptorstream: boosting mobile peer-to-peer streaming with raptor codes. In *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, SIGCOMM '12, pages 291–292, New York, NY, USA, 2012. ACM.

[14] S.M. ElRakabawy and C. Lindemann. A practical adaptive pacing scheme for tcp in multihop wireless networks. *IEEE/ACM Transactions on Networking (TON)*, 19(4):975–988, 2011.

[15] B. Francis, V. Narasimhan, A. Nayak, and I. Stojmenovic. Techniques for enhancing tcp performance in wireless networks. In *Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on*, pages 222–230. IEEE, 2012.

[16] Y. Gu and R.L. Grossman. UDT: UDP-based data transfer for high-speed wide area networks. *Computer Networks*, 51(7):1777–1799, 2007.

[17] Peter Anthony Iannucci, Jonathan Perry, Hari Balakrishnan, and Devavrat Shah. No symbol left behind: a link-layer protocol for rateless codes. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, Mobicom '12, pages 17–28, New York, NY, USA, 2012. ACM.

[18] L. Brakmo, S. O'Malley, and L. Peterson. TCP Vegas: New technique for congestion detection and avoidance. In *Proceedings of ACM SIGCOMM'94*, August 1994.

[19] Uichin Lee, Joon-Sang Park, Joseph Yeh, Giovanni Pau, and Mario Gerla. Code torrent: content distribution using network coding in vanet. In *Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking*, MobiShare '06, pages 1–5, New York, NY, USA, 2006. ACM.

[20] Xinfeng Li, Jin Teng, Boying Zhang, A.C. Champion, and Dong Xuan. Turfcast: A service for controlling information dissemination in wireless networks. In *INFOCOM, 2012 Proceedings IEEE*, pages 298 –306, march 2012.

[21] C. Liu. Leading causes of tcp performance degradation over wireless links. *Embedded Software and Systems*, pages 494–505, 2005.

[22] M. Luby, M. Watson, T. Gasiba, T. Stockhammer, and Wen Xu. Raptor codes for reliable download delivery in wireless broadcast systems. In *Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE*, volume 1, pages 192 – 197, January 2006.

[23] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. RFC2018: TCP Selective Acknowledgement Options. *RFC Editor United States*, 1996.

[24] Matthew Mathis and Jamshid Mahdavi. Forward acknowledgement: refining tcp congestion control. *SIGCOMM Comput. Commun. Rev.*, 26:281–291, August 1996.

[25] T. Mladenov, S. Nooshabadi, and K. Kim. Implementation and evaluation of raptor codes on embedded systems. *Computers, IEEE Transactions on*, 60(12):1678 –1691, dec. 2011.

[26] S.A. Mondal. Improving performance of tcp over mobile wireless networks. *Wireless Networks*, 15(3):331–340, 2009.

[27] Sakib A. Mondal and Faisal B. Luqman. Improving tcp performance over wired-wireless networks. *Computer Networks*, 51(13):3799 – 3811, 2007.

[28] Muhammad Saeed Akbar, Syed Zubair Ahmed, and Muhammad Abdul Qadir. Performance Optimization of Transmission Control Protocol in Heterogeneous Wireless Network during Mobility. *IJCSNS International*

Journal of Computer Science and Network Security, 8(8):70–80, august 2008.

[29] C. Parsa. TULIP: A link-level protocol for improving TCP over wireless links. In *Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE*, pages 1253–1257. IEEE, 2002.

[30] R. Jain and S. A. Routhier. Packet trains - measurement and a new model for computer network traffic. *IEEE Journal on Selected Areas in Communications*, 4:986–995, September 1986.

[31] Wenyu Ren, Yong Li, Depeng Jin, Li Su, and Lieguang Zeng. Optimal vehicles and coding decision for mobile data sharing in vehicular delay tolerant networks. In *Network Protocols (ICNP), 2012 20th IEEE International Conference on*, pages 1 –2, 30 2012-nov. 2 2012.

[32] M. Sathiamoorthy, A.G. Dimakis, B. Krishnamachari, and Fan Bai. Distributed storage codes reduce latency in vehicular networks. In *INFOCOM, 2012 Proceedings IEEE*, pages 2646 –2650, march 2012.

[33] A. Shokrollahi. Raptor codes. *Information Theory, IEEE Transactions on*, 52(6):2551–2567, 2006.

[34] P. Sinha, T. Nandagopal, N. Venkitaraman, R. Sivakumar, and V. Bharghavan. WTCP: A reliable transport protocol for wireless wide-area networks. *Wireless Networks*, 8(2/3):301–316, 2002.

[35] C. Stefanovic, V. Stankovic, M. Stojakovic, and D. Vukobratovic. Raptor packets: A packet-centric approach to distributed raptor code design. In *Information Theory, 2009. ISIT 2009. IEEE International Symposium on*, pages 2336 –2340, 28 2009-july 3 2009.

[36] W. Stevens. TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. Internet Engineering Task Force, RFC 2001, January 1997.

Chapter 5

Assessing the impact of obstacle modeling accuracy on IEEE 802.11p based message dissemination

Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. *Assessing the impact of obstacle modeling accuracy on IEEE 802.11p based message dissemination.* In Mirela Sechi Moretti Annoni Notare and Abdelhamid Mammeri, editors, Proceedings of the third ACM interna- tional symposium on Design and analysis of intelligent vehicular networks and applications, DIVANet@MSWiM 2013, Barcelona, Spain, November 3-4, 2013 , pages 123128. ACM, 2013.

Abstract

Deploying real IEEE 802.11p vehicular network testbeds is a challenging but difficult option for most researchers. In these cases, the research community relies on simulation tools to test their protocols. However, since simulation accuracy is a critical issue, real testbed results should be used as a reference to improve simulation behavior. Our proposal adjusts common propagation models to mimic samples taken from real environments, and it uses a building aware model to achieve as much accuracy as possible in urban scenarios. We

evaluate the performance differences obtained with this model against other usual simulation schemes, like line-of-sight propagation models, models where no building blockage is taken into account, and models where propagation is only allowed along streets, achieving differences of up to 70% in some measurements. Going a step further, the model is used to study the radio propagation behavior along different city layouts, showing that the actual building layout is one of the key factors affecting protocol performance in urban environments.

5.1 Introduction

New devices compliant with the 802.11p standard are being developed by several hardware companies [1]. This enables researchers to start obtaining real performance results [3] and make them available for the research community [12]. There are also some current [9] and future [15] projects attempting to develop real testbeds to allow deploying software for performance testing. However, these initiatives are either too expensive, too closed or too ethereal to serve the purposes of many researchers. Therefore, they must rely on simulations to check the efficiency of their protocols.

One of the key pieces of a simulator is the propagation model, especially when addressing vehicular networks. There are some works that focus on this problem, such as Killat et al. [10], whose model adds vehicle traffic density to the Nakagami model; CORNER [7], which uses road topology to work out the received power accordingly; Hosseini Tabatabaei et al. [8], whose model enhances the two-ray-ground model to take into account reflections from buildings; Boban et al. [3], whose model is aware of non line-of-sight conditions to calculate the final power of a signal; Martinez et al. [11], whose hybrid model allows line-of-sight transmissions for two vehicles if they are in the same or adjacent streets, blocking signals otherwise; and Sommer et al. [14], whose hybrid model relies on obstacle information to heavily penalize any transmission traversing walls. The work of Cozzetti et. al. [4], who present RUG to improve current corner-based propagation models, and Scopigno et. al. [13], whose RADII uses a ray tracing technique to simulate the propagation for any building layout, are also relevant in this context. However, most of them are not aware of the building topology, they are not based on real measurements, or they are only focused on certain topologies or situations.

Once a simulator is realistic enough, we can close the circle. We can assess the effectiveness of any proposal in many different cities by first studying how a simpler solution performs in a specific scenario. Going one step further, we can study different simulations to determine the key factors affecting message

propagation in a city, such as road shapes, obstacles, road interconnections, etc. This offers researchers, designers and developers the most immediate knowledge of how their protocol will perform under certain conditions, allowing them to tune or modify it properly, or to work out a new solution.

In this chapter we tune common propagation models to produce results that resemble the real testbed measurements in [12]. This tuned model is combined with the buildings aware model presented in [14] to provide a model able to offer a realistic outcome, regardless of line-of-sight or non-line-of-sight conditions, realistic or synthetic road topologies, or highway or urban environments. After assessing its properties, the model is used to evaluate different city layouts in order to find the key factors that have significant effects in the final behavior of a protocol.

The chapter is organized as follows: in section 5.2 the proposed model is compared against simpler existing models to highlight the performance differences detected when simulating vehicular scenarios. In section 5.3 different cities are evaluated, and the factors affecting performance are identified. Finally, section 5.4 concludes the chapter.

5.2 Model tuning and performance assessment

In this section we extend the model presented in [2], adjusting the signal propagation model parameters according to real-life experiments. This propagation model is made of two different components. The first one deals with signal propagation in free space without or with small mobile obstacles, like cars. It is based on a Free Space model and a Nakagami model whose α (Free Space's attenuation coefficient) and m (Nakagami's shape factor) propagation parameters can be tuned in order to get a behavior as close to reality as possible. The second component takes advantage of the model by Sommer et. al. [14] to show a realistic behavior in the presence of buildings. Combining these two models, not only regular signal propagation situations are properly simulated, but also the presence of buildings is simulated accurately.

We take special care about α and m propagation parameters. Using this model, we will assess the impact of buildings and other obstacles on communications. These studies were performed using the same methodology described in our previous paper.

5.2.1 Channel model tuning

In this section we aim at replicating real life transmission conditions in 802.11p vehicular environments. We aim at tuning two different models. The first one is the *Vehicles as obstacles* model, which simulates common urban vehicular communication where mobile obstacles partly block the radio signal. The second one is the *Line of sight* model, which simulates the simplest transmission situation, where two vehicles can communicate when in line-of-sight.

As a reference for the first model we take the experimental results provided by Meireles et al. [12] for vehicle obstructed conditions. We establish the reference for the second model in order to get a relative reference for data concerning vehicles as obstacles. This reference is based on the line-of-sight model in [12], and the configuration presented in [5]. Notice that data presented in both works is compatible and complementary.

Our proposal combines the *Modified Free Space* model and the *Nakagami* model. In a first step we tune our model by adjusting the α and m parameters. In the simulated scenario used to adjust our model, two nodes are deployed respecting line-of-sight restrictions. We took different measurements of the received packet ratio by varying the distance between the nodes from one case to another.

Since both α and m parameters affect the received packet ratio, we simultaneously vary both values to get the best curve fitting. It was done for two both *Line of sight* and *Vehicles as obstacles* models.

Figure 5.1 shows the differences between several propagation models according to the value of the α and m parameters and the *Vehicles as obstacles* results from [12]. We first focus on the m values since they affect both the slope and the packet delivery ratio. The optimal curve slope corresponds to a value of 0.7 for the m parameter because smaller values cause the slope to be too low, as can be seen in the figure. Focusing on the value of α, we find that the value offering good accuracy in terms of packet delivery ratio is 2.3. As can be seen, there is a minimal error between the model and data in [12]: the squared error is lower than 0.006. We find that the reliable communication range (>90% of packet delivery ratio) of 100m obtained in [12] is also maintained.

Figure 5.2 shows the differences between several propagation models according to the value of both α and m parameters based on the line-of-sight results. This model will be used as reference in following the subsections that follow. If we focus on slope, values in the range from 1 to 1.1 offer a similar fading behavior. For the α parameter, the value of 2.2 is the best possible fit. Finally, we can see that the reliable communication range in [12] is main-

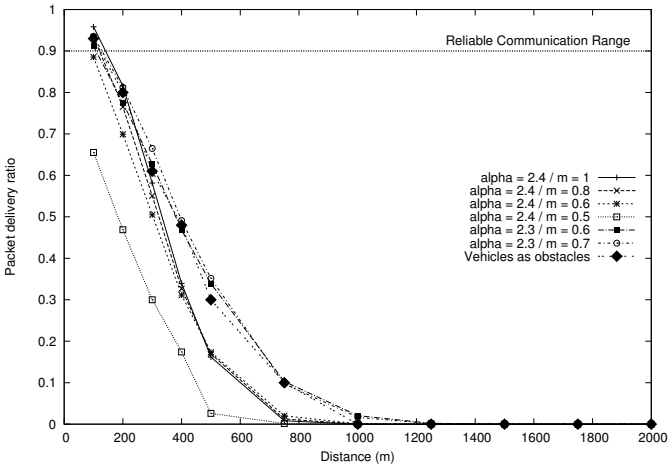


Figure 5.1: Similarity of *Vehicles as obstacles* model and various α and m configurations

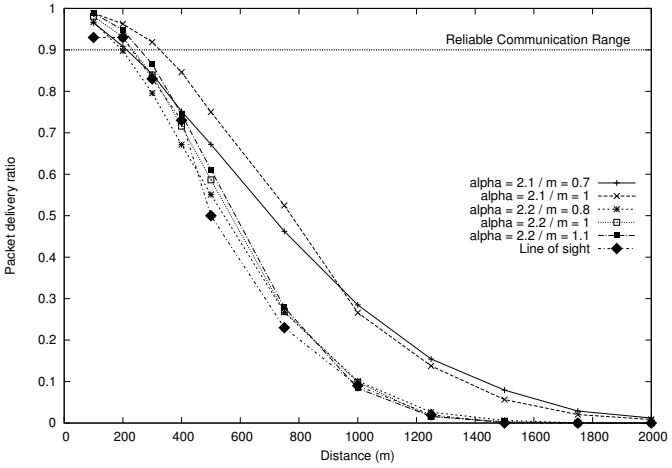


Figure 5.2: Similarity of *Line of sight* model and various α and m configurations

tained, and that this model is also consistent with the reference value for the maximum transmission range defined in [5], with a packet delivery ratio lower than 5% at 1400m.

5.2.2 Assessing the performance impact of static and mobile obstacles

After adjusting the propagation model parameters in the previous section, and considering either line-of-sight or vehicle obstructions for the signal propagation, we now present a new set of experiments to evaluate two main features: (i) the impact of mobile obstacles (vehicles), and (ii) the impact of static obstacles (buildings) on message delivery. To evaluate these features we used OMNeT++ combined with Sumo to simulate a scenario where 500 cars move following random routes within a $12km^2$ area in the peripheral area of Moscow.

The propagation models have been tested in three different scenarios with different building layouts. The first one is the *No buildings* scenario, where the inexistence of buildings is assumed. The second one is the *Real buildings* scenario, where the real building layout, as defined in the OpenStreetMap database, is used. The third one is the *Synthetic buildings* scenario, where the building layout is such that radio transmissions only become possible along the roads. This scenario was included because this is a common assumption in current research.

In terms of traffic generation, all cars broadcast packets at regular intervals of 10s, and we consider two cases: (i) broadcasts are limited to one hop; and (ii) broadcasts are rebroadcasted by other vehicles (flooding).

5.2.2.1 Impact of obstacles

Figure 5.3 shows the average percentage of cars that are reachable in a one-hop beaconing process, with different obstacle models, in all scenarios. We can find a reduction in the *Vehicles as obstacles* model near to one third over the *Line-of-sight* model in the worst case. These differences will be a key factor in one-hop beaconing protocols, not only due to the final amount of neighbors detected, but also due to the quality of these links and the possible forwarders in a routing protocol algorithm.

This figure also reveals that the final vehicle reachability ratio highly depends on the existing buildings. Since Moscow is a city with a low building density, a simulation in a highly restrictive configuration will provoke a higher error degree compared to a less restrictive one. We have the opposite problem in a high building density scenario. Simulations with weak restrictions also

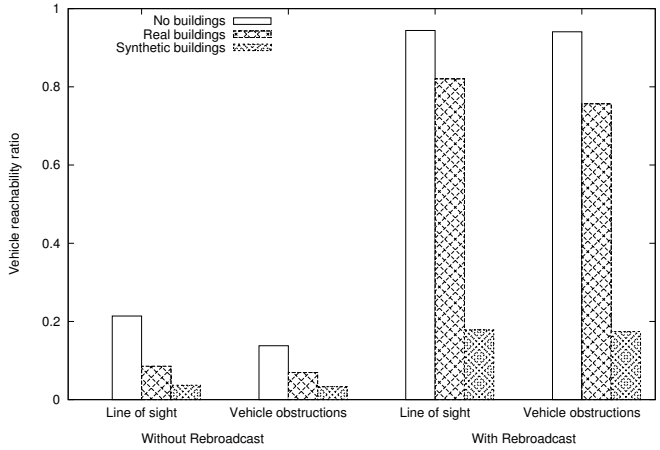


Figure 5.3: Impact of static obstacles under different conditions

give us a high error degree. To avoid these problems, accurate building maps are needed.

Focusing on the broadcasting strategy, we can see that it evidences the effect of static obstacles because it amplifies its effects. With no rebroadcasting, only the nodes near the sender are reached (1 hop neighbors). However, in a multi-hop approach, the building obstruction effect is amplified at every hop, making it more evident. Results show that, except for the synthetic obstacles scenario, the flooding process is successful, reaching the majority of the vehicles circulating in the target area, as desirable.

5.2.2.2 Delay study

To analyze the flooding evolution in more detail, figure 5.4 shows the vehicle reachability behavior. We can observe here how the values presented in figure 5.3 are achieved along time, allowing to understand how the propagation model and the presence of static obstacles influences the perceived delay.

Observing the differences between the *Line-of-sight* model and the *Vehicles as obstructions* model, we see how a lower transmission radius causes the *Vehicles as obstacles* model to be always below the *Line-of-sight* model at any given time. The greatest difference occurs for the real building model, where buildings allow a moderate car reachability, as stated in previous subsections. Here we can find differences greater than half a millisecond for a car

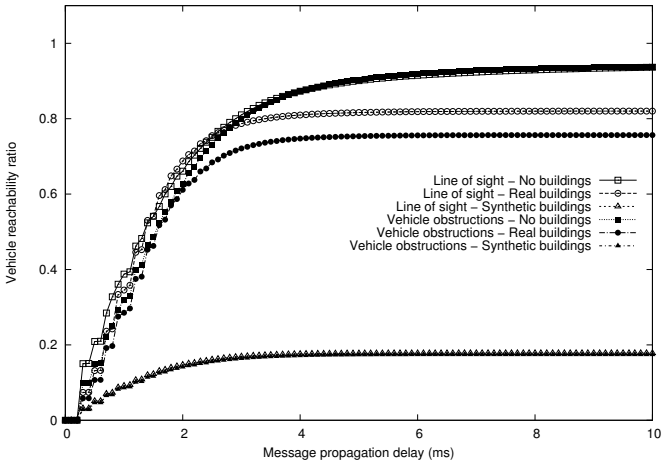


Figure 5.4: Vehicle reachability behavior along time

reachability of about 70%.

However, the main impact in terms of performance is caused by the static obstacles factor. We can see clearly the delay increment that different static obstacles add when focusing on synthetic and real buildings. Not only a lower percentage of cars is reached, but also they are reached with a higher temporal cost. Therefore, we can conclude that an excessive or an insufficient building obstructiveness will lead our experiments to a variable error in terms of delay. In the following section we will examine other cities with different buildings densities to quantify how different values for this parameter can vary the final outcome shown by each of them. Since cities are a very complex environment, each having different road layouts and other parameters that can affect our results, in the sections that follow we will try to determine which are the main parameters affecting performance, isolating the effects of different building layouts.

5.3 Evaluation of different city profiles

As stated before, buildings block signal propagation causing transmission and connectivity failures. Since different cities present different building densities and layouts, every city will present a different connectivity degree. In this section we aim at evaluating this connectivity degree as a critical factor that

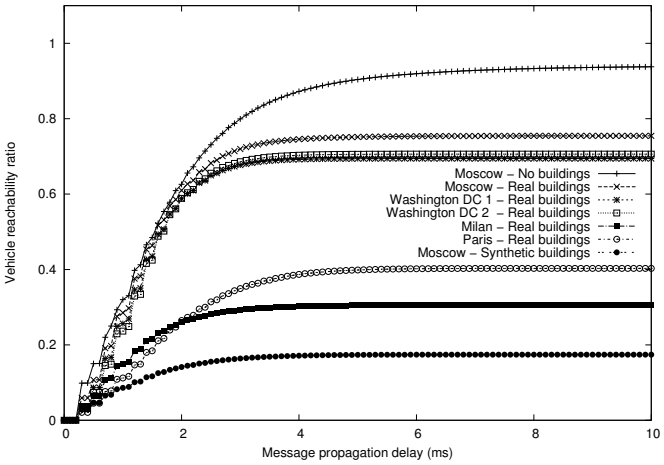


Figure 5.5: Vehicle reachability behavior along time for different city layouts

could affect the final performance.

5.3.1 Cities comparison

We first selected a few cities to be used as a representative sample. The information for all the cities was extracted from the OpenStreetMap database. Despite correspondence to reality is not fully assured, the realism of these samples is much higher than what synthetic models can produce.

The test scenarios include residential areas, such as the Moscow maps presented before; grid areas, such as Washington DC; and urban canyon areas, such as Milan and Paris.

The conditions of the experiments are the same as above: we used OMNeT++ combined with Sumo to simulate a scenario where 500 cars move by following random routes. In terms of traffic generation, all cars broadcast packets at regular intervals of 10s, and we consider the case where broadcasts are rebroadcasted by other vehicles (flooding). We tune our attenuation model with the parameters found according to the *Vehicles as obstacles* paradigm.

Figure 5.5 shows the vehicle reachability behavior along time for different city layouts. We also include there two reference series. The first of them is the Moscow version with no buildings. This curve shows a city where the impact of static obstacles is not taken into account. If all road layouts of all

cities were the same, this would be the maximum value of reachability that a city may achieve. However, there are a lot of factors that also affect the final behavior, so it must only be used as a reference, and not as an upper bound. We also include a Moscow version using synthetic buildings in order to obtain a highly restrictive version, almost unreal, of the maximum obstacle blockage level. Similarly to the former one, this is used as a reference alone.

Looking at results we can see that different cities have different outcomes. In the figure, two groups are evident. The lower group is composed by cities which we intuitively associate to high density building areas, as can be our synthetic layout or the real downtown of European cities. However, the upper group, constituted by the Moscow and Washington cases, is more heterogeneous. The construction intensity varies significantly: from low density scenarios (Moscow scenario), and residential scenarios (first Washington scenario) to more dense scenarios (second Washington scenario).

This variable effect shown in different cases takes place because there are many factors that impact the final transmission effectiveness. Transmissions along roads are possible and relevant, in addition to transmissions to roads different from the source one. Therefore, not only obstacle density must be taken into account, but also the actual city road layout is important.

In the following subsection we will examine different factors to isolate the effect of building and road layout for our data set.

5.3.2 Cities profiling and result analysis

Fogue et al. [6], investigated some factors related to city street layouts, assessing their impact. They have identified a trend which relates message dissemination performance to the road layout and the vehicular density. Since in our analysis we introduced factors related to signal blockage by buildings, we aim at identifying a dependence in the performance of our sample of two main components: one of them will be related to the road layout, and another one will be related to the building layout. These two factors must define the trend in our experiments in order to effectively check the influence of obstacles in the final performance.

In order to isolate the effects that different building layouts have in the performance of our sample, we will examine all the cities that we have simulated in the previous subsection. We will examine the city maps as extracted from OpenStreetMaps to create a profile for each city, and identify a trend that ties the results of our simulations with these static measurements.

Table 5.1 shows some of the static measurements made over the OpenStreetMap definition files. Since obstacle blockage depends on building size,

Table 5.1: City profiles

(a) Set one

	Moscow / No buildings	Moscow	Washington 1
Latitude	55º32'N	55º32'N	38º56'N
Longitude	37º31'E	37º31'E	77º4'W
Number of junctions/km2	59.50	59.50	66.43
Total street length (km)	107.11	107.11	62.42
Average street length (m)	199.84	199.84	262.27
Obstacle coefficient	0.8398	0.5492	0.3478

(b) Set two

	Washington 2	Milan	Paris	Moscow / Synthetic obstacles
Latitude	38º54'N	45º27'N	48º50'N	55º32'N
Longitude	77º2'W	9º11'E	2º19'E	37º31'E
Number of junctions/km2	104.64	189.29	269.75	59.50
Total street length (km)	131.70	127.95	198.82	107.11
Average street length (m)	195.41	111.16	104.92	199.84
Obstacle coefficient	0.5111	0.4809	0.5412	0.2004

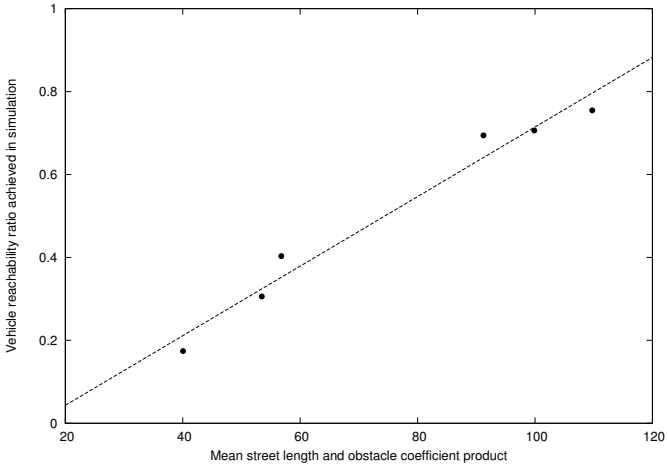


Figure 5.6: Combined street length and obstacle coefficient trend in our sample with lineal regression.

building position, and other streets availability, we create a metric, called *obstacle coefficient*, that tries to group all these factors in a single a priori value. This value is the mean number of rays that, starting from a street, can reach a different one. We calculate it by defining random points in the street map, and checking whether streets different from the source one can be reached. With this coefficient we aim at providing a statistical approach of the building, as well as density and the blockage probability of a transmission to a vehicle in other streets.

To obtain a definitive metric, we propose selecting the average street length for the streets layout component, and the obstacle coefficient for the building layout component. The average street length is a good choice since it is directly proportional to the number of vehicles that are reachable along that street. In addition, a city density measure is also taken into account since smaller roads are usually related to a more dense and meandering city, and vice versa. This value is the ideal complement for our obstacle coefficient, only taking into account transmissions to different streets from the source one.

Figure 5.6 shows the trend that both the average street length and the obstacle coefficient form together. We can see that, for our sample, the street length and the obstacle coefficient are tied to the final performance in a linear way. Since there are a lot of factors that can affect final performance, a

generalization of these effects is difficult to achieve, but it becomes clear that the building coefficient is a factor that must be taken into account in city profiling.

Therefore, we can see that different building layouts affect the message propagation performance for very different urban environments, even if they are masked by other factors like road length. Although a complete understanding of how many different city factors actually have an impact on final performance is still lacking, it is worth getting an a priori knowledge of the performance that a city will achieve in a broadcast process. This information will allow us to select the aggressiveness, understood as number of rebroadcasting nodes, the number and periodicity of duplicated alerts, the number of RSU nodes in a DTN process, etc., required by broadcast protocols to provide effective data diffusion.

5.4 Conclusions

In this chapter we rely on empirical results from real vehicular testbeds to adjust a propagation model based on the *Free Space* model, the *Nakagami* model, and an obstacle model to achieve more accurate performance results in our simulations. We evaluated the impact of this new model against other usual simulation schemes, like line-of-sight propagation models, models where no building blockage is taken into account, and models where propagation is only allowed along streets. We detected significant differences between all models in terms of total vehicle reachability and delay in a broadcast process, with differences greater than 75% in the total number of reached cars.

We also checked how the obstacle model affects performance for different city profiles. We found that its influence, as a building layout component, may be presented jointly with a road layout component. We found, for our sample, that a good candidate for this combination is the street length.

As future work, we want to further explore the trend we detected to obtain a general relationship between the different city layout parameters, thus obtaining a city profiling system that is as accurate as possible, being able to get a priori knowledge of how a protocol will perform in a city. We also want to study the impact of car density on performance.

Acknowledgements

This work was partially supported by the *Ministerio de Ciencia e Innovación*, Spain, under Grant TIN2011-27543-C03-01, and by the *Ministerio de Educación*, Spain, under the FPU program, AP2010-4397.

Bibliography

[1] Mobiwave - on-board equipment.

[2] M. Baguena, C.T. Calafate, J. Cano, and P. Manzoni. Towards realistic vehicular network simulation models. In *Wireless Days (WD), 2012 IFIP*, pages 1–3, 2012.

[3] M. Boban, T.T.V. Vinhoza, M. Ferreira, J. Barros, and O.K. Tonguz. Impact of vehicles as obstacles in vehicular ad hoc networks. *Selected Areas in Communications, IEEE Journal on*, 29(1):15–28, 2011.

[4] H.A. Cozzetti, C. Campolo, R. Scopigno, and A. Molinaro. Urban vanets and hidden terminals: Evaluation through a realistic urban grid propagation model. In *Vehicular Electronics and Safety (ICVES), 2012 IEEE International Conference on*, pages 93–98, 2012.

[5] David Eckhoff, Christoph Sommer, and Falko Dressler. On the Necessity of Accurate IEEE 802.11p Models for IVC Protocol Simulation. In *75th IEEE Vehicular Technology Conference (VTC2012-Spring)*, Yokohama, Japan, May 2012. IEEE.

[6] M. Fogue, P. Garrido, F.J. Martinez, J.C. Cano, C.T. Calafate, and P. Manzoni. Pawds: A roadmap profile-driven adaptive system for alert dissemination in vanets. In *Network Computing and Applications (NCA), 2011 10th IEEE International Symposium on*, pages 1–8. IEEE, 2011.

[7] E. Giordano, R. Frank, G. Pau, and M. Gerla. Corner: a step towards realistic simulations for vanet. In *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking*, pages 41–50. ACM, 2010.

[8] S.A. Hosseini Tabatabaei, M. Fleury, N.N. Qadri, and M. Ghanbari. Improving propagation modeling in urban environments for vehicular ad hoc networks. *Intelligent Transportation Systems, IEEE Transactions on*, 12(3):705–716, 2011.

[9] H.Y. Huang, P.E. Luo, M. Li, D. Li, X. Li, W. Shu, and M.Y. Wu. Performance evaluation of suvnet with real-time traffic data. *Vehicular Technology, IEEE Transactions on*, 56(6):3381–3396, 2007.

[10] M. Killat and H. Hartenstein. An empirical model for probability of packet reception in vehicular ad hoc networks. *EURASIP Journal on Wireless Communications and Networking*, 2009:4, 2009.

[11] F.J. Martinez, C.K. Toh, J.C. Cano, C.T. Calafate, and P. Manzoni. Realistic radio propagation models (rpms) for vanet simulations. In *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE*, pages 1–6. Ieee, 2009.

[12] R. Meireles, M. Boban, P. Steenkiste, O. Tonguz, and J. Barros. Experimental study on the impact of vehicular obstructions in vanets. In *Vehicular Networking Conference (VNC), 2010 IEEE*, pages 338–345. IEEE, 2010.

[13] R. Scopigno, H.A. Cozzetti, L. Pilosu, and F. Fileppo. Advances in the analysis of urban vanets: Scalable integration of radii in a network simulator. In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2012 IEEE 8th International Conference on*, pages 563–570, 2012.

[14] C. Sommer, D. Eckhoff, R. German, and F. Dressler. A computationally inexpensive empirical model of ieee 802.11 p radio shadowing in urban environments. In *Wireless On-Demand Network Systems and Services (WONS), 2011 Eighth International Conference on*, pages 84–90. IEEE, 2011.

[15] H. Stubing, M. Bechler, D. Heussner, T. May, I. Radusch, H. Rechner, and P. Vogel. simtd: a car-to-x system architecture for field operational tests [topics in automotive networking]. *Communications Magazine, IEEE*, 48(5):148–154, 2010.

Chapter 6

VACaMobil: VANET Car Mobility Manager for OMNeT++

Miguel Baguena, Sergio Martínez Tornell, Alvaro Torres, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. *VACaMobil: VANET car mobility manager for omnet++.* In IEEE International Conference on Communications, ICC 2013, Budapest, Hungary, June 9-13, 2013, Workshops Proceedings , pages 10571061. IEEE, 2013.

Abstract

The performance of communication protocols in vehicular networks highly depends on the mobility pattern. Therefore, one of the most important issues when simulating this kind of protocols is how to properly model vehicular mobility. In this chapter we present VACaMobil, a VANET Car Mobility Manager for the OMNeT++ simulator which allows researchers to completely define vehicular mobility by setting the desired average number of vehicles along with its upper and lower bounds. We compare VACaMobil against other common methods employed to generate vehicular mobility. Results clearly show the advantages of the VACaMobil tool when distributing vehicles in a real scenario, becoming one of the best mobility generators to evaluate the performance of different communication protocols and algorithms in VANET

environments.

6.1 Introduction

The reproducibility of experiments is a major issue when evaluating smart communication protocols and algorithms, especially over Vehicular Ad-hoc NETworks (VANETs). In [6] the authors provide a complete review of the minimum set of parameters that should be identified in order to allow other researchers to reproduce simulation experiments. They pointed out several key parameters, such as the simulated hardware, the network simulator, the scenario, and the road traffic simulator. However, regarding node mobility, there are other parameters that have been mostly ignored by the research community: the density of traffic and the traffic demand.

As other authors pointed out in previous studies, mobility models [10] and the chosen scenario [3], as well as the node density, heavily influence the final network performance. However, since mobility generators and road traffic simulators are often tough to configure, the simulated node density and distribution may depend on complex data that are usually not included in the published academic results, thereby compromising reproducibility.

In this chapter we present VACaMobil (VANET Car Mobility manager), a mobility manager module for the OMNeT++ simulator which is the first, to the best of our knowledge, able to generate SUMO [1] driven nodes in a vehicular network while ensuring certain user-defined parameters, such as the average, maximum, and minimum number of vehicles. This goal is useful for mid-length simulations, typically one hour, allowing researchers to assume that the vehicle density is stable. At the same time, since our solution is tightly coupled with SUMO through the TraCI interface, it is able to mimic real vehicle behavior. By running in parallel with SUMO, VACaMobil executes the following tasks: (i) it manages when a new vehicle must be introduced in the network, (ii) it assigns a random route from a predefined set to each vehicle, and (iii) it determines which type of vehicle should be added. When using VACaMobil, and given a specific road map, researchers will be able to completely define the network mobility merely by defining the desired average number of vehicles and its standard deviation value (upper and lower bounds).

Going a step further, our tool also aids researchers at selecting among the different types of vehicles previously defined in SUMO, such as “cars”, “buses”, or “trucks”. This allows researchers to easily define road traffic simulations with heterogeneous vehicles.

The rest of this chapter is organized as follows: In section 6.2, we shortly

introduce the different methods for generating VANET mobility patterns that the research community commonly uses. In section 6.3, VACaMobil is fully described. In section 6.4, we compare our proposal with the duarouter and duaIterate.py tools, both included in SUMO. Finally, in section 6.5, we present our conclusions and some future plans to improve VACaMobil.

6.2 A review of existing mobility generators for VANETs

Before presenting the details of our proposal, we analyze some of the methods commonly used to obtain suitable mobility patterns in urban vehicular scenarios. We have analyzed several papers published during the last few years, most of them published in conferences and journals related to Intelligent Transportation Systems. Early approaches relied on too simple mobility models based merely on random mobility. Since these simple models do not represent vehicle mobility properly, other mobility models have been recently developed based on real world traces and also on artificial mobility models from the field of transportation and traffic science. In this section, we briefly describe the most relevant works.

6.2.1 Random Vehicle Movement

At the beginning of the previous decade, the “Random Way-Point" was extensively used in Mobility Ad-Hoc NETwork (MANET) research. However, in 2003, the authors in [15] demonstrated how harmful the Random Way-Point mobility model really is. Moreover, the effects described in this work are even worse when simulating VANETs. Later on, some other authors have extended the “Random Way-Point" mobility model by restricting the mobility of nodes to a map layout, as in [14]. However, this improvement does not solve the majority of the “Random Way-Point” model problems stated previously.

In our research group we developed a tool called “CityMob" [9]. CityMob allows users to create random mobility patters restricted to a grid. It also adds support for *downtown* definition, where a *downtown* is a region inside the simulated map which concentrates the majority of the selected routes along the simulation. Although CityMob presents a big improvement compared to non restricted mobility models, as well as random mobility models, it also presents some problems; the most important one is that vehicular mobility is not influenced by other vehicles, *i.e.* two different vehicles can occupy the same location and no minimal distance between vehicles is required. Moreover,

vehicles do not change their speed during a trip. However, in the real world, vehicles continuously change their speed according to traffic conditions and road characteristics. Last but not least, vehicles keep moving throughout the whole simulation, which especially influences the performance of protocols that keep data stored in buffers. The research community quickly realized the problems derived from inaccurate simulation patterns, and started to work in other methods to obtain suitable mobility traces.

6.2.2 Real Mobility Traces

Compared to the use of random mobility, real traces present a clear improvement. Such traces are usually obtained from a certain set of nodes, e.g. from taxis in the city of Shangai [7]. Mobility traces can be obtained by tracking the mobility of nodes using On-Board units, as in [7], or by using road-side equipment, as in [5]. Although real traces represent the most realistic mobility patterns, we can not obviate the fact that the mobility of the tracked nodes is highly influenced by the movement of other non tracked vehicles, *e.g.* taxis' mobility is influenced by other users on the road whose movement is not reflected in the collected traces. Moreover, real traces lack the flexibility to allow for an exhaustive evaluation of VANET protocols, *e.g.* changing the vehicle density without modifying their speed is clearly unreal.

6.2.3 Assisted Traffic Simulation

The restrictions of real traces can be overcome, with almost no loss of realism, by using mobility models taken from the field of transportation and traffic science. Several road traffic simulators are widely used among the VANET research community. One of the most widely used mobility generators is SUMO [1]. When simulating traffic mobility for VANETs not only the vehicles' behavior is important, but also the traffic demand. SUMO allows defining traffic demand in two different ways: trips and flows. The former defines only a vehicle, its origin and its destination, while the latter defines a set of vehicles which execute the same trip. SUMO currently provides several tools to generate traffic demand:

- randomTrips.py: A random trip generator. This tool generates a trip every second having a random origin and destination. It does not check if the origin and destination are connected, or whether the trip is possible.
- duarouter: A Dijkstra router. Given a file with trips and flows, this tool generates the actual traffic demand, expressed in vehicles with an

assigned route. Routes are calculated using the Dijkstra algorithm, and every unconnected trip is discarded.

- duaIterate.py: This python script will produce a set of optimal routes from a trip file, *i.e.* all the nodes will follow that route which minimizes the total trip-time for all nodes. This tool repeats a routing-simulation loop until optimal routes are found.

Authors have used these tools in order to generate traffic demands for SUMO. The most simplistic one is to define different flows inside the network. Although drivers usually move from certain districts to others, following patterns associated with their working and living places, defining the traffic only by creating fixed flows lacks of any realism, as we can see in [2] where only a few flows are defined by the user. Another common approach is to generate random trips using *randomTrips.py*. This approach presents the problem that only one vehicle is introduced every second, which leads to long transitory periods until the network reaches a stable state. A more sophisticated traffic demand generation strategy is presented in [8], where a predefined number of vehicles following random routes are randomly placed at the beginning of the simulation. Following this trend, in previous works we used C4R [4], which is a software developed by our group to automate the task of generating random vehicles with random routes at random places. The work presented in [13] is the only one that we could find which uses the *duaIterate.py* script to generate a "stable and optimal distribution of flows". This type of traffic definition presents a problem: the trip duration can not be predicted before running the simulations, and, as a consequence, there is no way to ensure, or even determine, if the road traffic simulation will last until the end of the network simulation. As some works have stated before, this lack of realism and generality in mobility patterns can lead to biased results [10].

6.2.4 Bidirectionally coupled network and traffic simulations

In [11] its authors go a step further and present a new simulation framework called Veins, which includes the TraCI interface to allow the network simulator to interact with the traffic simulator running in parallel. Although, it presents much novelty and opens a lot of possibilities for VANET simulation, authors do not address the traffic demand generation problem. This framework demonstrated its new characteristics in [12], and it is one of the main elements of our VACaMobil module, allowing us to interact with SUMO during the network simulation and create new vehicles.

6.3 VACaMobil Mobility Manager

In this section we present our VANET mobility generator providing the main characteristics and its implementation details.

6.3.1 Characteristics

The main characteristic of VACaMobil is to offer realistic mobility scenarios. To that end it can guarantee an average number of vehicles while keeping the current number of vehicles within the given upper and lower bounds, the latter being associated to the defined standard deviation value. These features allowed creating a tool that ensures repeatability of network simulations under the same road traffic conditions just by defining the average number of vehicles and the standard deviation value.

Another characteristic it offers is the possibility of introducing types of vehicles in simulations, such as “car”, “bus” and “truck”, each one with its own characteristics. This is an important feature because high level decisions may be based on the type of the vehicle.

VACaMobil also provides a realistic vehicle distribution based on a list of different predefined routes.

Finally it works on-line with the SUMO traffic simulator, obtaining all the needed information about routes and type of vehicles through the TraCI communication interface, thereby avoiding the duplicity of configuration files.

6.3.2 Implementation details

This tool extends the module collection available in the Veins framework [11] with new capabilities. We explain, for each of the characteristics described before, the different implementation decisions taken.

6.3.2.1 Average number of vehicles

Figure 6.1 shows the VACaMobil iterative control loop. At every step of the mobility simulation, VACaMobil compares the current number of vehicles in the simulation with the target number of vehicles. Depending on whether it is greater or lower than the target value, VACaMobil waits until the number of vehicles decreases towards the target value, or, in the second case, starts inserting several new vehicles in every step of the mobility simulation in order to increase the current value until the desired value is achieved. To avoid having an average number of vehicles higher than the one defined by the user,

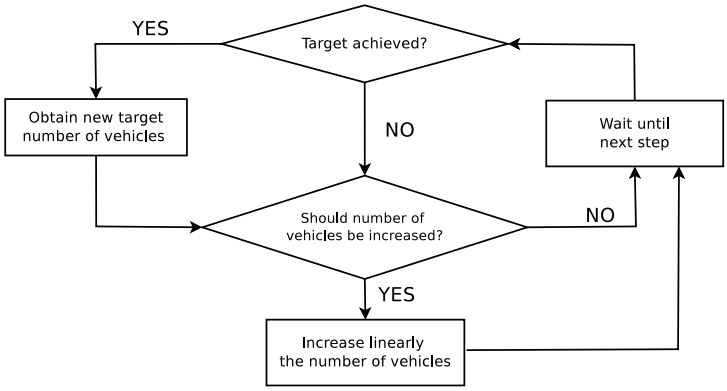


Figure 6.1: Main loop of the VACaMobil tool.

the time during which new vehicles are inserted is as long as the last period where the number of vehicles has decreased. By following this approach we also avoid high frequency fluctuations in the total number of vehicles throughout the simulation time.

Values for the *target number of vehicles* variable are obtained from a normal distribution whose mean is the desired average number of vehicles, and whose standard deviation is defined by the user. Its value is limited to the upper and lower bounds, which are defined as the $mean \pm 3 * standard\,deviation$: this value avoids extremely high or extremely low values for the current number of vehicles.

6.3.2.2 Different types of vehicles

One of the parameters we can obtain via TraCI indicates the types of vehicles that are available in the traffic simulation. The user can set different probabilities associated to each vehicle type. In this case, every time a new car is generated, we obtain a uniform random value and select the correspondent vehicle type. If no probability is defined for a certain type of vehicles, we assume it is equal to 0. However, if no probability value is assigned to any of the defined types of vehicle, only vehicles of the first defined type will be generated.

6.3.2.3 Routing set and vehicle distribution

Since SUMO itself loads all the different routes at startup, we can also retrieve them through TraCI, and, as in the previous item, we select one of them with

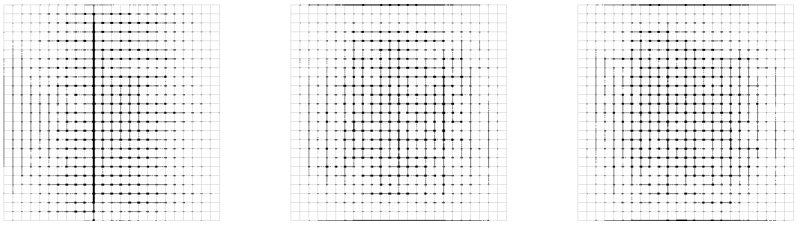


Figure 6.2: Heat map for the Manhattan scenario when using duarouter, *duaIterate.py*, and VACaMobil (from left to right).

Figure 6.3: Heat map for the urban scenario when using duarouter, *duaIterate.py*, and VACaMobil (from left to right).

an uniform probability every time we generate a new vehicle.

VACaMobil does not compute routes at simulation time; instead it relies on the goodness of the different routes made available by SUMO. To guarantee that vehicles are distributed realistically, we also developed a tool based on *duaIterate.py* and *randomTrips.py* which creates a SUMO route file with several random routes.

Finally, to ensure that a new vehicle is correctly added, the default behavior is to attempt to insert the vehicle at any of the lanes available on the first edge of the route. If the edge is full, the module selects a new route sequentially from a list, repeating the operation until it finds a free place to insert the vehicle, or until the first selected route is selected again, which means that there is no room on the road map for the new vehicle.

6.4 Evaluation

In this section we evaluate VACaMobil to verify whether the objectives and characteristics described in section 6.3.1 have been accomplished. In order to do that, we have compared VACaMobil against the tools currently included in

SUMO, *i.e.* duarouter and dualterate.py, that were described in section 6.2. We have selected the following scenarios:

- **Synthetic Manhattan scenario:** We created a road map consisting of a 25 x 25 grid with segments of 200 meters (Figure 6.2).

- **Urban real map scenario:** We extracted an urban road layout from the OpenStreetMap database. It is a scenario of about 7 km2 from the city of Milano characterized by short road segments and a high road density (Figure6.3).

In both scenarios, the set of random routes provided by VACaMobil is extracted from the traffic demand generated by dualterate.py. In the following subsection, we compare the vehicle density and its evolution along the simulation time for the aforementioned tools and scenarios.

6.4.1 Vehicle distribution study

We first evaluate one of the most important issues in vehicular mobility: how vehicles are distributed over the simulated road map.

Figure 6.2 shows how the compared methods perform in the Manhattan scenario. Due to its lack of randomness, duarouter is unable to select different routes for vehicles when there are several streets with the same travel-time. This prevents the simulator to properly distribute vehicles, and so all them are routed through the same street (eleventh from the left). When using the *dualterate.py* script, a better distribution of the vehicles is achieved due to the fact that many simulations are sequentially executed to optimize vehicle routes. Since the VACaMobil's random routes set is obtained from dualterate.py, it achieves a similar nodes distribution.

Figure 6.3 shows performance results for the urban scenario. In this case, duarouter is also unable to spread the vehicles properly. Since some roads are faster than others, all the vehicles are routed through them, even when these streets are congested. Therefore, an undesired traffic congestion is created in the fastest inner roads. However, this is an unrealistic scenario because drivers tend to avoid traffic jams whenever possible. When either using *dualterate.py* or VACaMobil, vehicles are routed through alternative streets, avoiding traffic jams. This strategy has a higher degree of similitude compared to real road traffic, since drivers prefer faster roads but often change their route to avoid traffic jams.

Table 6.1: VACaMobil configuration

	Vehicle number	Std. dev.
Manhattan	320	6
Urban scenario	370	8

Table 6.2: Vehicle statistics summary

	Manhattan		Urban scenario	
	mean	std. dev.	mean	std. dev.
duarouter	313.767	58.8271	880.546	465.716
duaIterate.py	304.487	55.5174	393.717	96.414
VACaMobil	319.349	6.14267	369.691	7.84640

6.4.2 Vehicle density study

To make simulations more easily comparable, a similar traffic density is desirable in all simulated city layouts. Current tools can not correctly handle this problem. In order to compare the behavior of the three selected methods previously presented, we have measured the average number of vehicles, its standard deviation, and its evolution along simulation time.

Table 6.2 shows the differences in terms of number of vehicles for the two target scenarios for each traffic generation tool. In the simplest scenarios (Manhattan), the three methods can achieve a stable value for the mean vehicle density with a low standard deviation. However, neither duarouter nor duaIterate allow to *a priori* configure the value of this parameter. On the contrary, VACaMobil is not only able to populate the network with the desired number of vehicles, but also allows defining a maximum and a minimum number of vehicles by using the standard deviation feature, which will bound the number of vehicles. In complex maps like the urban scenario, VACaMobil is the only tool able to maintain the standard deviation value within the predefined bounds.

To better understand the aforementioned values, figures 6.4, and 6.5 show the number of vehicles in the scenario along time for each tool. Since *duarouter* and *duaIterate.py* are only able to add one vehicle per second, the user cannot predict when vehicles will arrive to their destination and disappear from the network. Therefore, the number of vehicles when the simulation reaches the steady-state for the Manhattan scenario is not known *a priori*, converting protocol analysis based on number of vehicles in a mere act of faith. Moreover, in urban maps where traffic jams are very common, as in the urban scenario, it

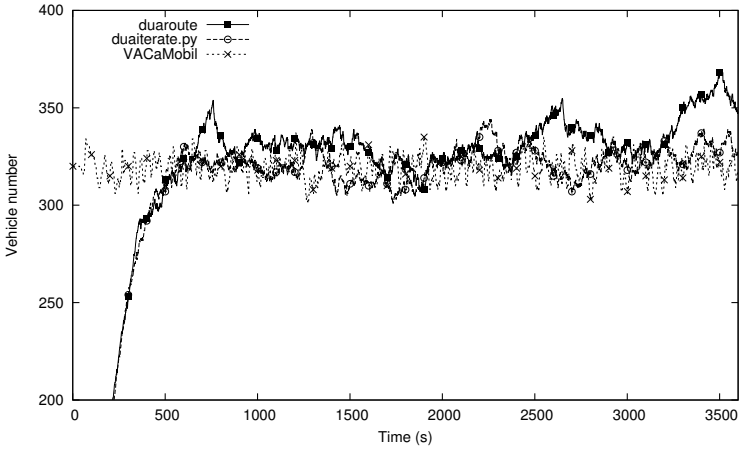


Figure 6.4: Vehicle number evolution for the Manhattan scenario.

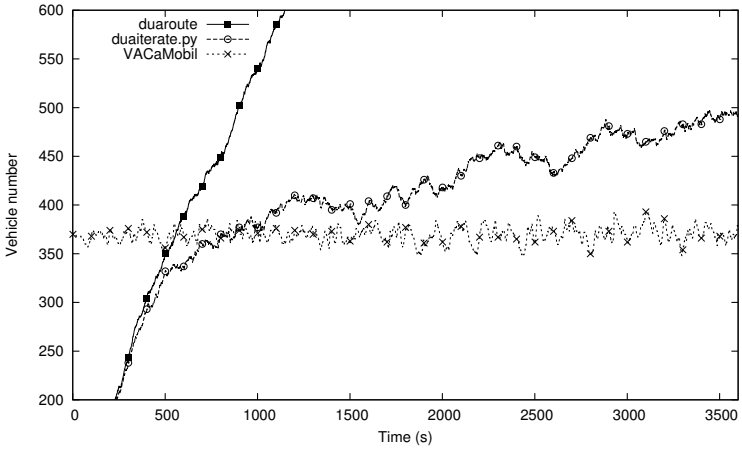


Figure 6.5: Vehicle number evolution for the urban scenario.

takes more time for vehicles to reach their destination and leave the network, which leads to a constantly increasing number of vehicles in the network when not using VACaMobil. Comparing the configuration in table 6.1 and the results in table 6.2, we can conclude that both the target number of vehicles and the standard deviation goal are clearly achieved with our VACaMobil approach.

6.5 Conclusions and future work

In this chapter we presented VACaMobil[1], a new mobility manager for the OMNeT++ simulator which promotes the full repeatability of VANET simulations. In particular, we added critical features to previously existing tools, such as ensuring a constant number of vehicles during the entire simulation period, disseminating vehicles throughout the whole route-map, and offering the possibility of defining different vehicle types with different probabilities.

Contrarily to other existing tools, which are not able to control the mean number of vehicles nor its standard deviation, VACaMobil is able to maintain the mean number of vehicles and the standard deviation value within user-defined bounds. To the best of our knowledge, this is currently the only tool that allows studying a vehicular network in a steady situation without losing the realistic vehicle behavior provided by SUMO.

As future work we plan to improve VACaMobil offering downtown definition and automatic placement of Road Side Units (RSU).

Acknowledgements

This work was partially supported by the *Ministerio de Economía y Competitividad*, Spain, under Grants TIN2011-27543-C03-01 and BES-2012-052673, and by the *Ministerio de Educación*, Spain, under the FPU program, AP2010-4397, AP2009-2415.

[1] VACaMobil is freely available at *www.grc.upv.es/software*.

Bibliography

[1] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. Sumo - simulation of urban mobility: An overview. In *SIMUL 2011, The Third International Conference on Advances in System Simulation*, pages 63–68, Barcelona, Spain, October 2011.

[2] Ling-Jyh Chen, Ying-Yu Chen, Kun chan Lan, and Chien-Ming Chou. Localized data dissemination in vehicular sensing networks. In *Vehicular Networking Conference (VNC), 2009 IEEE*, pages 1 –6, oct. 2009.

[3] M. Fogue, P. Garrido, F. J. Martinez, J.-C. Cano, C. T. Calafate, and P. Manzoni. An adaptive system based on roadmap profiling to enhance warning message dissemination in vanets. *Networking, IEEE/ACM Transactions on*, PP(99):1, 2012.

[4] Manuel Fogue, Piedad Garrido, Francisco J. Martinez, Juan-Carlos Cano, Carlos T. Calafate, and Pietro Manzoni. Using roadmap profiling to enhance the warning message dissemination in vehicular environments. In *36th IEEE Conference on Local Computer Networks (LCN 2011)*, Bonn, Germany, October 2011.

[5] M. Gramaglia, M. Calderon, and C.J. Bernardos. Trebol: Tree-based routing and address autoconfiguration for vehicle-to-internet communications. In *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, pages 1 –5, may 2011.

[6] Stefan Joerer, Christoph Sommer, and Falko Dressler. Toward Reproducibility and Comparability of IVC Simulation Studies: A Literature Survey. *IEEE Communications Magazine*, 50(10):82–88, October 2012.

[7] Xu Li, Wei Shu, Minglu Li, Hongyu Huang, and Min-You Wu. DTN Routing in Vehicular Sensor Networks. In *Global Telecommunications*

Conference, 2008. IEEE GLOBECOM 2008. IEEE, pages 1–5. IEEE, November 2008.

[8] Chun-Chih Lo, Jeng-Wei Lee, Che-Hung Lin, Mong-Fong Horng, and Yau-Hwang Kuo. A cooperative destination discovery scheme to support adaptive routing in vanets. In *Vehicular Networking Conference (VNC), 2010 IEEE*, pages 202 –208, dec. 2010.

[9] F.J. Martinez, J.-C. Cano, C.T. Calafate, and P. Manzoni. Citymob: A mobility model pattern generator for vanets. In *Communications Workshops, 2008. ICC Workshops '08. IEEE International Conference on*, pages 370 –374, may 2008.

[10] C. Sommer and F. Dressler. Progressing toward realistic mobility models in vanet simulations. *Communications Magazine, IEEE*, 46(11):132 –137, november 2008.

[11] C. Sommer, R. German, and F. Dressler. Bidirectionally coupled network and road traffic simulation for improved ivc analysis. *Mobile Computing, IEEE Transactions on*, 10(1):3 –15, jan. 2011.

[12] C. Sommer, O.K. Tonguz, and F. Dressler. Adaptive beaconing for delay-sensitive and congestion-aware traffic information systems. In *Vehicular Networking Conference (VNC), 2010 IEEE*, pages 1 –8, dec. 2010.

[13] C. Sommer, O.K. Tonguz, and F. Dressler. Traffic information systems: efficient message dissemination via adaptive beaconing. *Communications Magazine, IEEE*, 49(5):173 –179, may 2011.

[14] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, WDTN '05, pages 252–259, New York, NY, USA, 2005. ACM.

[15] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 1312 – 1321 vol.2, march-3 april 2003.

Chapter 7

An Adaptive Anycasting Solution for Crowd Sensing in Vehicular Environments

Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. *An adaptive anycasting solution for crowd sensing in vehicular environments.* IEEE Trans. Industrial Electronics , 62(12):7911-7919, 2015.

Abstract

Vehicular networks can be seen as the new key enablers of the future networked society. Vehicles traveling can act as mobile sensors and collect a variety of information that can be used to enable various new services such as environment monitoring, traffic management, urban surveillance, and so on. In this chapter, we present "adaptive Anycasting solution for Vehicular Environments" (AVE), which is a message delivery protocol that combines geographical and topological information to dynamically adapt its behavior to network conditions. We focus on vehicle-to-infrastructure connectivity for cloud services, where the vehicles send the sensed information as individual and independent messages to a cloud service in the Internet. This scenario requires access to any available close-by roadside unit, thus making anycasting the ideal delivery mechanism. Simulations results show that the hybrid and adaptive approach of AVE is able to improve network performance. For example, regarding delivery

ratio, AVE outperforms DYMO by 10% in sparse scenarios and outperforms delay-tolerant networking techniques by 10% in dense scenarios.

7.1 Introduction

Vehicular networks (e.g., [12, 19]) can be seen as the new key enablers of the future networked society. Vehicles traveling can act as mobile sensors and collect a variety of information. The vast information collected by vehicles can be used to enable various new services like environment monitoring, pollution measurements, safety, smart navigation, traffic management, and urban surveillance. The acquisition, sharing, processing, and transmission of data from vehicles foresee a new way to manage the communications. Vehicular networks create very varying topologies along time, often including isolated nodes, variable quality links, and variable nodes densities.

Such highly mutable and heterogeneous context represents a problem for network protocols, which are typically designed for more homogeneous scenarios: state-of-the-art routing protocols designed for one environment either work poorly or are unsuitable in others; limitations have been demonstrated (see, for example, [26]). Most proactive as well as on-demand mobile ad hoc network (MANET) protocols (e.g., [22, 27]) also assume the availability of a contemporaneous end-to-end path. Geographical routing (e.g., [25]), i.e., taking advantage of information about the location, emerged as a more efficient solution. However, geographical algorithms cannot achieve the best performance in every situation, the worst of which is related with the handling of "local minimum". Likewise, delay-tolerant networking (DTN) protocols commonly use packet replication to reduce delays, but packet replication performs poorly in predictable, dense vehicular networks ([17]). For a complete review of the current status of the art in this area please refer to [8].

Focussing on vehicular sensing, anycasting is the ideal mechanism since it provides the transport of information towards intended receivers, i.e., those specifying interest in the information. We therefore present "adaptive Anycasting solution for Vehicular Environments" (AVE), an adaptive message delivery protocol which combines geographical and topological information to dynamically adapt its behaviour to network conditions. The AVE approach has been evaluated in simulated urban scenarios under realistic settings. Realistic propagation models have been used, and urban layouts from the OpenStreetMap [11] database have been included in order to achieve a realistic road layout combined with an accurate building distribution. Simulations results show that the hybrid and adaptive approach of AVE is able to improve net-

work performance. For example, regarding delivery rate, AVE outperforms DYMO [22] by 10% in sparse scenarios and outperforms DTN [17] by 10% in dense scenarios.

The rest of this chapter is organized as follows: Section 7.2 gives an overview of the current status of research in the area and Section 7.3 presents the overall protocol behaviour and its implementation. Section 7.4 evaluates the protocol and discusses the results obtained. Finally, Section 7.5 concludes the chapter.

7.2 Related works

In the context of vehicular sensing, anycasting is the ideal mechanism since it provides information delivery towards the intended receivers while meeting certain design objectives; in anycasting, the planned receivers are those specifying interest in the information. Users may also define arbitrary interests: "parking spots availability in the center of the town", "traffic status close to the stadium", etc.

Robust message dissemination approaches would involve low delay, high reliability, low memory occupancy, and low message passing overhead. For example, in [23] the autor addresses the problem of stability of anypath communications in VANET networks in the presence of inter-vehicle link failures associated to vehicles mobility. In [5] the authors present a functionally complete realization of Try-Once-Discard (TOD) in wireless multihop networks. This mechanism allows network resources to be dynamically scheduled based on actual needs.

In [20], Niu *et. al.* present R3E, which is designed to enhance existing reactive routing protocols to provide reliable and energy-efficient packet delivery against unreliable wireless links by taking advantage of local path diversity. At higher layers [9] presents a mobile information system named *vehicle-to-anything application* (V2Anything App) aimed at giving relevant information to full electric vehicle (FEV) drivers by supporting the integration of several sources of data in a mobile application, thus contributing to the deployment of the electric mobility process. A more complete review of the current status of the art in this area can be found in [21]. The novelty in our proposal revolves around the adaptivity concept for a message dissemination proposal where a single solution by itself is not enough, no matter how flexible it is. The crucial point in providing adaptivity is the decision criteria to be adopted by the forwarding approach. AVE wants to be a step ahead in this sense.

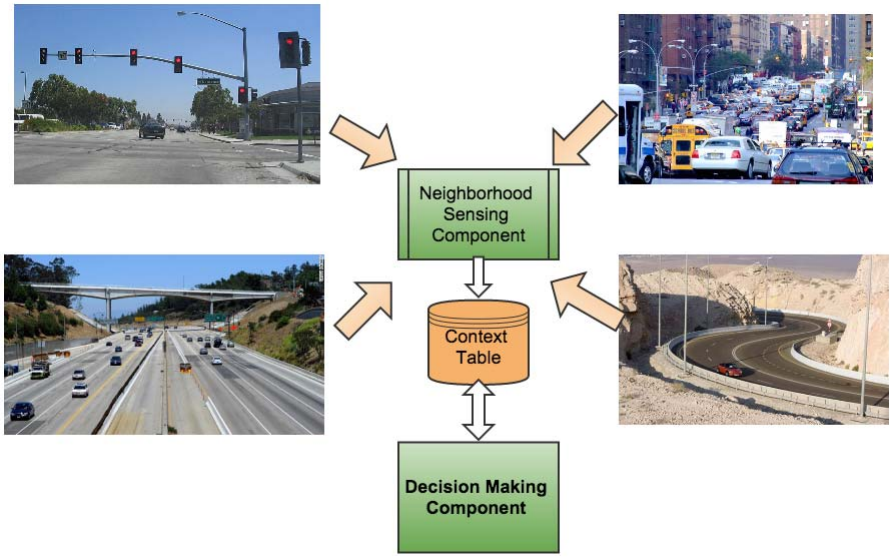


Figure 7.1: Protocol architecture.

7.3 Adaptive Anycasting Solution for Vehicular Environments

In anycasting the planned receivers are those specifying interest in the information. In this sense, our "adaptive Anycasting solution for Vehicular Environments" (AVE) focuses on a scenario where a mobile node wants to send the sensed information to a cloud service in the Internet, thus requiring to deliver individual and independent messages (bundles) to any available close-by Road Side Unit (RSU).

AVE aims to be robust to diverse connectivity possibilities. It autonomously adapts to the current status and context of the network, and reacts by choosing the best forwarding strategy. Figure 7.1 shows its architecture within four possible contexts: sparse urban, dense urban, sparse highway, and sparse rural.

AVE basically integrates three main elements: 1) the neighborhood sensing component, 2) the context table, and 3) the decision making component. The *neighborhood sensing component* provides the required input either to update the context table and to make forwarding decisions. In the *context table*, AVE stores and classifies the neighbors detected, maintains updated location data from one-hop neighbors, as well as information about the stability of the

different links, too. The *decision making component* is in charge of deciding which strategy is the most appropriate to forward the packet according to the network state.

The AVE neighborhood sensing component makes use of periodic beaconing to gain awareness of nearby nodes. It is based on the NeighborHood Discovery Protocol (NHDP) [6] and uses the standard packet format definition described in [7]. The beacon interval can be set arbitrarily, although recommended values are discussed in Section 7.3.3.

Beacons include data about the neighbor topological and geographical state; Figure 7.2 shows the proposed packet format. It includes relevant information such as, regarding topology: the IP address, the 1-hop neighbors IP addresses, and the link state; regarding location: the current location (coordinates), the current speed/direction, and the final destination (coordinates). Since beacons follow the NHDP packet format definition, their structure could be easily extended according to the rules in [7] to accommodate relevant information.

7.3.1 AVE overall operations

The decision-making component executes an algorithm that selects, among the available routing strategies, the most adequate one by taking into account the information retrieved about the observed network status maintained in the context table. The decision algorithm is shown in Figure 7.3.

AVE basically relaxes the delivery delay requirements in a progressive manner to increase delivery reliability. At the same time, it slowly increases the involved memory occupancy in nodes and the signaling overhead. The four strategies used are: local delivery, reactive routing, geographical routing and store-carry-forward delivery.

The local delivery strategy is the fastest and less resource consuming message delivery approach; it is used when the target RSU is only one or two hops away. In case a target RSU is further away, i.e., is not present in the context table, AVE starts a route discovery process. This action takes place only if a beacon from an RSU was received during the last T_{MAX} seconds. The T_{MAX} parameter prevents the route discovery process from starting. Its value was set taking into account the vehicle speed, the transmission range, and the mean inter-contact rate in a typical city. Studies on the traffic patterns and the contact rates between nodes in the city of Shangai [29] revealed that the inter-contact time of a vehicle with any other vehicle is in the order of several minutes. Nevertheless, a vehicle can still reach an RSU in 5 hops with a delay of about 3 minutes when setting the transmission range to 500m,

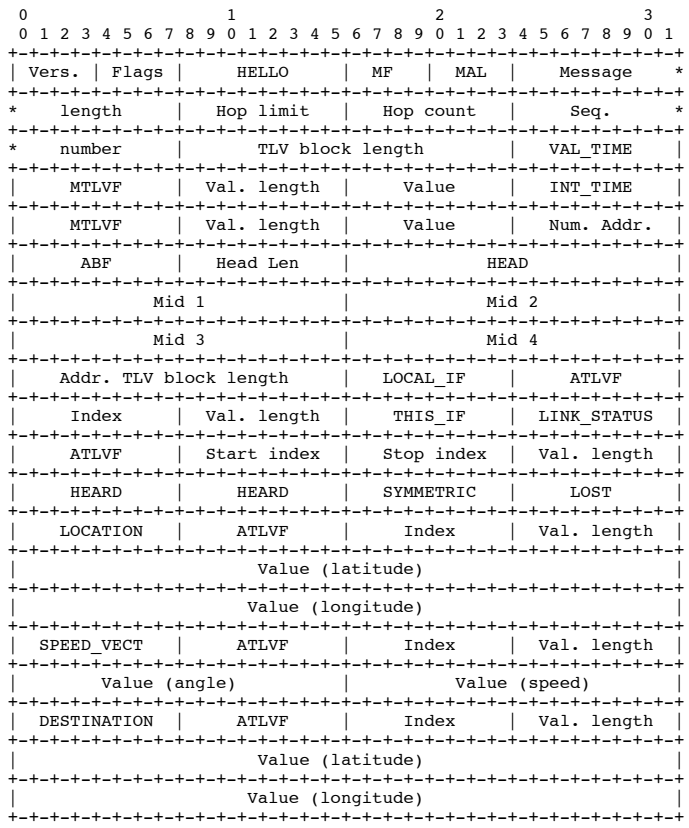


Figure 7.2: AVE beacons packet format.

considering the case where the route exists, and supposing it travels at the typical maximum city speed of 50 km/h.

In our tests we set T_{MAX} to 6 minutes to increase the chances of having a successful route discovery process. The basic reason behind this choice is twofold: (1) we consider that assuming that all vehicles move throughout a city at maximum speed is unrealistic, and (2) propagation in urban scenarios has a vast set of interference sources thus making it even more unreliable. If the route discovery process is unable to find a route within a specified time-out time (T_{END} = 10s), then the decision making component switches to geographical routing, thus relying on the position information of the closest RSU. Eventually, if no neighbors are known, i.e., the node is temporary isolated, the message is stored and carried until another connectivity possibility is available.

7.3.2 AVE algorithm details

The four strategies used, namely: local delivery, reactive routing, geographical routing and store-carry-forward delivery, were selected since they complement each other in a wide range of vehicular scenarios.

The first one, local delivery, is a fast approach that performs well in scenarios with a high RSU density, and where long paths are not expectable. In environments characterized by a low RSU density, the second strategy, based on a reactive protocol takes over to find an optimal path for the message to be delivered, regardless of the roads structure. The third strategy, i.e., the geographical approach, complements the reactive protocol by using a low delay greedy forwarding scheme, although it cannot deal with all road layouts, thereby presenting delays and inaccuracies related to its location service. Finally, the last strategy, the store-carry-and-forward approach, can deal with disconnected networks, covering the remaining network states; it makes predictions about the neighbors location at a future time, using a movement estimation that takes into account their actual position, their speed, and their direction. According to this prediction, we select the best one, that is the node that is supposed to get closer to an RSU sooner, as the message "custodian". This process is repeated whenever a contact between two mobile nodes occurs. Specific issues had anyway to be solved for each of the selected routing strategies and so, as described hereafter, a specific solution had to be implemented for each adopted strategy.

Purely MANET-like protocols generally tend to consume most available bandwidth just to get an updated set of routes to destination, or they are too slow and routes become rapidly unavailable. Proactive routing protocols,

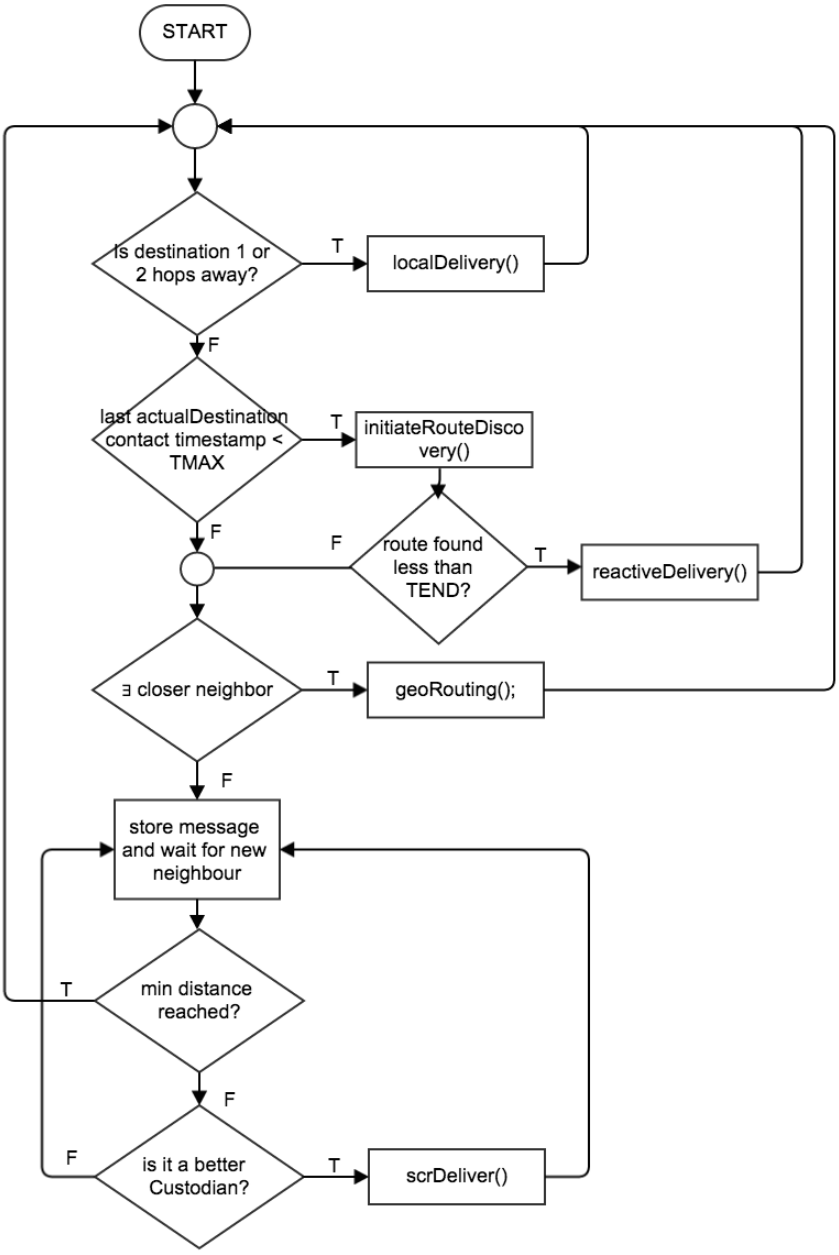


Figure 7.3: Decision algorithm.

for example, need a high beacon frequency to keep route information fresh, thus consuming a lot of bandwidth. Additionally, reactive routing protocols need too much time and too much bandwidth to discover long routes. This is why AVE first attempts to make a local delivery, and only if this is not successful it does activate path search. Moreover, we use a bounded reactive approach by reducing the maximum scope of the route search. This allows us i) to avoid long routes that can be possibly broken while the sending process is being carried out, and ii) to avoid wasting network resources with frequent broadcasts storms.

Focussing on the geographic routing protocols, we observe that they are based on a simple idea: look for the closest neighbor, compute the euclidean distance, and forward the packet to it. However, this strategy has two basic drawbacks: i) a greedy strategy does not guarantee finding the right path to the destination, so a recovery protocol must be included, and ii) a location service is needed. As will be shown in Section 7.4, although georouting should be, in theory, equally effective in dense and sparse networks, the combination of these two factors make a reactive routing approach performs better than geographic approaches in terms of packet delivery in more than 10% of the cases.

In geographical greedy protocols, a recovery process is triggered to find a route to destination when local minimums exist. It is usually based on the right-hand method, a method which sends the packet along the border of the network to find a node closer to the destination than the one which started the process. In AVE the recovery procedure is taken care by the reactive protocol or by the store-carry-and-forward approach. When a node is a local minimum for georouting, a route discovery procedure can be started if the restrictions of the algorithms allows (see Section 7.3.3), and the actual route to the destination can be found. If this route cannot be found, the store-carry-forward delivery takes over.

7.3.3 Specific issues

Here, we highlight several common problems of communications protocols in vehicular networks, and we show how AVE deals with them. Most of these issues have been addressed by other authors, but they still remain open.

7.3.3.1 The location service

In geographical routing, a component able to translate IP addresses into geographical coordinates is required. Location services are used for this task, but

packet losses associated to the lack of precision from this service can degrade the overall performance.

AVE uses periodic beaconing as the main source of geographical information, along with the RSU location, which is *a priori* known and stored in the context table. We have selected beaconing as a location service because it is able to maintain basic location knowledge about the neighbors of the vehicle without requiring other communication technologies (e.g., 3G networks). In this sense, and according for example to [14, 10, 15], choosing an adequate beacon frequency is a critical issue. A high beacon frequency will saturate the network, while a low beacon frequency offers the protocol a poor perception about the network state, thus reducing performance. In AVE we used the recommended value described in the NHDP's RFC document, namely 2 seconds.

7.3.3.2 Neighbor detection

Since wireless networks are deployed over lossy channels, detecting when a node can be a viable neighbor is a difficult task. Wireless links suffer from many communication problems, such as signal fading, interferences or ray reflection. Therefore, nodes cannot be assumed to be stable neighbors immediately after they are detected. Hysteresis is often used to detect when a node is a real neighbor, but this is a slow technique. Thus, it can be deployed in static environments, but it is not able to detect neighbors fast enough in highly dynamic environments.

Instead of using hysteresis, we decided to filter the packet dissemination. Since every beacon contains the location data, we can use this information to exclude as a neighbor those vehicles which are farther away than the theoretical maximum distance that wireless networks can cover. This theoretical maximum distance can be estimated from the local node configuration using the Free Space or the Two-ray ground propagation equations.

7.3.3.3 Loop avoidance

Since AVE involves many independent nodes, network loops may occur unless they are adequately coordinated. The highest probability of loops may appear when the decision algorithm in one node switches to a different strategy from the one selected by, for example, the previous node. In order to avoid this kind of loops, we set a sequential check in the algorithm to control the process, so preventing the selection of the delivery strategy in such an order that could create a loop. Essentially, supposing the routing protocols are numbered se-

quentially (1 - local delivery, 2 - reactive routing, 3 - geographical routing, and 4 - store-carry-forward delivery), if node A selects the routing protocol m to forward the packet to node B, node B would only choose a protocol n in such a way that $n \leq m$. This behavior is achieved with local delivery, because if node A selects it, this implies that node B is a neighbor of the destination and will use local delivery as well. It is also the case with reactive routing, because discovering a route to destination through node A implies that node B knows the route and the packet can also be routed with reactive routing. However, in order to adapt the protocol from dense networks to sparse networks, a transition between geographical routing and the store-carry-and-forward discipline is allowed. To avoid loops in that case, a *distance threshold* (d_{th}) similar to the one implemented in [4] is used in our algorithm. This parameter has a twofold function in our protocol: it does not only avoids loops, but it also prevents congestion caused by performing too many route discovery flooding.

7.4 Performance evaluation

Here, we assess the effectiveness of AVE by comparing its performance against a representative protocol for each of the different strategies adopted by AVE, namely with DYMO [22], Greedy georouting [13], and MSDP [18] for DTN routing. We evaluate AVE through simulation using the OMNeT++ [28] simulator and the INETMANET package [1]. Simulated nodes are configured to communicate using 802.11p devices; we used different extensions for signal propagation modelling [16, 2] and urban mobility generation [24] to greatly improve its realism.

Vehicular mobility has been defined using VACaMobil [3], a vehicular mobility manager which uses SUMO to achieve a realistic vehicular simulation while maintaining a constant average number of vehicles throughout the whole simulation. We have simulated our network in a real urban scenario: a $12km^2$ area from the Muscovite suburbs (see Fig. 7.4). Vehicles send short bursts of packets to an RSU at random points of their routes. Using this traffic pattern we can test the protocol performance for traffic sources in different points of the network, checking where the protocols are able to properly deliver the packets.

The evaluation was made in terms of packet delivery ratio (i.e., the percentage of packets that are actually delivered to the destination), packet delay (i.e., the average delay that packets suffer during their delivery), mean packet hop number (i.e., the average number of intermediate nodes that a node has to pass through), and *strategy usage ratio*; this last metric is detailed in the

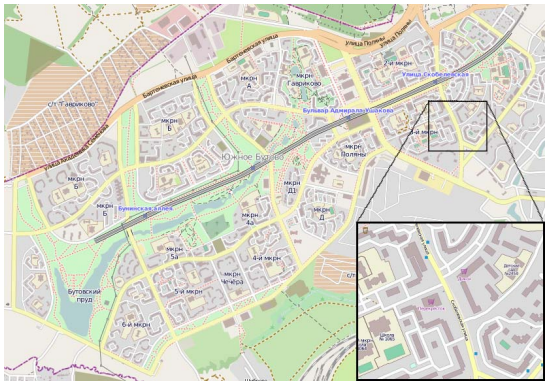


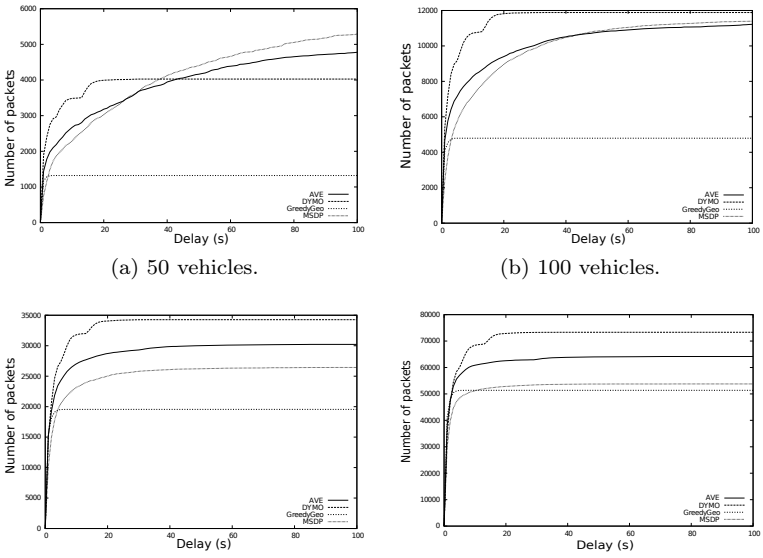
Figure 7.4: Simulated scenario: General view and detail.

corresponding subsection. All results represent an average over 25 executions with different random seeds, presenting all of them a degree of confidence of 90%.

7.4.1 Overall protocol performance

We first focus on AVE's message delay. Figure 7.5 compares the performance obtained for message delivery delay for AVE and the other three representative protocols. This scenario can reach two different states depending on the network density. In low densities, only store-carry-and-forward approaches are able to correctly deliver the packets due to network fragmentation. However, under high density networks, location service problems and channel issues makes topological approaches the best option. In the literature we can find some cases where low density (LD) is considered for values around $15 cars/km^2$ and high density (HD) for values of $75 cars/km^2$. In our case, we considered for LD a value of $4 cars/km^2$ to stress the case for disconnected scenarios, and as HD value we used $41 cars/km^2$, that represents scenarios with a sufficient concentration of cars as to frequently activate local delivery. Higher values for HD would have created situations where basically only local delivery would have been used.

AVE obtains an average behavior in the range between the two best options: DYMO and MSDP. The packet delay increments because AVE is trying to increment the message delivery ratio and to avoid network congestion. On the other hand, AVE outperforms MSDP delay since it can route some packets



(a) 50 vehicles. (b) 100 vehicles.

(c) 250 vehicles. (d) 500 vehicles.

Figure 7.5: Message delivery delay distribution.

more quickly using the topological alternatives.

Considering AVE's delivery ratio (see Figure 7.6), we see that it follows a similar trend. AVE obtains an average behavior in the range between the two best options. This makes AVE the most flexible protocol, achieving a good performance in all the available scenarios. In fact, AVE outperforms DYMO by 10% in low density scenarios. In dense networks, AVE also outperforms MSDP by 10% in dense scenarios at the expense of losing only 6% of data in sparse scenarios.

7.4.2 AVE detailed analysis

Here, we aim at analyzing AVE and exposing its behavior step by step. Due to its modular architecture, AVE can be easily examined and its performance in different network densities can be characterized.

To evaluate which protocols are chosen when the next hop is selected, we define the *Strategy Usage Ratio* (SUR) metric. This metric is defined as the total number of hops travelled by the message using a specific strategy over

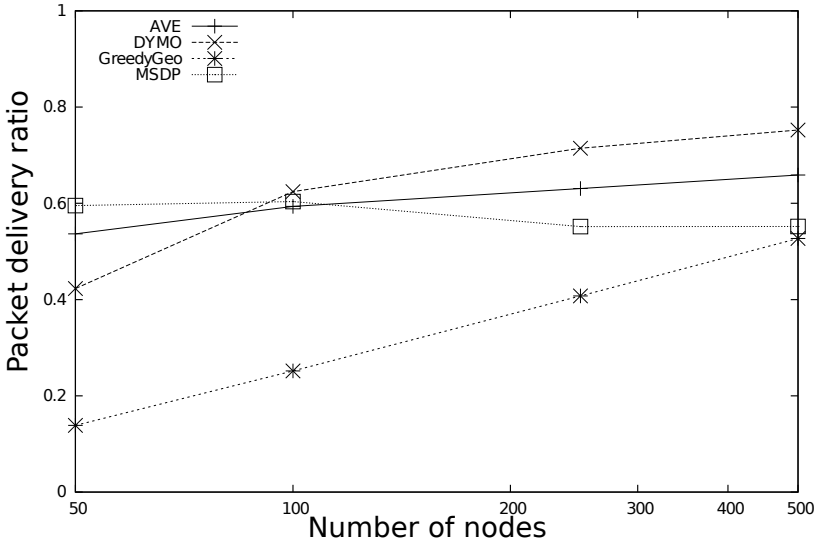


Figure 7.6: Packet delivery ratio.

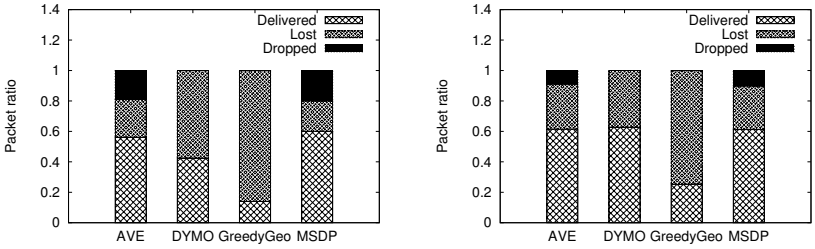
the total number of hops for the entire simulation. Its mathematical definition can be seen in Equation 7.1:

$$SUR(x) = R(x)/\sum_{i=0}^{n} R(i) \tag{7.1}$$

where $R(x)$ is the total number of hops travelled by the message using strategy x, and n is the total number of strategy items.

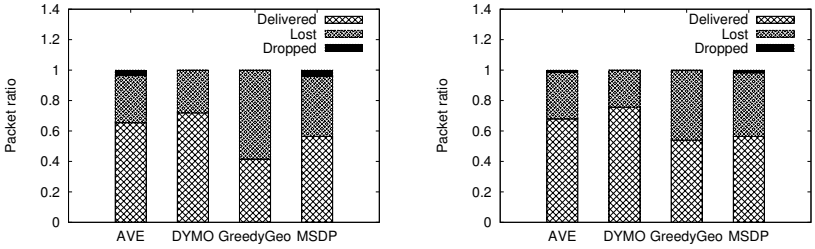
Figure 7.8 shows the SUR value for each strategy under different network densities. We can see how AVE cuts down the usage of store-carry-and-forward strategies by more that 20% when vehicle density increases. Therefore, the usage of the rest of approaches is incremented by about 10% of the geographical approaches, and by an additional 10% of the topological approaches. Moreover, this balance takes place automatically depending on the network disconnection states at different instants of time.

The packet loss causes are shown in Figure 7.7. There are two kinds of losses: (1) the node ends its trip but it was carrying a packet, and (2) the packet is transmitted to a new node, but it cannot reach its destination. The first group can only be reduced by improving the DTN algorithm. The second group includes both channel losses and location inaccuracy losses. Since the



(a) 50 vehicles.

(b) 100 vehicles.



(c) 250 vehicles.

(d) 500 vehicles.

Figure 7.7: Packet loss causes.

latter is the biggest cause of losses, research must be focused there to improve the final behavior.

This set of figures also shows us an interesting piece of information related to geographical routing. We can see that packet losses in high density networks are not related to network disconnection, as occurs for greedy georouting in sparse networks, nor to a bad custodian selection, because the packet drop rate is low. In this scenario, environmental losses, such as location system inaccuracies and channel fading effects, are more relevant.

7.4.3 Influence of the propagation model on performance

Finally, in this section we evaluate the performance impact of using a different propagation model. This new propagation model [2] accounts for small moving obstacles that can partly block the signal. Thus, it is a more realistic propagation model for vehicular networks because, in such environments, there are other vehicles, pedestrians, or even trees that can obstruct the signal. Experi-

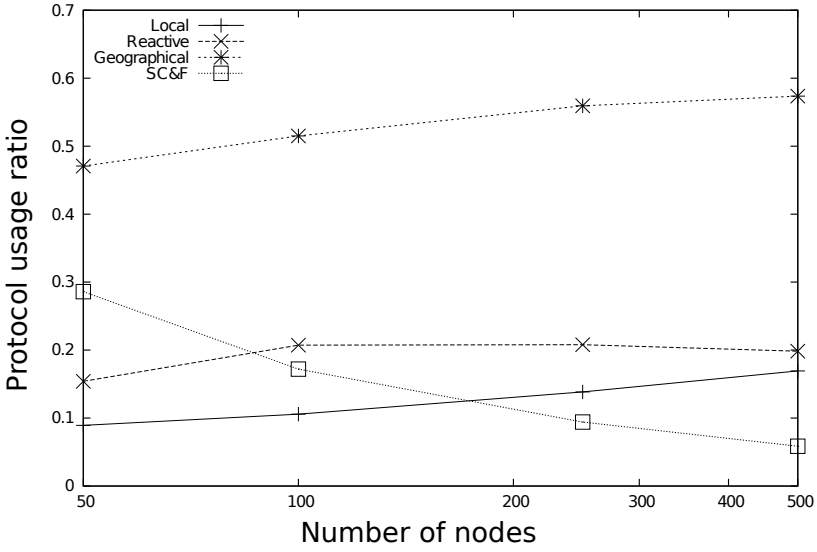


Figure 7.8: AVE SUR.

mentally, it has been measured that the mean propagation distance under this restriction is lower than the one achieved without taking these small obstacles into account.

Figure 7.10 shows the Strategy Usage Ratio when adopting this more realistic environment. In this case, the environment requires a higher flexibility in terms of protocol usage, and AVE doubles its usage of non-DTN approaches in order to improve its delay. In fact, DTN usage is reduced by 25% when comparing the densest against the sparsest scenario; we therefore reduce this result by a factor of two with respect to the case based on the previous propagation model.

Figure 7.9 shows the packet delivery ratio of AVE compared to the performance obtained by each of the representative protocol separately. This propagation model cuts down the overall performance of topological routing protocols, making geographical approaches the best choice for all network states. AVE achieves an average performance in sparse networks that is between two best options and, in addition it outperforms the best option in dense scenarios.

Figure 7.11 shows a similar trend to the one observed with the previous propagation model. However, in dense networks, we observe that AVE outperforms MSDP latency due to the use of topological approaches. Notice that topological knowledge is a more precise way to route a packet, since an actual

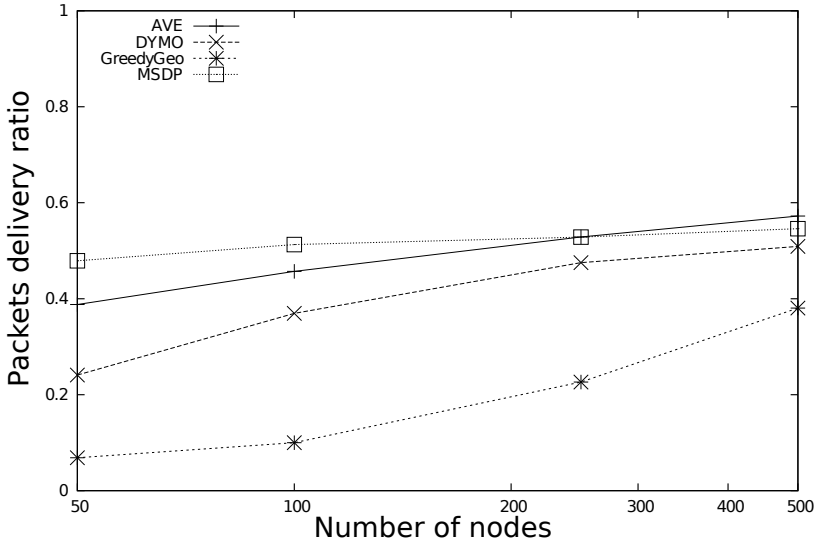


Figure 7.9: Packet delivery ratio.

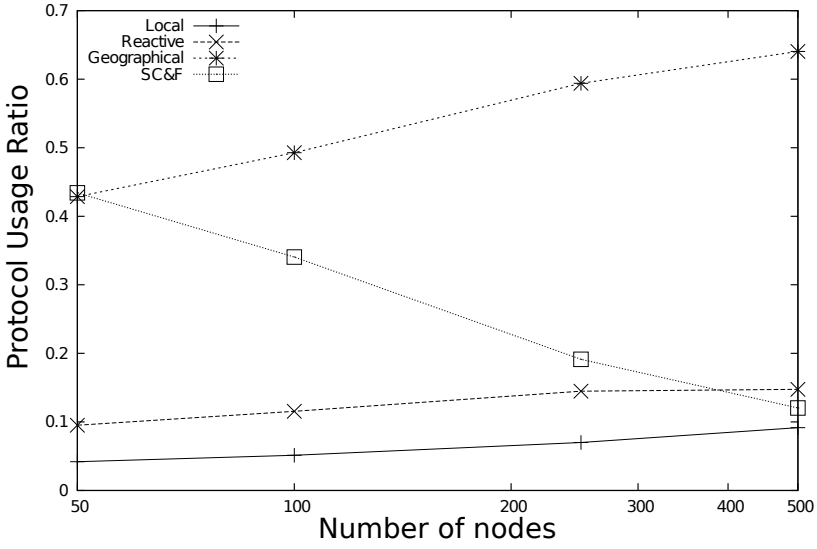
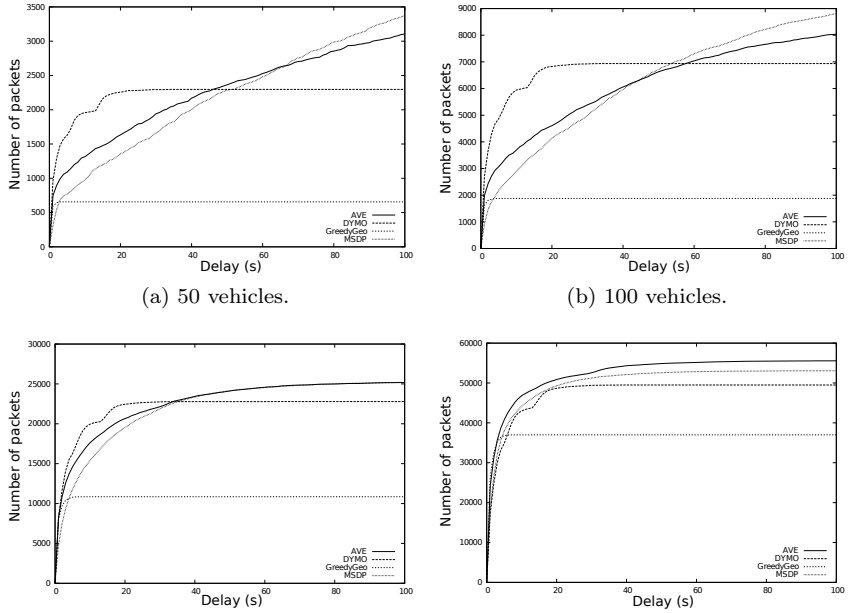


Figure 7.10: AVE SUR.

network topology awareness allows to discover the optimal path to route packets to their destination; with this strategy such information is gained through a flooding procedure.



(a) 50 vehicles.

(b) 100 vehicles.

(c) 250 vehicles.

(d) 500 vehicles.

Figure 7.11: Delay distribution.

7.5 Conclusions

Vehicles traveling can act as mobile sensors and collect a variety of information; but vehicular networks create very varying topologies along time, often including isolated nodes, variable quality links, and variable nodes densities. Such highly mutable and heterogeneous context represents a problem for network protocols, which are typically designed for more homogeneous scenarios.

In this chapter we presented AVE ("adaptive Anycasting solution for Vehicular Environments"), an adaptive message delivery protocol which combines geographical and topological information to dynamically adapt its behavior to network conditions. It autonomously selects the most appropriate approach to forward each message based on the input obtained from an adapted im-

plementation of the NeighborHood Discovery Protocol (NHDP). In AVE we focussed on a basic scenario: V2I connectivity for cloud services. In this scenario, the vehicles send the sensed information as individual and independent messages (bundles), to a cloud service in the Internet, thus requiring to deliver the messages to any available close-by Road Side Unit (RSU).

The AVE approach was evaluated in simulated urban scenarios under realistic settings. Realistic propagation models were used and urban layouts from the OpenStreetMap database were included in order to achieve a realistic road layout combined with an accurate building distribution. Results were analyzed and showed AVE's adaptation capabilities under different conditions, while also allowing to detect areas of possible improvement, highlighting the different trade-offs and how they are addressed by the protocol. Simulations results showed that the hybrid and adaptive approach of AVE was able to improve network performance. For example, regarding delivery rate, AVE outperformed DYMO by 10% under sparse scenarios, and outperformed MSDP by 10% under dense scenarios.

We consider that AVE could be successfully used in application where there is a need to disseminate specific data to a group of consumers whose membership could vary dynamically in time. The information we are referring to can be anything like: "parking spots availability in the center of the town", "traffic status close to the stadium", etc.

Bibliography

[1] A. Ariza-Quintana, E. Casilari, and A. Triviño Cabrera. Implementation of manet routing protocols on omnet++. In *Simutools '08*, Simutools '08, ICST, Brussels, Belgium, Belgium, 2008. ICST.

[2] M. Baguena, C.T. Calafate, J. Cano, and P. Manzoni. Towards realistic vehicular network simulation models. In *Wireless Days (WD), 2012 IFIP*, pages 1–3, 2012.

[3] M. Baguena, S.M. Tornell, A. Torres, C.T. Calafate, J. Cano, and P. Manzoni. VACaMobil: VANET Car Mobility Manager for OMNeT++. In *Workshop on Smart Communication Protocols and Algorithms (SCPA), 2013 IEEE*, pages 1077–1081, 2013.

[4] Pei-Chun Cheng, Kevin C. Lee, Mario Gerla, and Jérôme Härri. Geodtn+nav: Geographic dtn routing with navigator prediction for urban vehicular environments. *Mob. Netw. Appl.*, 15(1):61–82, February 2010.

[5] D. Christmann, R. Gotzhein, S. Siegmund, and F. Wirth. Realization of try-once-discard in wireless multihop networks. *Industrial Informatics, IEEE Transactions on*, 10(1):17–26, Feb 2014.

[6] T. Clausen, C. Dearlove, and J. Dean. Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP). *RFC 6130*, 2011.

[7] T. Clausen, C. Dearlove, J. Dean, and C. Adjih. Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format. *RFC 5444*, 2009.

[8] F. Dressler, H. Hartenstein, O. Altintas, and O. Tonguz. Inter-vehicle communication: Quo vadis. *Communications Magazine, IEEE*, 52(6):170–177, June 2014.

[9] J.C. Ferreira, V. Monteiro, and J.L. Afonso. Vehicle-to-anything application (v2anything app) for electric vehicles. *Industrial Informatics, IEEE Transactions on*, 10(3):1927–1937, Aug 2014.

[10] Jerome Haerri, Fethi Filali, and Christian Bonnet. Performance comparison of aodv and olsr in vanets urban environments under realistic mobility patterns. In *Proc. of 5th IFIP Mediterranean Ad-Hoc Networking Workshop (Med-Hoc-Net-2006), Lipari, Italy*, 2006.

[11] Mordechai Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE*, 7(4):12–18, 2008.

[12] Wu He, Gongjun Yan, and Li Da Xu. Developing vehicular data cloud services in the IoT environment. *Industrial Informatics, IEEE Transactions on*, 10(2):1587–1595, May 2014.

[13] Brad Karp and Hsiang-Tsung Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 243–254. ACM, 2000.

[14] I. Khan and A. Qayyum. Performance evaluation of aodv and olsr in highly fading vehicular ad hoc network environments. In *Multitopic Conference, 2009. INMIC 2009. IEEE 13th International*, pages 1–5, 2009.

[15] Dong-Won Kum, Jin-Su Park, You-Ze Cho, and Byoung-Yoon Cheon. Performance evaluation of aodv and dymo routing protocols in manet. In *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE*, pages 1–2, 2010.

[16] Andreas Kuntz, Felix Schmidt-Eisenlohr, Oliver Graute, Hannes Hartenstein, and Martina Zitterbart. Introducing probabilistic radio propagation models in omnet++ mobility framework and cross validation check with ns-2. In *Simutools'08*, page 72. ICST, 2008.

[17] Yong Li, Mengjiong Qian, Depeng Jin, Pan Hui, Zhaocheng Wang, and Sheng Chen. Multiple mobile data offloading through disruption tolerant networks. *Mobile Computing, IEEE Transactions on*, 13(7):1579–1596, July 2014.

[18] S. Martinez Tornell, C.T. Calafate, J.-C. Cano, and P. Manzoni. A map-based sensor data delivery protocol for vehicular networks. In *Ad Hoc Networking Workshop (Med-Hoc-Net), 2012 The 11th Annual Mediterranean*, pages 1–8, June 2012.

[19] Wei Ni, I.B. Collings, Ren Ping Liu, and Zhuo Chen. Relay-assisted wireless communication systems in mining vehicle safety applications. *Industrial Informatics, IEEE Transactions on*, 10(1):615–627, Feb 2014.

[20] Jianwei Niu, Long Cheng, Yu Gu, Lei Shu, and S.K. Das. R3E: Reliable reactive routing enhancement for wireless sensor networks. *Industrial Informatics, IEEE Transactions on*, 10(1):784–794, Feb 2014.

[21] S. Panichpapiboon and W. Pattara-Atikom. A review of information dissemination protocols for vehicular ad hoc networks. *Communications Surveys Tutorials, IEEE*, 14(3):784–798, Third 2012.

[22] C. Perkins, S. Ratliff, and J. Dowden. Dynamic MANET on-demand (DYMO) routing. *draft-ietf-manet-dymo-26 (work in progress)*, 2013.

[23] J. Rak. LLA: A new anypath routing scheme providing long path lifetime in VANETs. *Communications Letters, IEEE*, 18(2):281–284, February 2014.

[24] Christoph Sommer, Reinhard German, and Falko Dressler. Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. *IEEE Transactions on Mobile Computing*, 10(1):3–15, January 2011.

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

[26] Xiaozheng Tie, Arun Venkataramani, and Aruna Balasubramanian. R3: Robust replication routing in wireless networks with diverse connectivity characteristics. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, MobiCom '11, pages 181–192, New York, NY, USA, 2011. ACM.

[27] J. Toutouh, J. Garcia-Nieto, and E. Alba. Intelligent OLSR routing protocol optimization for VANETs. *Vehicular Technology, IEEE Transactions on*, 61(4):1884–1894, May 2012.

[28] András Varga and Rudolf Hornig. An overview of the omnet++ simulation environment. In *Simutools '08*, pages 1–10, ICST, Brussels, Belgium, Belgium, 2008. ICST.

[29] Hongzi Zhu, Luoyi Fu, Guangtao Xue, Yanmin Zhu, Minglu Li, and L.M. Ni. Recognizing exponential inter-contact time in vanets. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–5, 2010.

Chapter 8

Towards Enabling Hyper-Responsive Mobile Apps at Scale Using Edge Assistance

Miguel Baguena, George Samaras, Andreas Pamboris, Mihail L. Sichitiu, Peter R. Pietzuch, and Pietro Manzoni. *Towards enabling hyper-responsive mobile apps through network edge assistance.* In 13th IEEE Annual Consumer Communications & Networking Conference, CCNC 2016, Las Vegas, NV, USA, January 9-12, 2016, pages 399-404. IEEE, 2016.

Abstract

Poor Internet performance currently undermines the efficiency of hyper-responsive mobile apps such as augmented reality clients and online games, which require low-latency access to real-time backend services. While *edge-assisted execution*, i.e. moving entire services to the edge of an access network, helps eliminate part of the communication overhead involved, this does not scale to the number of users that share an edge infrastructure. This is due to a mismatch between the scarce availability of resources in access networks and the aggregate demand for computational power from client applications.

Instead, this chapter proposes a *hybrid edge-assisted deployment model* in

which only part of a service executes on LTE edge servers. We provide insights about the conditions that must hold for such a model to be effective by investigating in simulation different deployment and application scenarios. In particular, we show that using LTE edge servers with modest capabilities, performance can improve significantly as long as at most 50% of client requests are processed at the edge. Moreover, we argue that edge servers should be installed at the core of a mobile network, rather than the mobile base station: the difference in performance is negligible, whereas the latter choice entails high deployment costs. Finally, we verify that, for the proposed model, the impact of user mobility on TCP performance is low.

8.1 Introduction

Recent technological trends such as the emergence of wearable devices (e.g. Google Glass), the availability of ubiquitous wireless networks (e.g. 3G, 4G/LTE and WiFi) and new cognitive algorithms that surpass human abilities in tasks such as computer vision, speech recognition and natural language translation, have brought a new class of hyper-responsive mobile apps to our doorstep. For such applications to work as intended, clients require *ultra low latency access* to *real-time backend services*, ideally with at most tens of milliseconds of delay [12] in order to attain the speed of human perceptual and cognitive processing.

The underlying network that interconnects clients and backend services is often a major hindrance to achieving the seamless responsiveness users expect from smartphones and wearable devices. As it stands, backend services are typically hosted at distant cloud-based machines, which in many cases translates to hundreds of milliseconds of round trip time (RTT) delay between clients and services. An alternative is to re-locate entire services to the edge of the access network, so as to avoid high Internet delays. Nevertheless, in light of the increasing number of mobile clients, it remains unclear to what extent this infrastructure can replace cloud-scale data centers due to the comparatively limited availability of resources at the network edge [15, 11].

We envision a *hybrid edge-assisted service deployment model* that allows for moving only part of the backend logic into the access network as the predominant model for supporting real-time services. The advantage of this model is that it can reduce network transmission delays for multiple mobile clients, while respecting edge resource limitations. This chapter explores the conditions that must hold in order to support hyper-responsive mobile apps at large scale using edge assistance. We first quantify the impact of 4G/LTE net-

works on application performance in terms of end-to-end latency and jitter, considering different types of networked application workloads. We then evaluate through simulation how performance can be improved using our proposed model, taking into account: (a) different placements of edge servers within the LTE infrastructure; (b) the number of clients sharing edge resources; (c) the ratio of user requests served by edge servers over the total requests made by a particular mobile client; (d) different processing capabilities of edge servers; and (e) user roaming.

Our key insights from this work are summarised below:
(1) A hybrid edge-assisted deployment model can improve collectively the responsiveness of mobile clients over LTE networks: it avoids part of the communication over slow Internet network links and, at the same time, it relieves edge servers from increased computational load.
(2) The difference in application responsiveness between the two alternative locations for installing edge servers, i.e. at the eNodeB or the Packet Data Network Gateway (PGW), is negligible. Therefore, in light of the deployment costs involved with the former choice, the PGW gateway should be preferred.
(3) Due to the limited processing capabilities of edge servers and the increasing number of clients sharing edge infrastructure, performance gains are only achieved when up to 50% of user requests are processed at the network edge.
(4) TCP performance is not affected by user roaming, therefore, no adaptation to the transport protocol is required.

The remainder of this chapter is organised as follows: §8.2 describes the architecture of 4G/LTE networks and discusses previous work on cloud-assisted application execution as well as the current support for deploying services at the network edge; §8.3 introduces the idea of a hybrid edge-assisted service deployment model; §8.4 evaluates the proposed model in simulation; and §8.5 concludes the chapter.

8.2 Background

LTE networks. 4G/LTE networks typically consist of three main components: *User Entities (UEs)*, which are commonly referred to as mobile devices; the *eNodeBs*, i.e. base stations communicating directly with UEs; and the *Evolved Packet Core (EPC)*, which comprises two main components for data transmission: the Serving Gateway (SGW), which connects eNodeBs to the EPC; and the Packet Data Network Gateway (PGW), which provides access to external IP networks. The primary communication protocol used is the GPRS Tunnelling Protocol (GTP). To handle user mobility, the two main

types of handover events supported are the X2- and S1-based handovers. This work considers the latter, which is the most time-consuming of the two and therefore more likely to degrade TCP performance.

Cloud-assisted execution. Systems that offload CPU-intensive computation from mobile devices to the cloud are the predominant remedy to work around resource and power limitations of today's mobile devices. Existing approaches either treat mobile devices as thin clients by offloading all of an application's logic to the cloud [1], or employ a more fine-grained application partitioning for increased performance gains [3, 7, 4, 2].

Despite the increasing popularity of code-offloading approaches, their efficiency remains restricted by network performance. Therefore, it has been proposed to leverage edge infrastructure, referred to as *cloudlets*, to replace distant cloud-based nodes when possible [14]. Besides improving application responsiveness, the ability to host applications at the network edge has been exploited in different ways, e.g. to relieve mobile networks from unused traffic using edge proxies that filter the data sent to mobile devices [13].

Cloudlet-based approaches presupposes the availability of enough resources at the edge of the network in order to accommodate mobile users that share the edge infrastructure. Especially in densely populated urban environments, moving backend services in their entirety to the edge of a mobile network requires edge resources that are comparable to what data centres offer today. However, this assumption in impractical as it requires significant changes to current mobile network infrastructure and therefore entails high operational costs.

Support for edge-assisted execution. The need for fast access to real-time backend services has led to re-visiting the concept of edge computing. This is evidenced by the *Mobile Edge Computing (MEC)* initiative [10], a new Industry Specification Group within the European Telecommunications Standards Institute (ETSI) that aims at providing application developers and content providers with cloud-computing capabilities at the edge of the mobile network.

The release of new edge computing platforms such as *IBM ASPN* [5] is a step towards this direction, which allows for running and managing applications directly within mobile networks, provided that the required edge infrastructure is in place. Nevertheless, the capabilities of *edge servers* installed in mobile networks are still limited by form-factor and placement constraints [6]. An example is the new *Radio Applications Cloud Server (RACS)* by Nokia Networks [11], designed explicitly to fit in base stations, albeit with limited processing and storage capabilities. In this chapter, we investigate how we can work around such limitations to take advantage of the available edge infras-

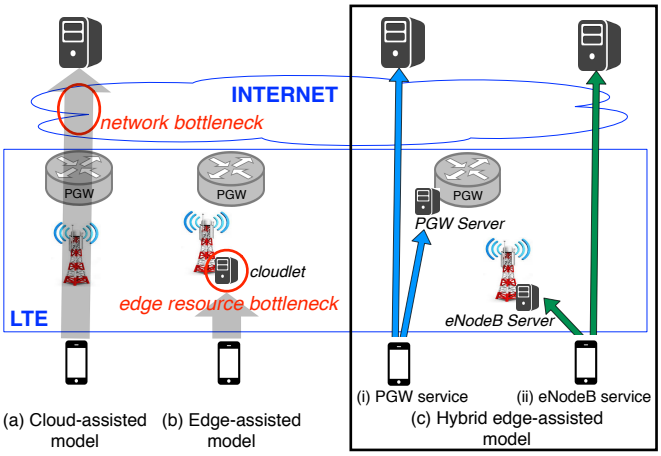


Figure 8.1: Hybrid edge-assisted deployment model

tructure to benefit many users collectively.

8.3 Proposed Hybrid Deployment Model

This section positions the proposed *hybrid edge-assisted service deployment model* in relation to extreme approaches that either deploy services in their entirety at cloud-scale data centres or at the edge of an access network. A comparison of all three deployment models is shown in Figure 8.1.

Currently the prevalent approach for deploying real-time backend services is the *cloud-assisted model*, which is illustrated in Figure 8.1(a). In this model, services execute at distant cloud-based servers with virtually unlimited resources. Although in terms of computational resources, this model scales well to an increasing number of mobile clients, it suffers from high Internet delays during the communication between mobile clients and services.

At the other end of the spectrum, a *pure edge-assisted model*, shown in Figure 8.1(b), moves entire services closer to end users in order to eliminate part of the communication overhead over the Internet. While this model requires no changes to existing services, it is constrained by the fact that resources at the edge are significantly less plentiful than what is available in large data centres. Therefore, this approach cannot scale to the demand of an increasing number of mobile clients sharing the edge infrastructure.

The *hybrid edge-assisted deployment model*, which is shown in Figure 8.1(c), sits between the aforementioned approaches. The intuition behind this model is that it prioritises client requests based on latency requirements and splits services accordingly to execute across a cloud and an edge environment. This allows for improving app responsiveness for many clients while reducing the resource burden on edge servers.

We investigate the effectiveness of this hybrid model in the remainder of this chapter. Note that how to realise the actual split of services into remote and edge components is specific to each service and thus not the focus of our work. Service providers, however, can base such decisions on the insights drawn from this study.

8.4 Experimental Study

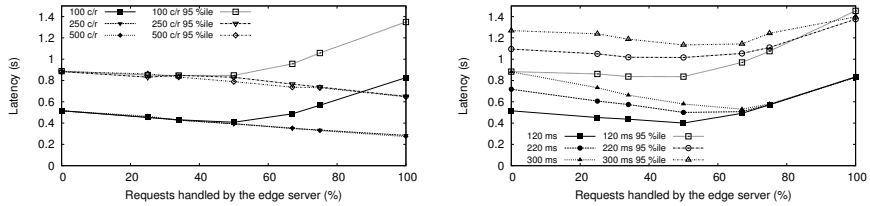
Through experiments in simulation, we evaluate the performance of edge-assisted applications in 4G/LTE networks. We first describe the simulation set-up and our experimental methodology. We then analyse the results, highlighting the benefits and drawbacks of different configurations for the proposed hybrid deployment model.

8.4.1 Simulation setup and methodology

We perform experiments in a simulated 4G/LTE network using the *OMNeT++* framework. We extended three publicly available models to simulate a complete 4G access network: INET[1] provides the tools for simulating a wired network and related protocols; SimuLTE[2] models the physical wireless link in a 4G network, as well as the link-level protocols between mobile devices and eNodeBs; and 4GSim[3] implements the network gateways and the control protocols that allow mobile devices to interact with control devices.

Simulation scenario. We focus on a 4G/LTE access network that contains two eNodeBs with several mobile devices attached to them. The eNodeBs are connected with the Evolved Packet Core (EPC) through a node that allows us to inject arbitrary delays in the access network, thus modelling the heterogeneity in current access networks. *Edge servers* can be placed at the eNodeB or the EPC, i.e. next to the PGW.

[1] http://inet.omnetpp.org
[2] http://simulte.com/
[3] https://github.com/4gsim/4Gsim



(a) Different number of concurrent requests (c/r) that can be processed simultaneously

(b) Different Internet latencies

Figure 8.2: Average latency and 95th percentile for different percentages of requests processed by the PGW edge server

Edge servers have limited processing, disk and memory resources due to their form-factor limitations compared to servers located outside of the access network. We model this by enforcing a maximum number of requests that an edge server can handle concurrently; additional requests are queued until a new processing slot becomes available. We model the probability that a request is served by the edge server (and not a cloud-based server) according to a variable service rate.

Application network traces. We use network traces to evaluate the performance of the proposed hybrid model that are based on three distinct app traffic patterns with different characteristics: (1) the first pattern is characterised by large response packet sizes and low processing delays. This is common for apps such as augmented reality clients that perform image recognition on the mobile device and retrieve increasing amounts of data from a remote server to be superimposed on the device's camera output; (2) the second pattern exhibits the opposite behaviour: apps receive smaller packets but the backend logic is more CPU-intensive, thus leading to high processing delays. A representative example is a role-playing online game [16] in which users exchange information such as their location within a shared virtual world and various combat actions, which need to be processed remotely to determine how these affect other clients; and (3) the third pattern presents a balance between the two previous traces, and thus represents any other client apps with intermediate delays and packet sizes.

For all traces, we assume non-concurrent request/response interactions between individual clients and services—only one request per client is pending a response at any point in time. For mobile client apps, typically the size of a

Table 8.1: Characterisation of workload traces

Representative application	Inter-request delay	Request size	Processing delay	Response size
Augmented reality	50-100 ms	20-30 B	50-100 ms	700-900 B
Online games	200-250 ms	20-30 B	200-250 ms	100-200 B
Other	100-150 ms	20-30 B	100-150 ms	400-600 B

single request is small (tens of bytes) compared to the size of responses, which ranges from hundreds to even thousands of bytes. A detailed analysis of these traces is given in Table 8.1. It reports packet sizes as well as both processing and inter-request delays (due to application-level delays or user idle states). The LTE latency as well as delays due to hardware constraints are added by our simulator.

Due to space constraints, and given that the evaluation results obtained for all three traffic patterns exhibit similar trends, we only present the results that correspond to the online gaming use case in detail, which is the most compute-intensive of them all. However, we highlight any significant differences observed for the other two patterns when applicable.

8.4.2 Results

Next, we evaluate the performance of apps with edge assistance for different configurations. We consider the average request/response latency and its 95th percentile as the primary metrics for performance. This latency includes the transmission delays for both requests and corresponding responses and the server-side processing delay.

8.4.2.1 Edge server collocated with PGW

First, we assume that edge servers are attached to the PGW gateway. To identify the conditions that must hold for this configuration to achieve better performance, we vary the ratio of requests that are served entirely at the network edge and either (i) the number of concurrent requests that can be processed simultaneously by the edge server; or (ii) the Internet latency.

Variable edge server capability. Figure 8.2(a) plots the average and the 95th percentile of the latency per request for different fractions of requests processed at the network edge and different capabilities of the PGW edge server. We assume 500 cooperating mobile clients. The access network and Internet latencies are set to 25 ms [8] and 120 ms [9], respectively.

When most of the requests are directed to the backend server, the average application latency is above 500 ms and dominated by the request/response transmission delays. *By allowing more requests to be served by the edge server, high Internet delays are avoided and latency gradually decreases. However, when the processing capacity of the edge server saturates (at around 50% of requests), latency increases due to server queuing to a point at which edge-assisted execution is outperformed by the original configuration.* The 95th percentile curves show that when increasing the number of requests processed at the edge server, due to CPU contention, jitter (inferred by the difference between the average and 95th percentile values) also increases.

Similar trends are observed for the other two types of application traces. However, the time to process a request is significantly less. As a result, queues are less congested, thus reducing the queuing delay.

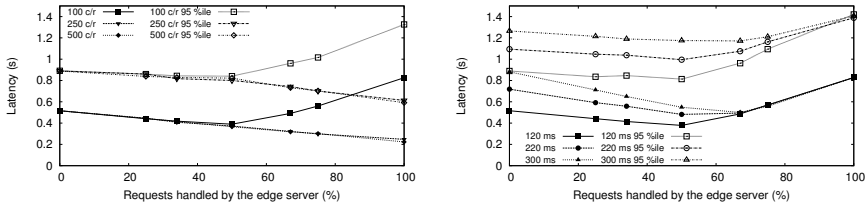
Variable Internet latency. Next, we focus on the impact of Internet latency on the performance of client apps. We consider three distinct Internet latency values: 120 ms, 220 ms and 300 ms, which correspond to the latency between a client in Spain and a server in the UK, the USA and Japan, respectively. These values were empirically obtained using ICMP Echo messages, which were sent at random times during the day. As before, we set the access network latency to 25 ms, while the processing capability of the edge server is now fixed to 100 concurrent requests, with a total load of 500.

Figure 8.2(b) shows that the performance degrades significantly when more requests are sent to the backend server for processing. With edge-assisted execution, however, savings in latency of approximately 300 ms for the worst of conditions are possible. *In summary, the higher the Internet latency, the more beneficial edge-assisted execution becomes. This trend is common for all configurations until the processing capacity of the edge server becomes a bottleneck, i.e. when more than 50% of client requests are handled by the edge server.*

8.4.2.2 Edge server collocated with eNodeB

We repeat the same experiments and now assume that the edge server is collocated with the eNodeB rather than the PGW gateway.

Variable edge server capabilities. Figure 8.3(a) shows the application latency when the processing capacity of the edge server is limited to 100, 250 and 500 concurrent requests. As before, edge-assisted execution improves application responsiveness up to the point at which queuing at the edge server is a major source of delay, i.e. when more than 50% of requests are processed at the edge.



(a) Different number of concurrent requests (c/r) that can be processed simultaneously

(b) Different Internet latencies

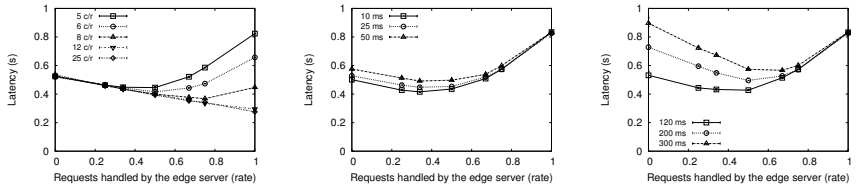
Figure 8.3: Average latency and 95th percentile for different percentages of requests processed by the eNodeB edge server

Variable Internet latency. Figure 8.3(b) shows the achievable performance when varying the Internet latency for this configuration. The results obtained are almost identical to the previous results from Figure 8.2(b)—moving the edge server from the PGW gateway to the eNodeB base station reduces network delays by a negligible amount, thus improving performance only marginally.

In general, there are few differences between the two alternative choices for placing the edge server, i.e. the eNodeB or the PGW. For traffic patterns that cause less congestion at the edge server, a reduction in latency by at most 50 ms (approximately 18% using the online gaming app as reference) is possible by moving the edge server from the PGW to the eNodeB. However, compared to other sources of delay such as the 300 ms Internet latency, the 200 ms average processing delay and queuing delays when the edge server has limited resources, a 50 ms latency reduction is negligible. In addition, one needs to take into account the costs of deploying edge servers at base stations. In any case, this slight performance gain can be negated by the challenges that it causes during handover events (see Section 8.4.2.4).

8.4.2.3 Inter-user delay

We also investigate the effect of the hybrid model on cooperating clients that share a particular service. For example, in online games such as first-person shooter games, multiple clients interact with each other in a shared virtual game world. Each client request thus leads to a broadcast of the corresponding response to all clients. For this experiment, we measure the inter-user delay, i.e. the time for a request to be sent and processed plus the time to propagate



(a) Different edge server processing capabilities (b) Different access network latencies (c) Different Internet latencies

Figure 8.4: Latency of broadcast events for different percentages of requests processed by the edge server

any state changes to all other affected users. We set the number of cooperating clients to 25 in order to avoid extra delays due to network saturation. The remaining configuration parameters are set according to the values used in the previous section.

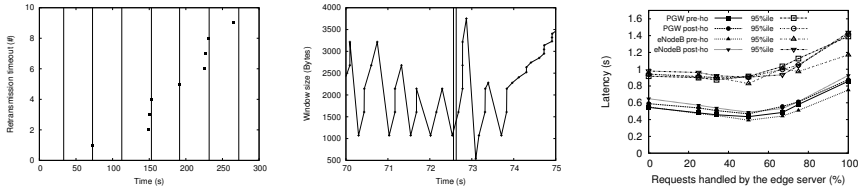
Figure 8.4(a) shows the average time for clients to receive updates due to a request made others, when the server is placed at the PGW. As before, we vary (1) the number of concurrent requests that can be processed by the edge server; and (2) the ratio of requests that can be served entirely at the network edge. The results follow closely those described in the previous experiments, suggesting that *this scenario does not create any asymmetries that change the average delay drastically.* The error bars indicate the maximum and minimum delays experienced, showing that the variability in inter-user delay is low.

A similar trend is observed in Figure 8.4(b). We assume that the edge server is capable of processing a maximum of 20% of the overall concurrent requests issued by the clients. In Figure 8.4(c), we repeat the same experiment but for varying Internet latencies (the access network latency is set to 25 ms). *The results indicate that the higher the Internet latency, the more beneficial edge-assisted execution becomes.*

8.4.2.4 Inter-cell mobility and handover events

Next we investigate the impact of mobility on application latency under the hybrid deployment model. We run a set of experiments that simulate a single handover event to understand how the additional delays introduced affect performance.

We assume that a mobile client roams between adjacent cells at 20 m/s and waits for 20 s in each cell before moving again. We assume a total of 500 mobile



(a) Retransmission timeouts during handover events

(b) Window evolution in a handover event

(c) Average latency and 95th percentile for different percentages of requests

Figure 8.5: TCP behaviour during handover events

clients, initially split equally among two cells. A client can be in either of two states during simulation: (1) the *pre-handover state* in which a client is connected to the originating cell before the handover event takes place; and (2) the *post-handover state* in which the client is connected to the destination cell after the handover finishes. We set the access network latency to 25 ms and the Internet latency to 120 ms. We vary both the number of concurrent requests that an edge server can handle in parallel and the percentage of requests that are processed at the edge server.

Figure 8.5 shows the TCP behavior during a simulation with multiple handover events. Figure 8.5(a) displays handover events as vertical lines and RTO timeouts as distinct points in the graph. Several retransmissions occur close to most handover events. The retransmissions are mostly due to the bad quality of the channel at the boundaries of a cell, with many of them occurring even before the handover initiates. Figure 8.5(b) shows the TCP window during a handover event, which is initiated 72.5 s in the experiment. The vertical lines illustrate the handover's start and end times. As shown in the figure, the server's TCP window increases for almost half a second after the handover event starts, therefore, it is unlikely that this is what causes the subsequent timeout. *We infer that the impact of handovers on TCP performance is small, with the impact of packet losses due to bad reception at the boundaries of a cell being far more important.*

Figure 8.5(c) presents the latency before and after a handover event for the online game traces. We observe a smooth transition between the two base cases, i.e. when all requests are processed at the edge server or they are directed to the backend server for processing. For the latter, both edge server placements yield similar results in the pre-handover state. In the post-

handover state, however, the eNodeB placement exhibits a larger degradation in performance compared to the PGW placement. This is due to the fact that each request must traverse the access network three times: once for the request to reach the SGW; once to be forwarded to the original eNodeB; and one last time to reach the backend server. In contrast, when the server is placed at the EPC, packets go through the access network only once.

As shown in previous sections, when all requests are handled at the network edge, better performance is achieved in pre-handover conditions for the eNodeB placement. However, in post-handover conditions, performance is worse for some applications because packets must traverse the access network twice to reach the edge server on the other eNodeB.

The above results reinforce the observation that there is no significant gain in placing the edge server at the eNodeB instead of the PGW gateway. In fact, the marginal performance improvement comes at the cost of degraded performance when handovers occur.

8.4.2.5 Latency breakdown

Figure 8.6 shows the breakdown of end-to-end application latency for the online gaming network traces, when the edge server is able to process 100 concurrent requests in parallel and 50% of the client requests are processed at the network edge. The backhaul and Internet latencies are set to 25 ms and 120 ms, respectively.

When requests are processed at the edge server, the eNodeB placement slightly outperforms the PGW placement because it avoids sending data through the access network. When requests are processed by the backend server, the application latency is significantly higher because requests and responses are sent to/from the distant backend server.

Figure 8.6 shows that with a hybrid edge-assisted deployment model, processing delays become by far the largest contributors to application latency. This is not the case for the cloud-assisted model, for which network delays due to request/response delivery times dominate performance. Overall, an improvement of approximately 40-45% in performance is possible when approximately 50% of client requests are served at the network edge, despite multiple users competing for the comparatively limited computational resources available there.

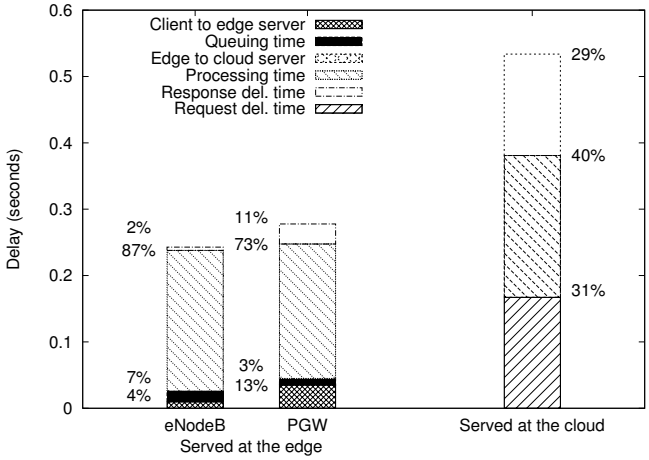


Figure 8.6: Breakdown of application latency

8.5 Conclusions

In this chapter, we investigated the feasibility of a hybrid edge-assisted service deployment model, which supports the hosting of real-time services jointly on cloud- and edge-based servers. This model can help reduce application latencies over LTE for many mobile clients users that share an edge infrastructure, despite the comparatively limited resources available at the edge. Our simulation-based experiments provide insights regarding the conditions for this model to yield performance gains with many mobile clients.

Contrary to existing wisdom in mobile edge cloud proposals, we showed that there is no advantage in hosting services at servers collocated with the eNodeB as opposed to the PGW because the performance gain is not enough to compensate for additional deployment costs involved. Moreover, we showed that a combination of both locally and remotely served requests is required in order to realise benefits in performance—to avoid contention on resource-constrained edge servers, up to 50% of the total requests should be processed at the network edge. Finally, we confirmed that the impact of user mobility on TCP is negligible in such a scenario.

Acknowledgments

This work was supported by the *Ministerio de Ciencia e Innovación*, Spain, under Grant TIN2011-27543-C03-01, and the *Ministerio de Educación, Cultura y Deporte*, Spain, under Grant AP2010-4397. Further support came from *EIT Digital* under the Future Cloud Action Line through the *MC-DATA* project. The work is a result of a mobility stay funded by the Erasmus Mundus Programme of the European Commission under the Transatlantic Partnership for Excellence in Engineering (TEE) Project.

Bibliography

[1] Eric Y. Chen and Mistutaka Itoh. Virtual Smartphone Over IP. In *WoWMoM*, 2010.

[2] ByungGon Chun, Sunghwan Ihm, et al. CloneCloud: Elastic Execution between Mobile device and Cloud. In *EuroSys*, 2011.

[3] Eduardo Cuervo, Aruna Balasubramanian, et al. Maui: Making Smartphones Last Longer with Code Offload. In *MobiSys*, 2010.

[4] Mark S. Gordon, D. Anoushe Jamshidi, et al. COMET: Code Offload by Migrating Execution Transparently. In *OSDI*, 2012.

[5] IBM. *IBM ASPN*, 2013. `http://www-03.ibm.com/press/us/en/pressrelease/40490.wss`.

[6] Intel. *Increasing Mobile Operators' Value Proposition With Edge Computing*, 2014. `https://networkbuilders.intel.com/docs/Nokia_Solutions_and_Networks.pdf`.

[7] Sokol Kosta, Andrius Aucinas, et al. ThinkAir: Dynamic Resource Allocation and Parallel Execution in the Cloud for Mobile Code Offloading. In *INFOCOM*, 2012.

[8] Markus Laner, P. Svoboda, et al. A comparison between one-way delays in operating HSPA and LTE networks. WiOpt, 2012.

[9] W. Matthews and L. Cottrell. The PingER project: active Internet performance monitoring for the HENP community. *IEEE Commun. Mag,*, 2000.

[10] MEC. *Executive Briefing - Mobile Edge Computing (MEC) Initiative*, 2014. `https://portal.etsi.org/Portals/0/TBpages/MEC/Docs/MEC%20Executive%20Brief%20v1%2028-09-14.pdf`.

[11] Nokia Networks. Nokia Siemens Networks, Intel work together to transform mobile broadband experience. `http://goo.gl/vC1anR`, 2013.

[12] Nokia Networks. Cloudlets: At the Leading Edge of Mobile-Cloud Convergence, 2014.

[13] Andreas Pamboris and Peter Pietzuch. EdgeReduce: Eliminating Mobile Network Traffic Using Application-Specific Edge Proxies. In *MobileSoft*, 2015.

[14] Mahadev Satyanarayanan, Paramvir Bahl, et al. The Case for VM-Based Cloudlets in Mobile Computing. *Pervasive Comput.*, 2009.

[15] Shiqiang Wang, Guan-Hua Tu, et al. Mobile Micro-Cloud: Application Classification, Mapping, and Deployment. In *AMITA*, 2013.

[16] Xiaofei Wang, Hyunchul Kim, et al. Measurement and Analysis of World of Warcraft in Mobile WiMAX Networks. In *NETGAMES*, 2009.

Chapter 9

Measurement-Based Modelling of LTE Performance in Dublin City

Miguel Báguena Albaladejo, Douglas J. Leith, and Pietro Manzoni. *Measurement-based modelling of LTE performance in dublin city.* 2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Valencia, 2016

Abstract

LTE/4G is the next generation of cellular network which specifically aims to improve the network performance for data traffic and is currently being rolled out by many network operators. We present results from an extensive LTE measurement campaign in Dublin, Ireland using a custom performance measurement tool. Performance data was measured for two local operators at a variety of locations within the city (including cell edge locations, indoors, outdoors etc) as well as for mobile users on public transport within the city, too. Using this data we derive a model of the characteristics of link layer RTT and bandwidth vs link signal strength. This model is suited to use for performance evaluation of applications and services, and since it is based on real measurements it allows realistic evaluation of performance.

9.1 Introduction

LTE/4G has undergone rapid rollout and now plays a major role in cellular communications, particularly within urban areas. The LTE link layer is, however, complex and contains numerous design parameters the choice of which is left proprietary. The complexity of the LTE MAC makes simulation computationally demanding, while the many unspecified design parameters make selection of realistic configurations problematic. Further, although there have been a number of measurement studies of transport and application layer performance over LTE, measurement studies of actual LTE link layer behaviour in the field are much more limited.

Motivated by these observations, in this chapter we present results from a large-scale LTE measurement campaign carried out in Dublin, Ireland. These measurements provide an accurate snapshot of the service actually offered to the users and the measurement tools developed therefore potentially provide a valuable source of information for operators and users.

Using this measurement data we derive an empirical probabilistic model of the characteristics of link layer RTT and bandwidth, taking account of the changes in the distribution of RTT and bandwidth with link quality and choice of network operator.

Our hope is that this model will assist the research community in realistic, yet reproducible, performance evaluation of new tools and applications. Since this empirical model is much simpler than detailed LTE simulation models, it should be more convenient for evaluation of application layer performance and for evaluation over longer time scales (where simulation would take too long). Since the model is based on measurements from actual LTE network deployments, it reflects realistic configurations.

9.2 Related work

Since LTE is a relatively new technology, the existing literature on performance of cellular networks mostly deals with earlier or alternative cellular technologies [17, 5] and on particular measurements of specific physical-level characteristics or events [11, 13].

A number of other studies measure throughput and/or latency information, but tend to focus on their impact on TCP or its derivatives [10, 7, 15, 18, 12, 6, 14]. Additionally, there are studies that present measurements but do not use these to develop a model suited to performance evaluation [16].

There has been work on simulation [1, 3, 2] and modelling [8, 9]. However,

such tools may be of limited utility because they do not provide the emulation needed to evaluate real-time aspects of actual applications, the models may be overly specific, or they may be at too low layer within the network stack.

Our study is different from the aforementioned work in a number of way, including because it focuses on LTE, it studies performance independently from TCP, and because our explicit aim is to build realistic yet simple models useful for evaluating a wide range of protocol and application performance. Since information is presented in terms of commonly used performance indicators (bandwidth and delay), channels with the approriate characteristics can be readlity emulated for testing of real applications now available on the market.

9.3 Experimental setup

Hardware. We use two Asus EeePC 4G (Intel Celeron M 353 - 900MHz, 512 MB RAM) laptops as cellular UEs for carrying out measurements, each equipped with a 4G USB modem (Alcatel One Touch L100V). These run Lubuntu 14.04 and use ModemManager v1.0.0 to connect to the cellular network. This tool uses libqmi (1.4.0) to operate the modem and it was also used to set the operating mode of the modem to 4G. The qmi_wwan driver in the Linux kernel 3.16.0_30 manages the modem. The modem internal state is gathered by using the Hayes command set. GPS is used to measure the UE location.

Measurement Tool. To measure link characteristics we used a custom tool. This connects with a remote server located in Trinity College Dublin (which is located in the city centre) and this server tests the downlink connection to the UE every second by performing a two-phase test (half a second each phase): 1) in the first one, the server sends a burst of fully loaded (1470 Bytes) UDP packets (50 packets) back-to-back to the UE

Measurement Campaign. Performance data was measured at a variety of locations within the city (including cell edge locations, indoors, outdoors etc) as well as for mobile users on public transport within the city, see Figure 9.1.

We selectd<two operators (T and M) which have its own cellular infrastructure. We selected them because they handle around the 60% of subscriptions in Ireland.

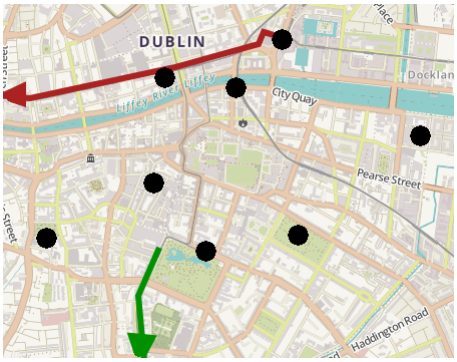


Figure 9.1: Map indicating measurement locations (black dots) and tram lines (in red and green) in Dublin.

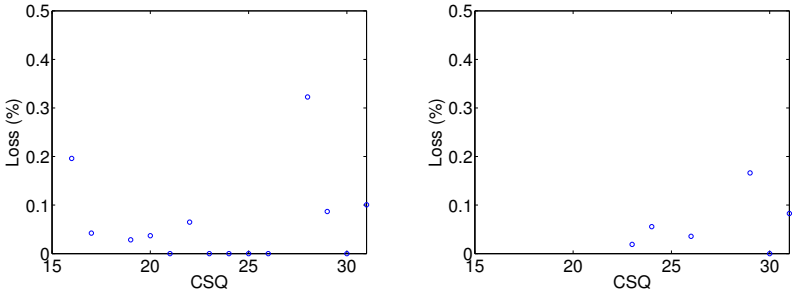
BW	μ	σ
100	99.22	7.11
75	74.13	4.12
50	48.76	1.61
25	24.17	0.37
10	9.77	~ 0
1	0.98	~ 0
0.5	0.49	~ 0

(a) Bandwidth (Mbps)

RTT	μ	σ
1000	1000.54	0.05
750	750.57	0.04
500	500.62	0.03
250	250.65	0.03
100	100.68	0.09
75	75.67	0.05
50	50.68	0.08
25	25.67	0.05
10	10.68	0.05
5	5.68	0.04

(b) RTT (ms)

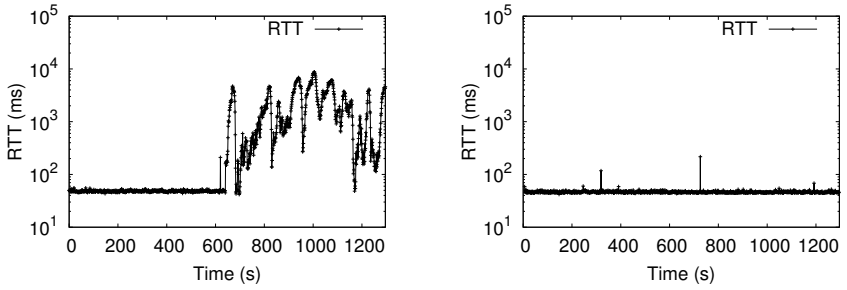
Table 9.1: Accuracy of measurement tool, lab measurements (μ denotes mean value, σ standard deviation).



(a) Operator T

(b) Operator M

Figure 9.2: Measured loss rate vs CSQ. Stationary UE.



(a) UE downloading file (b) Other UE

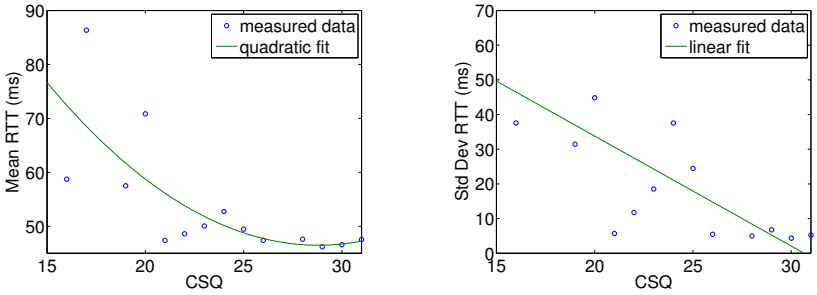
Figure 9.3: RTT measured at two UEs while one is downloading a large file. Both UEs connected to the same network cell and located beside one another.

9.4 Delay

In this section we show that the RTT over the LTE wireless hop can be modelled as i.i.d and following a mixture of Gaussians whose parameters depend on the link quality. Key to this is the observation that UE traffic is queued separately at the LTE basestation and so link layer RTT can be inferred despite fluctuations in cell traffic load.

9.4.1 Per-UE Queueing at Basestation

We placed two UEs side by side. On one UE we then started a large file download while simultaneously measuring the bandwidth and RTT on both UEs. Figure 9.3 plots the RTTs measured by the two UEs. It can be seen that on the UE carrying out the download the measured RTT increases from around 40ms to greater than 1000ms after the download starts. In contrast, for the second UE the RTT is unchanged. The data shown is representative of that measured at a number of locations. We infer that traffic for different UEs is queued is queued separately at the cellular basestation, with the inceased delay measured by the downloading UE being due to queueing delay at the basestation downlink for that UE. This also indicates that RTT measurements made by an unloaded UE (*i.e.* one which is not generating enough traffic to cause queueing) can be expected to be insensitive to network load and so potentially provide insight into the characteristics (scheduler latency, ARQ *etc*) of the wireless link layer.



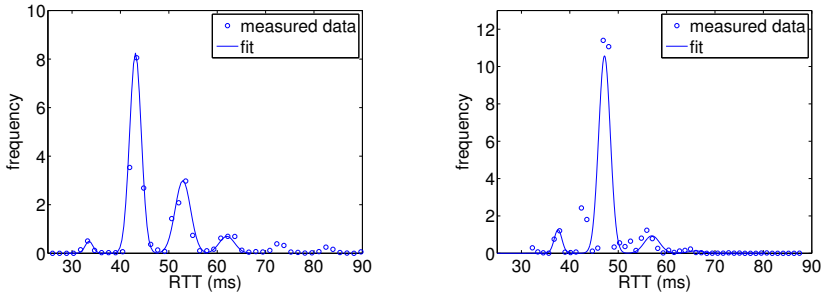
(a) Mean (b) Standard deviation

Figure 9.4: RTT mean and standard deviation, plus quadratic and linear fits respectively. Stationary UE, operator T.

9.4.2 RTT vs Link Quality: Operator T

The UE modem reports a CSQ value which indicates the RSSI on the wireless downlink. According to [4], this scale goes from 0, which is equivalent to -113 dbm (or less), and ends on 31, which is equivalent to -51 (or greater). In our measurements we observed CSQ values in the range 10-31. Figure 9.4 plots the mean and standard deviation of RTT vs CSQ for a stationary UE. This data was collected from six different locations in order to obtain a range of link qualities. It can be seen that the mean delay increases slowy as the CSQ (and so RSSI) falls, and similarly for the standard deviation of delay. The mean and standard deviation variation with CSQ can be modeled by, respectively, $\mu = 177.69 - 9.11 \times CSQ + 0.158 \times CSQ^2$ and $\sigma = 97.21 - 3.17 \times CSQ$ (these fits are indicated on Figure 9.4).

In more detail, Figure 9.5 shows two typical distribution of RTTs measured by a stationary UE. It can be seen that the RTT distribution is multi-modal, and on further inspection the intervals between the first peak and the second is 9.6ms, the first and the third is 19.4ms, the first and the fourth is 28.5ms. That is, the intervals between the peaks are 9.6ms, 9.8ms, 9.1ms. These spacings are consistent across measurements at different times and locations and so seem to be a genuine feature of the LTE MAC. Since the HARQ retransmit time in LTE is at least 8ms, one hypothesis is that the peaks correspond to packets transmitted with no retry, with two retries, with three retries etc. It can be seen that on the link with lower signal quality (Figure 9.5 (a)) there are additional peaks at around 73ms and 82ms and it is these which lead to the increase in the mean and variance of the RTT seen in Figure 9.4.



(a) Distribution, CSQ 19 (b) Distribution, CSQ 31

Figure 9.5: Measured RTT. Stationary UE, operator T.

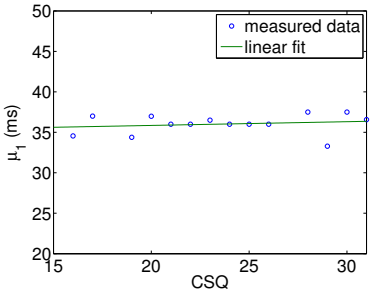
Each peak in the RTT distribution can be well fitted by a mixture of Gaussian distributions $\sum_{i=1}^{4} w_i N(\mu_i, \sigma_i)$, where $N(\mu, \sigma$ denotes a Gaussian distribution with mean μ and standard deviation σ, as indicated on Figure 9.5. The mean and standard deviation parameters of the Gaussians in the mixture are quite consistent across runs in our data and are detailed in Figure 9.6. The RTT μ_1 at which the first peak occurs does vary across data sets, ranging from 32.5ms to 37.5ms, but this variation does not seem to be correlated with link quality, see Figure 9.6(b). The main changes with CSQ are in the weights w_i, $i = 1, \cdots, 4$ of the components of the mixture and linear fits for these are detailed in Figure 9.6(a). This corresponds to a one-way latency of around 15ms, which seems to consistent with the LTE literature and is likely associated with a mix of wireless MAC delays and delay on the cellular wired backhaul Figure 9.7(a) plots the RTT autocorrelation. It can be seen that the RTT values show evidence of correlation over intervals of about 10 samples. However, the correlation decays quite quickly and so it is probably reasonable to model the RTTs as being i.i.d, at least to a first approximation.

9.4.3 RTT vs Link Quality: Operator M

We found that the measured RTT characteristics varied somewhat with the operator. The foregoing data is for operator T. When using operator M, Figure 9.8(b) plots the autocorrelation of the data shown in Figure 9.8(a). It can be seen that the RTT measurements are essentially periodic, with period 10 samples. This is in contrast to operator T where the RTT measurements show much less correlation. We are unsure as to the source of this periodicity, but it is consistent across all measurements with operator M and so presumably

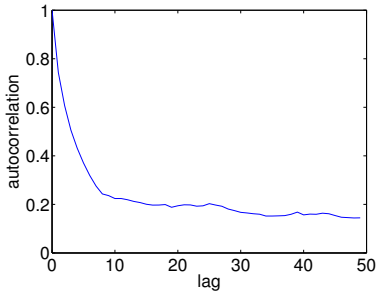
i	μ_i	σ_i
1	μ_1	0.02
2	$\mu_1 + 9.6$	0.03
3	$\mu_1 + 19.4$	0.04
4	$\mu_1 + 28.5$	0.04
	w_i	
1	$0.0079CSQ - 0.1285$	
2	$0.0430CSQ - 0.3834$	
3	$-0.0059CSQ + 0.3244$	
4	$-0.0096CSQ + 0.2937$	

(a) μ_i, σ_i and w_i

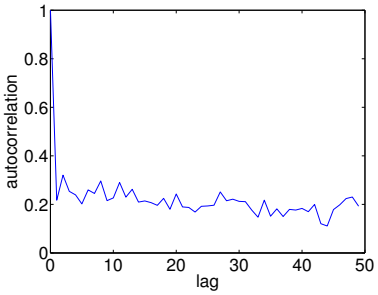


(b) μ_1 vs CSQ

Figure 9.6: Parameters of mixture of Gaussians fit to measured RTT distribution.

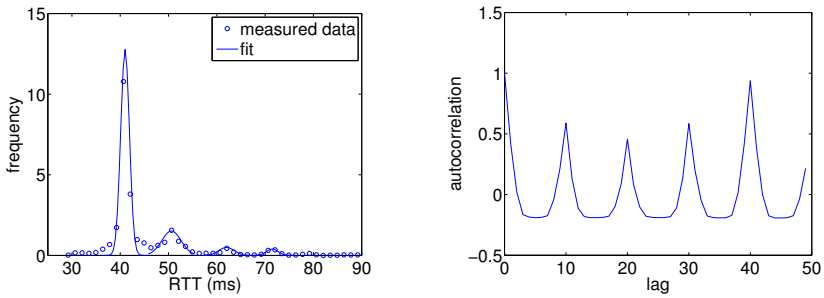


(a) Stationary UE



(b) UE on moving tram.

Figure 9.7: Autocorrelation of measured RTT for a stationary UE and a moving UE. Operator T, CSQ 19. Stationary data corresponds to that in Figure 9.5(a).



(a) Distribution (b) Autocorrelation

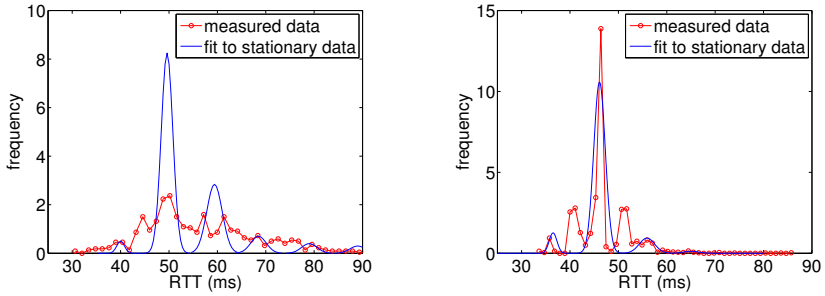
Figure 9.8: Measured RTT for a stationary UE with CSQ of 27, operator M.

reflects the scheduling strategy used.

9.4.4 Impact of Mobility

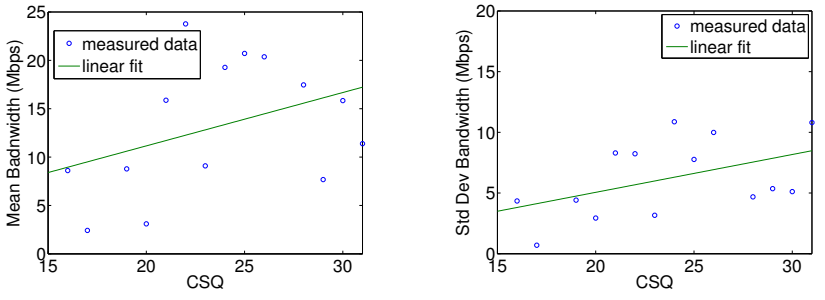
We also collected data for both operators on mobile UEs travelling on two tram lines within the city (data for both operators was collected simultaneously using two UEs). For both operators this data showed very similar RTT mean and standard deviation vs link quality behaviour to the above data for a stationary UE. RTT data for both operator exhibits low autocorrelation, see Figure 9.7(b) for the autocorrelation with operator T. Note that for operator M this is unlike the situation for a stationary UE, where the RTTs are observed to be essentially periodic.

In addition to removing the periodicity from operator M RTTs, mobility also changes the RTT distribution. Figure 9.9 plots the measured RTT distribution for a moving UE with operator T and links CSQs of 19 and 31. Also shown is the fit to the RTT distribution for a stationary UE with operator T and link CSQs of 19 and 31, taken from Figure 9.5. For a CSQ of 31 (a high quality link), the RTT distribution is much the same as in the stationary measurements. However, for a CSQ of 19, the RTT distribution is somewhat "smeared out" for the moving UE. For the moving UE the plot here is generated by taking all RTT measurements for which the CSQ is 19. While RTT measurements are taken at 50ms intervals, the CSQ is only updated once per second, and since the CSQ is changing as the tram moves some of the RTT data plotted may in fact belong to a nearby CSQ value. As CSQs increase above 19 the RTT distribution changes the weights on the different peaks, see Figure 9.5, and so this mixing of data may explain Figure 9.9(a).



(a) CSQ 19 (b) CSQ 31

Figure 9.9: Measured RTT for a moving UE, operator T.



(a) Mean (b) Standard deviation

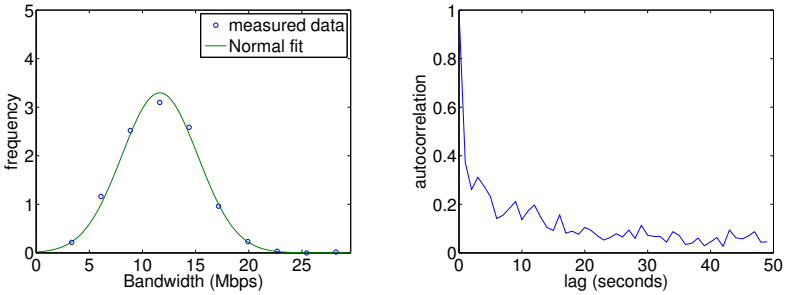
Figure 9.10: Mean and standard deviation of measured bandwidth vs CSQ, plus linear fits. Operator T.

9.5 Bandwidth

9.5.1 Bandwidth vs Link Quality: Operator T

Figure 9.10 plots the mean and standard deviation of the bandwidth measured by a stationary UE at locations with a range of link qualities. It can be seen that the mean bandwidth increases with link quality, rising from around 10Mbps for links with a CSQ of 15 to around 20Mbps for links with a CSQ > 30.

In more detail, Figure 9.11(a) plots the distribution of bandwidth measured on a link with CSQ=31, and also a Gaussian distribution fitted to this data.

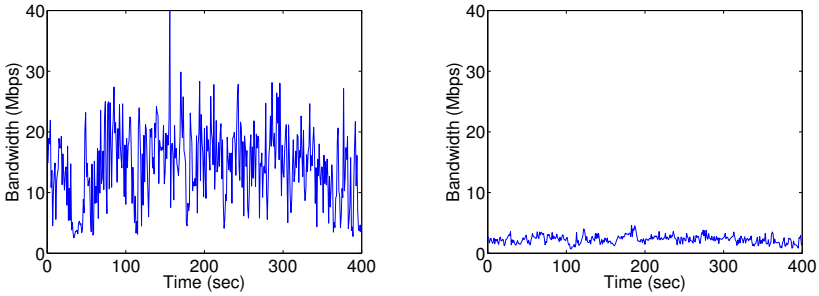


(a) Distribution (b) Autocorrelation

Figure 9.11: Measured bandwidth. Data shown in both plots is for the same link with CSQ 31. Operator T.

It can be seen that a Gaussian distribution, truncated to be non-negative, provides a reasonable fit to the bandwidth measurements. It can be seen from Figure 9.11 that the mean and standard deviation variation with CSQ can be approximately modelled as $\mu = 0.55 \times CSQ + 0.13$ for the mean and $\sigma = 0.31 \times CSQ - 1.17$ for the standard deviation. Figure 9.11(b) plots the autocorrelation of the measured bandwidth. It can be seen that although the bandwidth values show more correlation than the RTT data, the correlation is still quite weak and so the bandwidth probably reasonably modelled as i.i.d.

Observe from Figure 9.10(b) that the standard deviation of the bandwidth increases as the link quality increases, which is quite surprising (we might have expected variability to decrease with increasing link quality if it is associated with link layer retransmissions to recover from loss). This effect is also evident in Figure 9.12 which shows two example time histories of measured bandwidth, one at a location where the link quality is good (CSQ is 30) and one where the link quality is poorer (CSQ 18). We note that, although it is hard to confirm the level of cell load, we believe that these measurements are for lightly loaded cells and so the variability is *not* due to changing cell traffic load. The source of this variability is not clear at present, but we suspect that it is associated with the time granularity of scheduling updates by the cellular basestation since we note that the ratio of the standard deviation to the mean bandwidth is approximately constant with CSQ (assuming changes in bandwidth are mainly due to changes in the modulation and coding scheme used, then timing granularity would result in bandwidth fluctuations that scale with bandwidth, as observed).



(a) CSQ=30 (b) CSQ=18

Figure 9.12: Time histories of measured bandwidth. Stationary UE, operator T.

9.5.2 Bandwidth vs Link Quality: Operator M

Figure 9.13 plots the mean and standard deviation of the measured bandwidth vs CSQ for operator M. Also shown on the plots are the linear fits from operator T (i.e. the lines are the same as on Figure 9.10). It can be seen that the behaviour is similar to that of operator T. With operator M, the distribution of bandwidth values is also well approximated by a Gaussian distribution, see Figure 9.14(a). However, perhaps unsurprisingly in light of the periodicity observed in the RTT data for this operator, the bandwidth measured shows significant periodicity, see Figure 9.14(b).

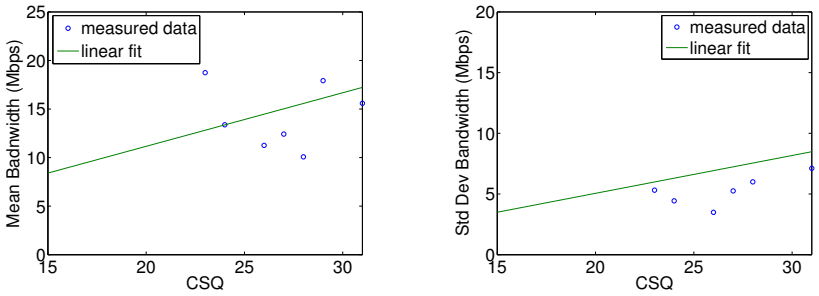
9.5.3 Impact of Mobility

Similarly to the case with RTT, for both operators bandwidth data collected using mobile UEs travelling on two tram lines within the city showed very similar behaviour to the above data for a stationary UE.

9.6 Empirical LTE Model

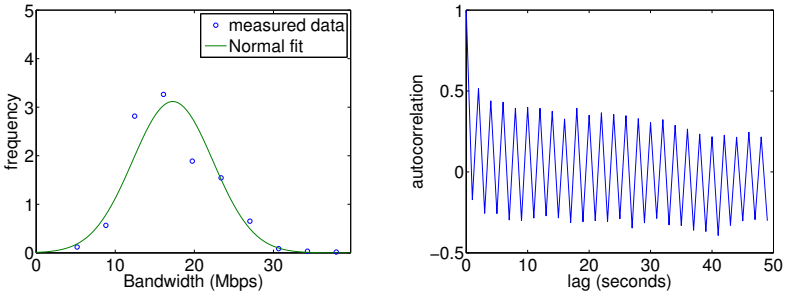
It can be seen from this measured data that a relatively simple empirical model of LTE RTT and bandwidth can be obtained, as follows.

1. For uplink and downlink there is per UE queueing, so queueing delay is decoupled between UEs sharing the same cell.



(a) Mean (b) Standard deviation

Figure 9.13: Mean and standard deviation of measured bandwidth vs CSQ, plus linear fits. Operator M.



(a) Distribution, CSQ=31 (b) Autocorrelation

Figure 9.14: Measured bandwidth. Data shown in both plots is for the same link with CSQ 31. Operator M.

2. RTT is modelled as an i.i.d mixture of Gaussians $\sum_{i=1}^{4} w_i N(\mu_i, \sigma_i)$, where for a given CSQ the parameters w_i, μ_i, σ_i are given by Figure 9.6.

3. Bandwidth is modelled as i.i.d Gaussian distributed, with distribution parameters $\mu = 0.55 \times CSQ + 0.13$ and $\sigma = 0.31 \times CSQ - 1.17$. That is, to generate a realistic sequence of bandwidth values for a link with a given CSQ value, draw a sequence of i.i.d Gaussian values distributed with pdf $e^{-(x-\mu)^2/(2\sigma^2)}/(\sigma\sqrt{2\pi})$ where $\mu = 0.55 \times CSQ + 0.13$ and $\sigma = 0.31 \times CSQ - 1.17$.

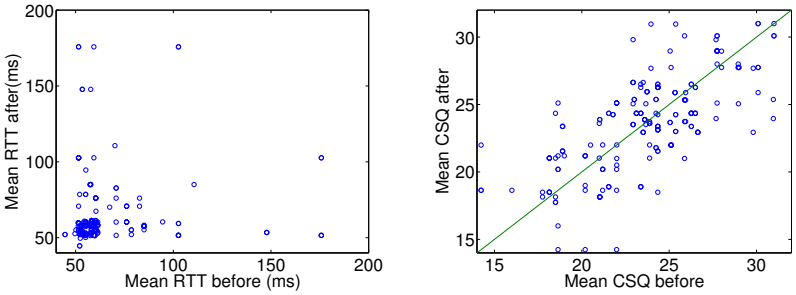
The LTE link layer is complex and contains numerous design parameters the choice of which is left proprietary. The complexity makes simulation computationally demanding, while the many unspecified design parameters make selection of realistic configurations problematic. Since the above empirical model is much simpler than detailed LTE simulation models, it should be more convenient for evaluation of application layer performance and for evaluation over longer time scales (where simulation would take too long). Since the model is based on measurements from actual LTE network deployments, it evidently reflects realistic configurations.

9.7 Summary

We present results from an extensive LTE measurement campaign in Dublin, Ireland using a custom performance measurement tool. Performance data was measured at a variety of locations within the city (including cell edge locations, indoors, outdoors etc) as well as for mobile users on public transport within the city. Using this data we derive a model of the characteristics of link layer RTT and bandwidth vs link signal strength. This model is suited to use for performance evaluation of applications and services, and since it is based on real measurements it allows realistic evaluation of performance.

9.8 Handover

While our measurement campaign provided extensive data on handover, we leave modelling of handover between cells to future work owing to lack of space. We do, however, note briefly that the handover strategy used by an operator can have a significant impact on performance and can vary between operators. For example, Figure 9.15(a) plots the mean RTT measured by a



(a) RTT (b) CSQ

Figure 9.15: Measured RTT and CSQ before and after handover. UE travelling on tram, operator M.

moving UE (travelling on a tram) before cell handover and afterwards. Figure 9.15(b) plots the corresponding CSQ before and after handover. It can be seen from Figure 9.15(b) that handovers tend to occur to cells with similar CSQ, as might be expected. From Figure 9.15(a) we can see that the mean RTT before and after handover tends to be bunched around 50ms, in line with our other measurements above. However, Figure 9.15(b) also shows that sometimes handover greatly increases the mean RTT, *e.g.* from 50ms to 180ms. Such a large mean RTT is indicative of a problematic link and we observe that handovers to more poorly performing cells need not be infrequent.

Bibliography

[1] LENA website. `http://networks.cttc.es/mobile-networks/software-tools/lena/`, 2011.

[2] 4GSim website. `https://github.com/4gsim/4Gsim`, 2014.

[3] SimuLTE website. `http://simulte.com/`, 2014.

[4] 3GPP. Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; AT command set for User Equipment (UE). TS 27.007 version 8.15.0 Release 8, 2012.

[5] D. Baltrunas, A. Elmokashfi, and A. Kvalbein. Measuring the reliability of mobile broadband networks. In *IMC '14*, pages 45–58, New York, NY, USA, 2014. ACM.

[6] Y.-C. Chen, Y. Lim, R. J. Gibbens, E. M. Nahum, R. Khalili, and D. Towsley. A measurement-based study of multipath tcp performance over wireless networks. In *IMC '13*, pages 455–468, New York, NY, USA, 2013. ACM.

[7] S. Deng, R. Netravali, A. Sivaraman, and H. Balakrishnan. Wifi, lte, or both?: Measuring multi-homed wireless internet performance. In *IMC '14*, pages 181–194, New York, NY, USA, 2014. ACM.

[8] R. Gupta, B. Bachmann, R. Ford, S. Rangan, A. Morelli, V. Mancuso, N. Kundargi, and A. Ekbal. Demo: Labview based framework for prototyping dense lte networks. In *WiNTECH '14*, pages 91–92, New York, NY, USA, 2014. ACM.

[9] C.-J. Hsu, J. L. Pino, and F.-J. Hu. A mixed-mode vector-based dataflow approach for modeling and simulating lte physical layer. In *DAC '10*, pages 18–23, New York, NY, USA, 2010. ACM.

[10] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck. An in-depth study of lte: Effect of network protocol and application behavior on performance. In *SIGCOMM '13*, pages 363–374, New York, NY, USA, 2013. ACM.

[11] R. Merz, D. Wenger, D. Scanferla, and S. Mauron. Performance of lte in a high-velocity environment: A measurement study. In *AllThingsCellular '14*, pages 47–52, New York, NY, USA, 2014. ACM.

[12] B. Nguyen, A. Banerjee, V. Gopalakrishnan, S. Kasera, S. Lee, A. Shaikh, and J. Van der Merwe. Towards understanding tcp performance on lte/epc mobile networks. In *AllThingsCellular '14*, pages 41–46, New York, NY, USA, 2014. ACM.

[13] V. M. Nguyen, C. S. Chen, and L. Thomas. A unified stochastic model of handover measurement in mobile networks. *IEEE/ACM Trans. Netw.*, 22(5):1559–1576, October 2014.

[14] M. Z. Shafiq, J. Erman, L. Ji, A. X. Liu, J. Pang, and J. Wang. Understanding the impact of network dynamics on mobile video user engagement. In *SIGMETRICS '14*, pages 367–379, New York, NY, USA, 2014. ACM.

[15] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, S. Venkataraman, and J. Wang. A first look at cellular network performance during crowded events. *SIGMETRICS Perform. Eval. Rev.*, 41(1):17–28, June 2013.

[16] J. Sommers and P. Barford. Cell vs. wifi: On the performance of metro area mobile connections. In *IMC '12*, pages 301–314, New York, NY, USA, 2012. ACM.

[17] F. P. Tso, J. Teng, W. Jia, and D. Xuan. Mobility: A double-edged sword for hspa networks: A large-scale test on hong kong mobile hspa networks. In *MobiHoc '10*, pages 81–90, New York, NY, USA, 2010. ACM.

[18] Y. Xu, F. Ren, S. Xu, C. Lin, and S. Das. Investigating the interacting two-way tcp connections over 3gpp lte networks. In *MSWiM '12*, pages 39–46, New York, NY, USA, 2012. ACM.

Chapter 10

Discussion

In this chapter we will provide an overview of all the contributions of this thesis, and the role that each one played in it. Additionally, we highlight the most important results per contribution.

10.1 General overview of contributions

Chapters 3 to 9 presented the publications produced by the author of this thesis that show the research results obtained in various aspects of the mobile communications field. Overall, they all focus on a same main issue: creating an efficient and reliable platform to deploy applications in vehicular networks. They can be classified into two big groups: (i) systems and protocols that improve network performance for the services in the upper layers, or (ii) support tools that can be used to test the behavior ad measure the performance of items in (i). In Figure 10.1 we show how all the pieces fit together.

Regarding the proposed protocols and systems, each of them plays a specific role:

- **Raptor-based Content Delivery Protocol (RCDP):** It is an application level transport protocol. It will ideally work over UDP/IP. The role of this protocol is shielding the wireless channel to offer reliable content delivery through it. Since TCP is unable to properly deal with random losses in wireless channels, this protocol will show higher performance in vehicular networks.

- **Adaptive Anycasting solution for Vehicular Environments (AVE):**

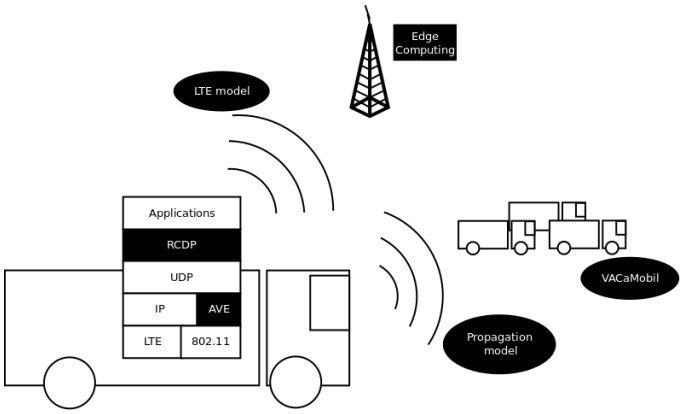


Figure 10.1: Role of all the technologies presented in this thesis.

This is an adaptive routing protocol. It can easily cooperate with any other network or transport protocol. It will provide efficient routes for the packets adapted to the current network conditions. This plays a key role because vehicular networks are highly mutable, so there is no single strategy that can operate efficiently in every network state.

- **Edge computing:** Edge computing based systems are those which improve response time by performing tasks a computer located at the network's edge instead of the main server. This helps reducing latency in communication between nodes that are geographically close. Such approach is particularly interesting in vehicular networks because delay added by the network itself (LTE/802.11p) is high, and so so it is important to reduce it as much as possible.

Regarding the support tools, each of them serves to a specific purpose:

- **Propagation model:** It implements a propagation model for vehicular networks for the OMNeT++ simulator. It helped us to evaluate our tools by offering a specific propagation behavior that takes into account the specific situations that only take place in vehicular networks, such as, moving obstacles.
- **VANET Car Mobility Manager (VACaMobil):** It implements a mobility manager that made us able to accurately control the network population along the simulation. It allows maintaining a realistic car behavior while having a controlled randomized simulation.

- **LTE model:** This is a lightweight stochastic model of an LTE channel that can be used to evaluate applications that are intended to run over such channels.

Below we proceed to highlight the most important findings associated to this thesis for each area described above, showing their importance in the overall scope of the thesis project.

10.2 Raptor-based Content Delivery Protocol (RCDP)

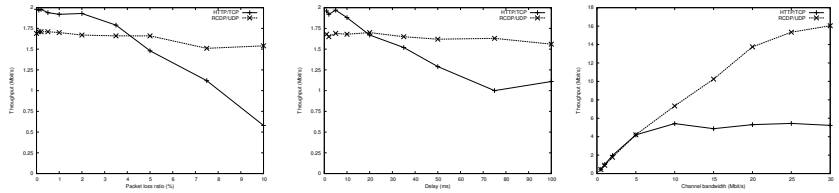
TCP is the most widely used transport protocol for communication. The congestion control of this algorithm is based on the detection of packet losses, which are directly linked to congestion issues in wired networks. However, in wireless networks, random losses may occur due to noise or low link quality. This creates an artificial performance degradation in the network.

Vehicular networks rely on wireless networks, so the impact of this issue becomes relevant. Moreover, the link quality, as well as the noise, are highly mutable on these networks, which is translated into many packet losses and, therefore, into lower network performance.

Some of our research focused on developing a transport protocol that offers a reliable delivery, as TCP does, but with a flow control that is not based on packet loss detection, since it is not reliable in this case. Instead, we used forward correction codes to create a lossless channel, and we detect the optimal transmission rate by sending packet bursts and measuring the packet inter-arrival time.

In order to evaluate our algorithm, we created a small testbed with a couple of computers connected to a small network. We used this network as a black box that could be tunned to show different bandwidth, delay and packet losses. We performed several file transmissions using this network using both our protocol and TCP in order to determine the performance differences.

The results in Figure 10.2 show that both content delivery schemes experience a performance degradation as the channel conditions become worse. However, the performance reduction for the HTTP/TCP solution is, in all cases, much more pronounced than the one for RCDP/UDP. In particular, the transmission rate reduction experienced with HTTP/TCP can be considered excessive when the packet loss ratio or the delay are significant. Such results clearly emphasize the lack of effectiveness of solutions based on HTTP and TCP protocols to handle scenarios where losses are not congestion related.



(a) Throughput vs. packet loss. (b) Throughput vs. delay. (c) Throughput vs. bandwidth.

Figure 10.2: RCDP/UDP vs. HTTP/TCP

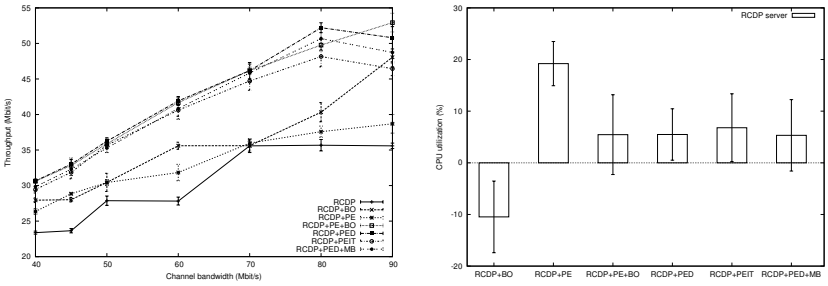
On the contrary, we find that the combination of RCDP and UDP allows maintaining the transmission rate much more stable despite of packet losses, clearly outperforming HTTP/TCP under poor channel conditions. The differences between both content delivery strategies become much more pronounced if the channel bandwidth increases beyond 5 Mbit/s, where we can see that the HTTP/TCP strategy is unable to make an efficient use of the available bandwidth.

Unfortunately, our approach still showed a couple of sensitive points that should be addressed. First, the bandwidth estimation strategy of our protocol should be improved in order to avoid bandwidth wastage and to ensure transmission fairness. Second, the Raptor Codes coding and decoding process are highly resource-demanding.

Focusing on the first of these issues, we remind that RCDP measures the available bandwidth in the channel for each packet burst. However, these values are very noisy. In order to reduce that noise, we implemented and evaluated a series of filters to get a cleaner information on the actual channel state. These filters created a trade off between the bandwidth accuracy and the delay introduced by the filter itself. We built different versions of our protocol and evaluated them in our testbed.

We found that unfiltered estimations showing very low or very high values occur about 13.15% of the times. This value is reduced to less than one per cent when introducing median filters. Regarding the delay, which can be seen as a stabilization delay until the correct bandwidth is detected, excessive smoothing has a negative impact on delay. In our tests, we found that the filter that behaved best was a median filter that uses the last 5 values.

Regarding the second issue, we developed several improvements aiming at reducing the time and the CPU used by the algorithm in coding and de-



(a) Throughput vs. available bandwidth. (b) Additional CPU overhead vs. original.

Figure 10.3: RCDP optimizations

coding. Those improvements were both in implementation (a set of baseline optimizations and the introduction of multithreading) and design (new algorithm behavior) to alleviate the system regarding both latency and CPU usage. Different versions were created and evaluated in our testbed.

In Figure 10.3a, we see that, in general, the different enhancements to RCDP bring benefits in terms of end-to-end throughput. In particular, combining baseline optimizations with parallel encoding allows increasing throughput by up to 46.4% with respect to the original RCDP implementation, which is a very substantial improvement. Besides throughput enhancements, the block pre-charge (PED) technique, along with the increment of buffer size, enables the generation of a continuous symbol flow which contributes to a more regular transmission rate compared to the original RCDP.

With respect to CPU utilization, we find that the complexity of the Raptor encoding process has a clear impact on CPU utilization, especially at the server side where it tends to increase when parallel coding is adopted, although it is partially mitigated by using the set of baseline optimizations proposed. This effect is shown in Figure 10.3b. At the client side, we find that the CPU load is usually quite lower than the server load. When decoding enhancements are introduced, the CPU utilization at the client side increases slightly, while at the server it decreases. This occurs because decoding enhancements are mostly oriented to alleviate CPU stress at the server side.

Focusing on RAM usage, we find that the data structures required to support parallel coding or decoding cause memory consumption to increase significantly. Overall, results show that, despite both client and server share the same architecture, the asymmetry associated to the Raptor encoding and de-

coding tasks results in reduced resource requirements at clients compared to the server side, which becomes more loaded due to the higher complexity of the encoding process.

RCDP offers a better performance than TCP in high delay and lossy environments. These are the typical conditions found in vehicular networks, so this protocol is a perfect fit for them. It is able to increase throughput under the conditions were TCP fails. However, many other pieces must also be also included to optimize communications in vehicular network environments.

10.3 Adaptive Anycasting solution for Vehicular Environments (AVE)

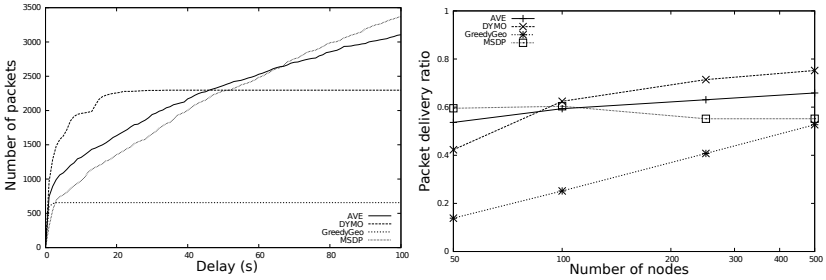
A vehicular network is in general highly mutable along time. Despite it can behave as a static MANET in highly congested scenarios, it typically behaves as a very dynamic network in sparser scenarios. There are routing protocols that can handle any of these states separately. However, there is none able to handle all of them.

In this thesis we proposed a routing protocol that can show different behaviors depending on the current state of the network. Each vehicle gets information from their neighbors using a modified version of the NHDP protocol and, depending on that information, it can use topological or geographical approaches to route packets.

The evaluation approach that we chose was simulation. We implemented our protocol for OMNeT++, as well as, several additional routing algorithms (DTN-focused, geographic, topological). We compare our protocol against the additional ones for different city layouts, and under different vehicle densities. We also implemented other support tools that will be discussed in section 10.5. Then, several experiments were launched, and the packet delivery ratio and the packet delay of the different solutions was retrieved.

As seen in Figure 10.4, we found that AVE outperforms DYMO by 10% in low density scenarios. In dense networks, AVE also outperforms DTN by 10% at the expense of only losing 6% in sparse scenarios. Georouting performance improves when the vehicle density grows, as expected, but the density increment is not enough to outperform the other approaches. The same trend appears for packet delay.

Routing and forwarding in vehicular networks is a different challenge depending on the current network state. The contribution of this protocol to the thesis is to give the system the ability of determining that state, and using the



(a) Delay distribution (b) Packet delivery ratio

Figure 10.4: AVE performance

most suitable strategy to get the most in every situation.

10.4 Edge computing

One of the biggest problems in communication is latency. Nowadays, the user is moving to more interactive applications such as VoIP, video calls, and online gaming, which have tight latency requirements. Vehicular networks are a kind of network characterized by a huge delay budget in any case. If instead the physical link is based on 802.11p, some extra delay is added in situations of multi-hop communication and contention. If the physical link is based on LTE, some extra delay is added because of the medium access protocol and the scheduling algorithm implemented by the operator.

The edge computing paradigm attempts to reduce delay by processing some or all the requests at the edge/access network in order to achieve a quicker response, and avoid delay in reaching the central server. This technique exploits locality of communications by connecting to closer nodes at a city level instead of forcing the communication to be at a planetary scale. In this case, we focused on LTE networks and gaming applications to evaluate the improvement when using this new scheme.

We set up a model for the OMNeT++ simulator based on INETMANET, SimuLTE and 4GSim. The handover feature was missing, and so we implemented it. An additional restriction that this paradigm presented is the fact that the computers at the edge are limited in size and characteristics. Therefore, we modeled this as a maximum number of concurrent requests that a server can handle. We explored the impact of different delay values in both

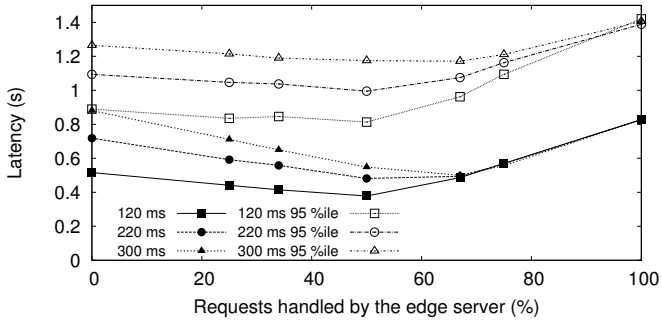


Figure 10.5: Edge computing performance

the access network and the Internet. We also created our traffic from a model based on application traces of popular online games and applications.

Experimental results have shown that, by allowing more requests to be served by the edge server, the high Internet delays are avoided and latency gradually decreases. However, when the processing capacity of the edge server saturates (about 50% of the requests), latency increases due to server queuing to a point where edge-assisted execution is outperformed by the original configuration.

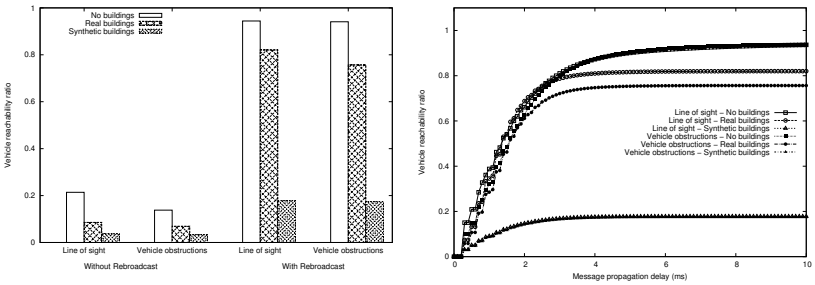
Additionally, we observe that there is not much to gain from placing the edge server at the eNodeB instead of the PGW gateway. In fact, we show that the marginal performance improvement obtained by collocating edge servers with eNodeBs comes at the cost of degraded performance when handovers occur.

Overall, the impact of handovers on TCP performance is negligible, with the impact of packet losses due to bad reception at the boundaries of a cell being far more important.

10.5 Support tools

Part of the research carried out throughout this thesis focused on creating models and tools that helped us and the research community to evaluate and test protocols and systems. In this section we will present all those models and tools.

Evaluating vehicular networks can be quite challenging: since nodes are moving, the transmission constraints can change along time. The usual way of



(a) Reachability ratio (b) Reachability along time

Figure 10.6: City layout impact

addressing this issue is setting up a mobility simulator, SUMO in our particular case, and make the nodes move according to a predefined urban layout. However, this setup leaves two issues still open. The first one is the propagation channel. Several moving and fixed obstacles may impact in the fading and radio coverage. The second one is mobility management. For the sake of making experiments reproducible, we must ensure that several constraints remain unchanged along the whole simulation. Therefore, some more controlled simulation environments must be set.

We will focus on the propagation model first. We developed a new propagation model by combining a visibility model and an attenuation model that is based on experimental measurements. We compared it with a line of sight model in simulation, and got information on vehicle reachability for some dissemination schemes.

For a one-hop dissemination scheme, we found a reduction in reachability for the *Vehicles as obstacles model* near to one third over the *Line-of-sight model* in the worst case. These differences will be a key factor for one-hop beaconing protocols, not only due to the final amount of neighbors detected, but also due to the quality of these links and the possible forwarders in a routing protocol algorithm.

We also found that the final vehicle reachability ratio highly depends on the existing buildings. Simulating a low restrictive (few buildings) scenario with a highly restrictive configuration (a lot of buildings) will introduce output inaccurate results. We have the opposite problem in a high building density scenario. To avoid these problems, accurate building maps are needed.

In Figure 10.6, we show the impact of the propagation model on the broad-

casting scheme (one-hop vs. rebroadcasting). Rebroadcasting strategies evidences the effect of static obstacles because it amplifies its effects. With no rebroadcasting, only the nodes near the sender are reached (one-hop neighbors). However, in the rebroadcasting approach, the building obstruction effect is amplified at every hop, making it more evident. Results show that the flooding process is successful, reaching the majority of the vehicles circulating in the target area, as desirable.

This new model shows a much more accurate behavior based on experimental measurements. This accuracy increment is of utmost importance in our research: since the propagation model can significantly change the results of an experiment, the more accurate our propagation model is, the more confidence we can have in our results.

The second issue is mobility management. SUMO has some mobility management features, such as flow definition, automatic route generation, etc. However, there are several additional features that are desirable to make experiments meaningful.

To this purpose, we developed VACaMobil, a mobility manager that introduces many new features. The most important one is its ability of controlling the total population of nodes in the simulation. It allows us to study the car density as a new additional parameter. This is really important because the performance of many algorithms depends on congestion, which in turn depends on network density. Additionally, with VACaMobil we can easily define different kinds of vehicles, and it includes new tools to generate a route database for each simulated map.

We compared our manager against the current available tools included with SUMO. With the current ones, the user cannot predict when vehicles will arrive to their destination and disappear from the network. Therefore, the number of vehicles when the simulation reaches a steady state (or whether a steady state will be reached at all) is unknown a priory. In our tests, our manager showed itself able to keep the number of vehicles constant, and to properly spread them along the network using the generated route database.

We also needed some tools to perform the experiments using LTE networks. Current solutions to simulate LTE networks do a fine-grained simulation that is expensive in computing resources, particularly in time. This very detailed simulation is not always needed in every case, and we can save resources that can be used for other tasks.

Hence, we created a lightweight LTE model based on experimental measurements. To do so, we took bandwidth and delay measurements in Dublin city in many different places, and we analyzed the results. We generalized those results by building a model with the following characteristics:

- There is per UE queueing for the uplink and downlink, so queueing delay is decoupled between UEs sharing the same cell.
- RTT is modelled as an independent and identically distributed (i.i.d) mixture of Gaussians.
- Bandwidth is modelled as i.i.d Gaussian distribution.

All the parameters of these distributions were tunned accordingly to show the same behavior as the one found in the real network.

This set of tools allowed us to get accurate results, thus being one of the main contributions to this thesis. However, it is also relevant to acknowledge the importance of the tools for the research community because, by using these tools, everybody can evaluate their work.

10.6 Summary

In this thesis, we presented a series of works that aim at increasing efficiency in vehicular networks, creating the perfect environment to deploy different services. Additionally, several tools for performance evaluation in simulation were included, too.

Throughput in vehicular network was dramatically improved by introducing the RCDP protocol. In fact, it is able to overcome the performance downgrade shown by TCP in wireless networks. Thanks to RCDP, the node in a vehicular network can make the most of the bandwidth in the wireless link, even if signal levels are very low.

Latency was also improved in the most widespread communication technologies used in vehicular networks: 802.11p and LTE. For 802.11p, where the multi-hop approach is commonly used, our adaptive routing protocol, AVE, is able to find a better route for all the network conditions. In LTE, our edge computing deployment reduces latency in local communications by computing the answer at a node closer to the source.

Finally, our simulation tools and models create the perfect testbed to test the performance of new tools, making evaluations easier for future researchers. This way, the path for future developments remains solid and open.

Chapter 11

Conclusions

11.1 Final remarks

This thesis focused on creating a set of protocols and systems to improve the performance of services deployed on vehicular networks. It implied research at many layers to raise service levels to meet user demands.

Since vehicular networks are highly mutable, setting up a reliable unicast communication solution is particularly challenging: paths between nodes can change, packets can get lost, and network partitioning can difficult communications. Some solutions have addressed these issues separately, but none of them present a comprehensive approach that handles all the aspects of this problem.

In this thesis we addressed performance in terms of throughput and latency, as well as reliability, to present a better overall solution to improve communication in vehicular networks. We developed a new application-layer transport protocol that creates a loss-resilient channel while keeping a good throughput. Additionally, we included a routing protocol that adapts to the network conditions to reduce latency and improve packet delivery ratio. Finally, we introduced an edge computing scheme to reduce latency for communications between nearby nodes using the deployed infrastructure.

According to the discussion in Chapter 10, the improvement in terms of throughput, latency and reliability was successful, achieving this way the main objective of this thesis.

11.2 Publications related with this thesis

The research related to this thesis was published in selected conferences and journals. They sum up to fifteen publications. We will include all of them here with a very brief description of each of them. We have organized them based on the topic that they belong to:

Publications related to the transport layer

- [2] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. "Design, implementation, and optimization of a raptor-based content delivery protocol." In Proceedings of the IFIP Wireless Days Conference 2011, Niagara Falls, ON, Canada, October 10-12, 2011 , pages 1-5. IEEE, 2011.

- [3] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. "Raptor-based reliable unicast content delivery in wireless network environments." In IEEE 36th Conference on Local Computer Net- works, LCN 2011, Bonn, Germany, October 4-7, 2011 , pages 841-849. IEEE Computer Society, 2011.

- [4] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. "RCDP: A novel content delivery solution for wireless networks based on raptor codes." In Ad-hoc, Mobile, and Wireless Networks - 11th International Conference, ADHOC-NOW 2012, Belgrade, Serbia, July 9- 11, 2012. Proceedings , volume 7363 of Lecture Notes in Computer Science , pages 288-301. Springer, 2012.

- [9] Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. "Analysis and evaluation of a raptor-based content delivery protocol and its high-performance extensions." Ad Hoc & Sensor Wireless Networks , 26(1-4):125-149, 2015.

- [13] Miguel Baguena, Chai-Keong Toh, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. "RCDP: Raptor-based Content Delivery Protocol for unicast communication in wireless networks for ITS." Journal of Communications and Networks , 15(2):198-206, 2013.

Regarding our content delivery protocol —Raptor-based Content Delivery Protocol (RCDP)— we presented in [2] our basic protocol and some optimizations. Later, in [3], we presented some filtering schemes to optimize its performance. Finally, in [4] and in its journal extension [9], we presented some additional

optimizations and results on performance. In [13], we presented an adaptation of our protocol for ITS.

Publications related to routing

- [6] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. "TGRP: topological-geographical adaptive routing protocol for vehicular environments." In 2014 IFIP Wireless Days, WD 2014, Rio de Janeiro, Brazil, November 12-14, 2014, pages 1-6. IEEE, 2014.

- [8] Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. "An adaptive anycasting solution for crowd sensing in vehicular environments." IEEE Trans. Industrial Electronics , 62(12):7911-7919, 2015.

In [6], we introduced the first version of our hybrid adaptive routing protocol, the Topological-Geographical Routing Protocol (TGRP), and we did an initial performance test. In a later article, [8], we presented the second version of our routing protocol, renamed to adaptive Anycasting solution for Vehicular Environments (AVE), and performed a more exhaustive performance evaluation.

Publications related to edge computing

- [11] Miguel Baguena, Andreas Pamboris, Peter R. Pietzuch, Mihail L. Sichitiu, and Pietro Manzoni. "Better performance in LTE networks with edge assistance: The World of Warcraft case." In Proceedings of the 12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, MobiQuitous 2015, Coimbra, Portugal, July 22-24, 2015 , pages 279-280. ICST / ACM, 2015.

- [12] Miguel Baguena, George Samaras, Andreas Pamboris, Mihail L. Sichitiu, Peter R. Pietzuch, and Pietro Manzoni. "Towards enabling hyper-responsive mobile apps through network edge assistance." In 13th IEEE Annual Consumer Communications & Networking Conference, CCNC 2016, Las Vegas, NV, USA, January 9-12, 2016, pages 399-404. IEEE, 2016.

- [15] Andreas Pamboris, Miguel Baguena, Alexander L. Wolf, Pietro Manzoni, and Peter R. Pietzuch. "Demo: NOMAD: an edge cloud platform for hyper-responsive mobile apps." In Proceedings of the 13th Annual

International Conference on Mobile Systems, Applications, and Services, MobiSys 2015, Florence, Italy, May 19-22, 2015 , page 459. ACM, 2015.

In poster [11] we presented our edge computing deployment. We evaluated the differences on server placement. In [12] we extend our previous publication. We evaluated the influence of handover and user interaction on new traffic patterns. Additionally, in [15], we collaborated with project NOMAD, an edge cloud platform for hyper-responsive mobile apps, in this demo. This work was performed under collaboration with the LSDS group at Imperial College London and North Carolina State University.

Publications related to simulation tools and models

- [5] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. "Towards realistic vehicular network simulation models." In Proceedings of the IFIP Wireless Days Conference 2012, Ireland, November 21-23, 2012 , pages 1-3. IEEE, 2012.

- [7] Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. "Assessing the impact of obstacle modeling accuracy on IEEE 802.11p based message dissemination." In Proceedings of the third ACM international symposium on Design and analysis of intelligent vehicular networks and applications, DIVANet@MSWiM 2013, Barcelona, Spain, November 3-4, 2013 , pages 123128. ACM, 2013.

Paper [5] is a short paper where we presented our channel propagation model and its impact on broadcasting. We extended this work in [7] with more extensive text, including new metrics to classify the cities.

- [14] Miguel Baguena, Sergio Martínez Tornell, Alvaro Torres, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. "VACaMobil: VANET Car Mobility manager for OMNeT++." In IEEE International Conference on Communications, ICC 2013, Budapest, Hungary, June 9-13, 2013, Workshops Proceedings, pages 1057-1061. IEEE, 2013.

- [10] Miguel Baguena, Sergio M. Tornell, Alvaro Torres, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. "A tool offering steady-state simulations for VANETs." Recent Advances in Communications and Networking Technology (Formerly Recent Patents on Telecommunication) , 2(2):102-112, 2013.

In [14], we presented our mobility manager, VACaMobil. It included VACaMobil's basic features and an initial performance evaluation. [10] is its extended version, where we presented additional features and evaluated them.

- [1] Miguel Báguena Albaladejo, Douglas J. Leith, and Pietro Manzoni. "Measurement-based modelling of LTE performance in Dublin city." 2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Valencia, 2016

In [1] we show the samples gathered in our LTE measurement campaigns and our LTE channel model. This work was performed under collaboration with Trinity College Dublin.

11.3 Future work

In coming years, vehicular networks will continue growing. Deployment of ITS systems is planned for many cities, and drivers will invest in vehicles with connectivity features. This will make node density increase, presenting new challenges.

We set a base to support these incoming changes with this thesis, while providing some tools to improve the network. Services like Internet browsing, VoIP and on-line gaming will become of common use in our roads. This bidirectional unicast communication must be carefully handled.

Using our research, all the traffic increment could be handled as easily as current traffic can be handled. This flexibility makes our approach a bet for the future, and provides a development path so that any person interested in deploying a vehicular network can follow it.

Regarding future work, some issues remain open. Focusing on transport protocols, the main issue that can be improved is CPU usage. Since network transmission is a very common task for applications, the CPU usage of solutions like our own (RCDP) must be optimized as much as possible. Additionally, it would be desirable to test the impact of different coding schemes in our protocol, seeking the one that better fits the goals of a vehicular network.

A second open issue would be keeping experimenting with adaptive routing. Some common routing algorithms were used, but the optimal set still remains unknown. A minimal set would simplify the overall design, making the solution easier to introduce in the day to day usage. Additionally, an implementation of this solution in a real environment would be greatly appreciated, since it is the last step to know the actual performance of the protocol and to make the fairest possible comparison.

The next logical step for edge computing would be developing scalable applications that could make use of this feature. A bidireccional interactive application, such as a videogame, would be the one getting the most of our platform, since it would need a fast communication between two (maybe close) nodes, and it would need some processing to be performed by the edge computer. Additionally, cloud gaming is also a kind of application that could take advantage of our edge computing system, and it will be easier to implement for current state-of-the-art games. Additionally, many companies would be interested in offering such a service for their games and devices.

The model and tools that we got from our research on LTE can be a base for many other research works. For example, the thin devices that we deployed to make our measurements will evolve to thin monitoring systems that could help to automatically perform network performance calibration or provider selection.

Bibliography

[1] Miguel Báguena Albaladejo, Douglas J. Leith, and Pietro Manzoni. Measurement-based modelling of LTE performance in dublin city. In *27th IEEE Annual International Symposium on Personal, Indoor, and Mobile Radio Communications, PIMRC 2016, Valencia, Spain, September 4-8, 2016*, pages 1–6. IEEE, 2016.

[2] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. Design, implementation, and optimization of a raptor-based content delivery protocol. In *Proceedings of the IFIP Wireless Days Conference 2011, Niagara Falls, ON, Canada, October 10-12, 2011*, pages 1–5. IEEE, 2011.

[3] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. Raptor-based reliable unicast content delivery in wireless network environments. In Chun Tung Chou, Tom Pfeifer, and Anura P. Jayasumana, editors, *IEEE 36th Conference on Local Computer Networks, LCN 2011, Bonn, Germany, October 4-7, 2011*, pages 841–849. IEEE Computer Society, 2011.

[4] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. RCDP: A novel content delivery solution for wireless networks based on raptor codes. In Xiang-Yang Li, Symeon Papavassiliou, and Stefan Rührup, editors, *Ad-hoc, Mobile, and Wireless Networks - 11th International Conference, ADHOC-NOW 2012, Belgrade, Serbia, July 9-11, 2012. Proceedings*, volume 7363 of *Lecture Notes in Computer Science*, pages 288–301. Springer, 2012.

[5] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. Towards realistic vehicular network simulation mod-

els. In *Proceedings of the IFIP Wireless Days Conference 2012, Ireland, November 21-23, 2012*, pages 1–3. IEEE, 2012.

[6] Miguel Baguena, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. TGRP: topological-geographical adaptive routing protocol for vehicular environments. In Luís Henrique Maciel Kosmalski Costa, Miguel Elias Mitre Campista, Artur Ziviani, Cigdem Sengul, José Marcos S. Nogueira, Josep Domènech, Marcelo G. Rubinstein, Pedro Braconnot Velloso, and Igor Monteiro Moraes, editors, *2014 IFIP Wireless Days, WD 2014, Rio de Janeiro, Brazil, November 12-14, 2014*, pages 1–6. IEEE, 2014.

[7] Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. Assessing the impact of obstacle modeling accuracy on IEEE 802.11p based message dissemination. In Mirela Sechi Moretti Annoni Notare and Abdelhamid Mammeri, editors, *Proceedings of the third ACM international symposium on Design and analysis of intelligent vehicular networks and applications, DIVANet@MSWiM 2013, Barcelona, Spain, November 3-4, 2013*, pages 123–128. ACM, 2013.

[8] Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. An adaptive anycasting solution for crowd sensing in vehicular environments. *IEEE Trans. Industrial Electronics*, 62(12):7911–7919, 2015.

[9] Miguel Baguena, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. Analysis and evaluation of a raptor-based content delivery protocol and its high-performance extensions. *Ad Hoc & Sensor Wireless Networks*, 26(1-4):125–149, 2015.

[10] Miguel Baguena, Sergio M. Tornell, Alvaro Torres, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni. A tool offering steady-state simulations for vanets. *Recent Advances in Communications and Networking Technology (Formerly Recent Patents on Telecommunication)*, 2(2):102–112, 2013.

[11] Miguel Baguena, Andreas Pamboris, Peter R. Pietzuch, Mihail L. Sichitiu, and Pietro Manzoni. Better performance in LTE networks with edge assistance: The world of warcraft case. In Pei Zhang, Jorge Sá Silva, Nic Lane, Fernando Boavida, and André Rodrigues, editors, *Proceedings of the 12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, MobiQuitous 2015, Coimbra, Portugal, July 22-24, 2015*, pages 279–280. ICST / ACM, 2015.

[12] Miguel Baguena, George Samaras, Andreas Pamboris, Mihail L. Sichitiu, Peter R. Pietzuch, and Pietro Manzoni. Towards enabling hyper-responsive mobile apps through network edge assistance. In *13th IEEE Annual Consumer Communications & Networking Conference, CCNC 2016, Las Vegas, NV, USA, January 9-12, 2016*, pages 399–404. IEEE, 2016.

[13] Miguel Baguena, Chai-Keong Toh, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. RCDP: raptor-based content delivery protocol for unicast communication in wireless networks for ITS. *Journal of Communications and Networks*, 15(2):198–206, 2013.

[14] Miguel Baguena, Sergio Martínez Tornell, Alvaro Torres, Carlos Miguel Tavares Calafate, Juan-Carlos Cano, and Pietro Manzoni. Vacamobil: VANET car mobility manager for omnet++. In *IEEE International Conference on Communications, ICC 2013, Budapest, Hungary, June 9-13, 2013, Workshops Proceedings*, pages 1057–1061. IEEE, 2013.

[15] Andreas Pamboris, Miguel Baguena, Alexander L. Wolf, Pietro Manzoni, and Peter R. Pietzuch. Demo: : NOMAD: an edge cloud platform for hyper-responsive mobile apps. In Gaetano Borriello, Giovanni Pau, Marco Gruteser, and Jason I. Hong, editors, *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys 2015, Florence, Italy, May 19-22, 2015*, page 459. ACM, 2015.