

**SOLUÇÕES DE DISSEMINAÇÃO DE DADOS
PARA REDES AD HOC SEM FIO**

JOÃO GUILHERME MAIA DE MENEZES

**SOLUÇÕES DE DISSEMINAÇÃO DE DADOS
PARA REDES AD HOC SEM FIO**

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação.

ORIENTADOR: ANTONIO ALFREDO FERREIRA LOUREIRO

COORIENTADOR: ANDRÉ LUIZ LINS DE AQUINO

COORIENTADORA: ALINE CARNEIRO VIANA

Belo Horizonte

Dezembro de 2013

JOÃO GUILHERME MAIA DE MENEZES

**DATA DISSEMINATION SOLUTIONS FOR
WIRELESS AD HOC NETWORKS**

Thesis presented to the Graduate Program
in Computer Science of the Federal Univer-
sity of Minas Gerais in partial fulfillment of
the requirements for the degree of Doctor
in Computer Science.

ADVISOR: ANTONIO ALFREDO FERREIRA LOUREIRO
CO-ADVISOR: ANDRÉ LUIZ LINS DE AQUINO
CO-ADVISOR: ALINE CARNEIRO VIANA

Belo Horizonte

December 2013

© 2013, João Guilherme Maia de Menezes.
Todos os direitos reservados.

Menezes, João Guilherme Maia de

M543d Data Dissemination Solutions for Wireless Ad hoc
Networks / João Guilherme Maia de Menezes — Belo
Horizonte, 2013
xxvii, 105 f. : il. ; 29cm

Tese (doutorado) — Universidade Federal de Minas
Gerais - Departamento de Ciência da Computação

Orientador: Antonio Alfredo Ferreira Loureiro
Coorientador: André Luiz Lins de Aquino
Coorientadora: Aline Carneiro Viana

1. Computação - Teses. 2. Redes de computadores -
Teses. 3. Redes de sensores sem fio - Teses.
I. Orientador. II. Coorientador. III. Coorientadora.
IV. Título. V. Soluções de disseminação de dados para
redes ad hoc sem fio.

519.6*22(043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Disseminação de dados para redes ad hoc sem fio


JOÃO GUILHERME MAIA DE MENEZES


Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROF. ANTONIO ALFREDO FERREIRA LOUREIRO - Orientador
Departamento de Ciência da Computação - UFMG


PROF. ANDRÉ LUIZ LINS DE AQUINO - Co-orientador
Instituto de Computação - UFAL


PROF. ALINE CARNEIRO VIANA - Co-orientadora
INRIA


PROF. HORÁCIO ANTONIO BRAGA F. DE OLIVEIRA
Departamento de Ciência da Computação - UFAM


PROF. LUIZ FILIPE MENEZES VIEIRA
Departamento de Ciência da Computação - UFMG


PROFA. RAQUEL APARECIDA DE FREITAS MINI
ICEI - PUC-MG


PROFA. ROSSANA MARIA DE CASTRO ANDRADE
Departamento de Computação - UFC

Belo Horizonte, 09 de dezembro de 2013.

To my parents, João and Cida, to my sister, Mirella, and to my wife, Luana, the most important people in my life.

Acknowledgments

Above all I thank my parents for their love and dedication in giving me the best possible education. To my sister, the most amazing person that I know. To my wife, my dear love.

I am grateful to my advisor, Loureiro, for his dedication, patience and for teaching me how to become a researcher.

I am also thankful to my co-advisers, André and Aline, and the committee members, Rossana Andrade, Horácio Oliveira, Raquel Mini and Luiz Viera, who gave invaluable insights on how to improve this thesis.

Finally, I am very thankful to all the friends that I have made during my Ph.D.

“I don’t want to learn what I’ll need to forget”
(Audioslave)

Resumo

As redes ad hoc sem fio têm recebido muita atenção nos últimos anos em função de sua capacidade em possibilitar a formação de redes espontâneas. Nessas redes, os nós devem cooperar de maneira distribuída para naturalmente criar ambientes de comunicação independentes de infraestruturas de gerenciamento centrais. Como exemplos de redes ad hoc sem fio temos as redes móveis ad hoc (MANETs), redes de sensores sem fio (RSSFs) e redes veiculares ad hoc (VANETs), as quais possuem características distintas.

A disseminação de dados é uma técnica muito empregada na realização de diferentes tarefas em redes ad hoc sem fio. Por exemplo, tal procedimento tem sido utilizado como um mecanismo de controle no estabelecimento de rotas em protocolos *unicast* e *multicast*, como um mecanismo para a criação de protocolos de replicação e armazenamento de dados, ou simplesmente como um procedimento de comunicação de dados. Os principais objetivos de qualquer solução de disseminação de dados são minimizar o número de pacotes retransmitidos e, ao mesmo tempo, garantir a entrega dos pacotes para o maior número de destinatários.

Dada a importância da técnica de disseminação de dados em redes ad hoc sem fio, esta tese investiga, inicialmente, como a disseminação de dados pode ser utilizada na concepção de um mecanismo de replicação e armazenamento de dados para RSSFs. Nesse contexto, propõe-se uma solução de disseminação de dados que depende de um pequeno número de nós sensores poderosos para criar estruturas de replicação e disseminar os dados sensorizados para os nós sensores. Nessa solução, os dados sensorizados são replicados e disseminados para os nós de tal forma que um nó sorvedouro móvel é capaz de visitar um pequeno número de sensores e coletar todos os dados produzidos pela rede. Resultados de simulação mostram que tal solução possui a menor sobrecarga de transmissão de mensagens quando comparada com soluções existentes, no entanto, ao custo de uma pequena piora na eficiência da disseminação e coleta de dados. De todo modo, mostra-se que ao explorar a redundância e correlação de dados inerente às RSSFs, é possível diminuir ainda mais a sobrecarga imposta pelo protocolo e, ao

mesmo tempo, garantir uma eficiência na disseminação e coleta de dados comparável às soluções existentes.

Em seguida, investiga-se como a disseminação de dados pode ser utilizada como um procedimento de comunicação de dados para reportar eventos para veículos que estão contidos em uma área de interesse em VANETs. Logo, propõe-se um protocolo de disseminação de dados para VANETs em rodovias, o qual é capaz de adaptar-se de forma transparente à condição de tráfego na rede com o objetivo de garantir a entrega dos dados para o maior número possível de destinatários. Resultados de simulação mostram que a solução proposta possui a melhor taxa de entrega de dados em cenários com tráfego esparsos, possui o menor atraso na entrega de mensagens, além de ser a solução com a menor sobrecarga de mensagens transmitidas em cenários com tráfego denso. Por fim, mostra-se que o protocolo proposto é robusto a erros de GPS.

Uma limitação da solução anterior é sua restrição a cenários de rodovias. Logo, propõe-se um novo protocolo de disseminação de dados que também é capaz de adaptar-se de forma transparente à condição de tráfego na rede em cenários urbanos. Além disso, o protocolo evita os efeitos de sincronização introduzidos pelo novo padrão de comunicação para ambientes veiculares. De forma a garantir um uso justo da largura de banda disponível e evitar a sobrecarga do canal de comunicação, tal solução é capaz de adaptar a taxa com que os veículos inserem os dados na rede. Resultados de simulação mostram que quando comparado a soluções existentes, o protocolo proposto possui a melhor taxa de entrega de mensagens, o menor atraso e a menor sobrecarga de mensagens transmitidas.

Abstract

Wireless ad hoc networks have gained a lot of momentum in the last few years due to their ability to enable spontaneous networking. In those networks, nodes cooperate in a distributed fashion way to spontaneously establish a communication environment independently of a centralized management infrastructure. Examples of wireless ad hoc network are mobile ad hoc networks (MANETs), wireless sensor networks (WSNs) and vehicular ad hoc networks (VANETs), which have distinct characteristics.

Network wide broadcasting or data dissemination is a very common procedure employed in different tasks in wireless ad hoc networks. For instance, such technique has been used as a control mechanism in route establishment of unicast and multicast protocols, as a method to create data replication and storage protocols, or simply as a data communication procedure. The main goals of any data dissemination solution are to minimize the number of packet retransmissions and to deliver as many packets as possible to the intended recipients.

Given the importance of the data dissemination procedure for wireless ad hoc networks, this thesis investigates how data dissemination may be used to build a data replication and storage mechanism for WSNs. Therefore, we propose a data dissemination solution that relies on a small subset of powerful nodes to create replication structures and disseminate the sensed data to the nodes in the network. In this scheme, the sensed data is intelligently replicated and disseminated to sensor nodes in such a way that a mobile sink can later visit a small subset of nodes to collect the sensed data produced by the whole network. Simulation results show that such solution has the lowest overhead when compared to existing approaches, however it possesses a slightly worse dissemination and collection efficiency. Nevertheless, we also show that by taking advantage of the data redundancy and correlation inherent to WSNs, it is possible to decrease the overhead of the proposed protocol and attain a dissemination and collection efficiency similar to existing approaches.

Thereafter, we investigate how data dissemination may be used as a data communication procedure to report events to drivers who are inside a region of interest in

VANETs. Hence, we propose a data dissemination protocol for highway VANETs that can seamlessly adapt to the perceived road traffic conditions to deliver messages to intended recipients. Simulation results show that when compared to existing solutions, our approach has the best delivery ratio under sparse traffic, and has both the lowest delay and the lowest overhead under dense traffic. Moreover, we also show that our solution is robust to GPS errors.

A limitation of the previous solution is that it is confined to highway scenarios. Therefore, we propose a new data dissemination protocol that can also seamlessly adapt to the perceived road traffic condition in urban environments. Furthermore, such solution avoids the synchronization effects introduced by the new data communication standard for vehicular networks. In order to enable fair use of the available bandwidth and avoid channel overloading, the protocol adapts the rate at which vehicles insert data into the channel. Simulation results show that when compared to existing solutions, our approach provides the best delivery, the lowest delay and the lowest overhead.

List of Figures

| | | |
|------|--|----|
| 1.1 | Examples of wireless ad hoc networks | 1 |
| 2.1 | Traditional data collection strategy may lead to the energy hole problem, i.e., nodes closer to the sink deplete their batteries early on, thus hampering the data collection process | 9 |
| 2.2 | Data collection using a mobile sink. Depending on how the data is disseminated within the network, the mobile sink does not need to visit all sensor nodes in order to collect all the sensed data | 10 |
| 2.3 | ProFlex enables different dissemination strategies depending on how the importance factor is assigned to each node in the network. However, in this thesis we will focus on a uniform dissemination strategy, therefore all nodes possess the same importance factor | 14 |
| 2.4 | The main steps of ProFlex | 16 |
| 2.5 | Network with 1000 sensor nodes of which 3 are <i>H-sensor</i> nodes | 18 |
| 2.6 | Forward data example | 22 |
| 2.7 | Trade-off between data gathering efficiency and messages overhead | 25 |
| 2.8 | Data gathering efficiency for a scenario with no message loss nor node failure | 28 |
| 2.9 | Data dissemination efficiency for a scenario with no message loss nor node failure | 28 |
| 2.10 | Data gathering efficiency for a scenario with message loss | 29 |
| 2.11 | Data gathering efficiency for a scenario with node failure | 30 |
| 2.12 | Average number of messages sent by a node as a function of its depth in the tree | 31 |
| 2.13 | Example of a network with correlated data when $d = 10$ | 33 |
| 2.14 | Data gathering efficiency and data dissemination efficiency for some versions of ProFlex under a reliable scenario | 34 |
| 2.15 | Data gathering efficiency for some versions of ProFlex under a scenario with message loss | 35 |

| | | |
|------|--|----|
| 2.16 | Data gathering efficiency for some versions of ProFlex under a scenario with node failure | 36 |
| 2.17 | Data gathering efficiency for a scenario with no message loss nor node failure and a network with $n_H = 5$ and $r_2 = 480$ m | 37 |
| 3.1 | After a collision, a source vehicle produces and disseminates a warning to vehicles approaching the accident (vehicles moving to the east [→] direction). The warning must be disseminated to all vehicles inside the region of interest (ROI) defined by the collision-avoidance application | 40 |
| 3.2 | In this example, the source vehicle produces a warning message to be disseminated to following vehicles. Hence, the MD = westerly and the intended recipients are vehicles A and B. Notice that, despite vehicle D is moving in the same direction as the source vehicle, it is outside the region of interest (ROI) defined by the application. Here, vehicle A has <IR = true, MDC = true, ODC = true>, vehicle B has <IR = true, MDC = false, ODC = true>, vehicle C has <IR = false, MDC = false, ODC = true> and vehicle D has <IR = false, MDC = false, ODC = false> | 47 |
| 3.3 | Broadcast suppression mechanism. Using the sender-based part of the algorithm, the transmitter chooses a priori the next forwarder, i.e., vehicle C. If this procedure fails, the receiver-based part guarantees that some other vehicle will rebroadcast, e.g., vehicles B, D and E | 49 |
| 3.4 | Store-carry-forward. HyDi uses the last vehicles in a group of connected vehicles to perform the store-carry-forward task, e.g., vehicles B and D. Moreover, it employs extra vehicles for this task to act as backup in case the first ones fail, e.g., vehicles A and C | 53 |
| 3.5 | In HyDi, multiple vehicles store-carry-forward a message, for instance, vehicles C and D. However, when they meet a new neighbor able to resume the dissemination process, e.g., vehicle E, only the vehicle with MDC = false will broadcast the message, in this example, vehicle D | 54 |
| 3.6 | The base highway scenario considered in our performance analysis | 57 |
| 3.7 | The delivery ratio for all road traffic scenarios. Notice that, HyDi has the best delivery ratio, especially under low traffic scenarios, where the store-carry-forward communication model prevails | 58 |

| | | |
|------|--|----|
| 3.8 | The total number of data messages transmitted by all vehicles during the dissemination process. HyDi transmits more messages under low traffic scenarios because it delivers more messages than the other protocols. Moreover, it employs many vehicles to the store-carry-forward task, which increases the reliability of the protocol at the cost of an increase in the overhead. As the traffic increases, HyDi transmits less messages than the other protocols | 59 |
| 3.9 | The average delay to deliver data messages to intended recipients. HyDi has a higher delay under low traffic scenarios because it delivers more messages to intended recipients. As the traffic increases, HyDi's delay is only higher than Flooding's | 60 |
| 3.10 | The delivery ratio for a highway with an exit. Notice that, the delivery ratio for HyDi is not affected by vehicles leaving the network | 61 |
| 3.11 | The total number of data messages transmitted for a highway with an exit. Under low traffic, HyDi is the protocol with the highest overhead. However, as the network becomes connected, HyDi transmits less messages than related protocols | 62 |
| 3.12 | The average delay to deliver data messages to intended recipients in a highway with an exit. HyDi has a higher delay under low traffic scenarios because it delivers more messages to intended recipients. As the traffic increases, HyDi's delay is only higher than Flooding's. In general, HyDi's delay is not much affected by the fact that vehicles are leaving the network during the dissemination process | 62 |
| 3.13 | The delivery ratio for HyDi in the presence of GPS errors. Essentially, HyDi's delivery is not affected by wrong reported positions | 63 |
| 3.14 | The total number of data messages transmitted for HyDi in the presence of GPS errors. Its is possible to notice a increase at traffics of 500 and 600 vehicles/hour | 63 |
| 3.15 | The average delay for all traffic scenarios | 64 |
| 4.1 | After a collision, a source vehicle produces and disseminates a warning to all vehicles inside the region of interest (ROI) defined by the application | 68 |
| 4.2 | In UV-CAST, vehicles selected as border vehicles should go to the store-carry-forward state. In this figure, vehicles <i>A</i> , <i>D</i> , <i>E</i> , <i>F</i> and <i>G</i> are considered as border vehicles with respect to the <i>Src</i> vehicle. Therefore, they store-carry-forward messages received from the <i>Src</i> . Image Source: [Viriyasitavat et al., 2010] | 72 |
| 4.3 | The main components of the ADVENT protocol | 73 |

| | | |
|------|---|----|
| 4.4 | The general idea of the forwarding zone | 74 |
| 4.5 | The channels in the 5.85 GHz frequency range allocated for vehicular communication | 79 |
| 4.6 | The channel hopping mechanism used in vehicular communication. When a message that must be sent on a SCH arrives at the MAC layer, but the CCH is currently active, then such message must wait until the SCH becomes active in order to be transmitted. For instance, the messages with IDs 1 and 2 are assigned for transmission on a SCH, but when they arrive at the MAC layer from an upper layer, the CCH is currently active. Then, the MAC layer buffers them until the SCH becomes active, when it finally proceeds to transmit them. Notice that, the same may happen with a message assigned for the CCH when the SCH is currently active | 80 |
| 4.7 | The synchronization effect introduced by the channel hopping mechanism used in the IEEE 802.11p MAC layer. Here, vehicles <i>A</i> and <i>B</i> schedules with different waiting delays a message to be sent down to the MAC layer. Indeed, the messages are sent down to the MAC layer of each vehicle at different moments in time. However, since the CCH was active when the MAC layer receives the messages and they are assigned for transmission on the SCH, the MAC layer buffers them and waits for the SCH to become active. When the SCH finally becomes active, both vehicles transmit the messages at the same time, thus leading to a collision | 81 |
| 4.8 | Example showing how the desynchronization mechanism works | 83 |
| 4.9 | Delivery ratio for the Manhattan grid scenario | 87 |
| 4.10 | Total number of data messages transmitted for the Manhattan grid scenario | 88 |
| 4.11 | Average delay to disseminate the data messages to all intended recipients in the Manhattan grid scenario | 88 |
| 4.12 | Delivery ratio for the real city scenario | 89 |
| 4.13 | Total number of data messages transmitted for the real city scenario . . . | 90 |
| 4.14 | Average delay to disseminate the data messages to all intended recipients in the real city scenario | 90 |

List of Tables

| | | |
|-----|--|----|
| 2.1 | Main concepts of ProFlex | 17 |
| 2.2 | Simulation parameters | 24 |
| 2.3 | Protocols overhead | 32 |
| 2.4 | Overhead of some versions of ProFlex | 33 |
| 2.5 | Overhead in a network with $n_H = 5$ and $r_2 = 480$ m | 36 |
| 2.6 | Comparison of protocols | 38 |
| 3.1 | Simulation parameters. | 56 |
| 4.1 | Simulation parameters used in our assessment | 85 |
| 4.2 | Comparison of protocols studied in chapters 3 and 4 | 91 |

Contents

| | |
|---|--------------|
| Acknowledgments | xi |
| Resumo | xv |
| Abstract | xvii |
| List of Figures | xix |
| List of Tables | xxiii |
| 1 Introduction | 1 |
| 1.1 Motivation | 1 |
| 1.2 Goals | 3 |
| 1.3 Contributions | 4 |
| 1.4 Outline | 5 |
| 2 Data Dissemination in Heterogeneous Wireless Sensor Networks with Mobile Sinks | 7 |
| 2.1 Introduction | 8 |
| 2.2 Related Work | 10 |
| 2.3 System Model | 13 |
| 2.4 Proposed Protocol | 15 |
| 2.4.1 Tree Construction | 16 |
| 2.4.2 Importance Factor Distribution | 18 |
| 2.4.3 Data Distribution | 21 |
| 2.5 Performance Analysis | 23 |
| 2.5.1 ProFlex Assessment | 24 |
| 2.5.2 ProFlex vs. Literature Protocols | 27 |
| 2.5.3 Improving ProFlex | 32 |

| | | |
|----------|--|-----------|
| 2.6 | Chapter Remarks | 37 |
| 3 | Data Dissemination in Highway Vehicular Ad hoc Networks with Extreme Traffic Conditions | 39 |
| 3.1 | Introduction | 40 |
| 3.2 | Related Work | 42 |
| 3.3 | Proposed Protocol | 45 |
| 3.3.1 | Broadcast Suppression | 47 |
| 3.3.2 | Store-carry-forward | 51 |
| 3.4 | Performance Analysis | 55 |
| 3.4.1 | Evaluated Scenario | 55 |
| 3.4.2 | Evaluated Metrics | 57 |
| 3.4.3 | Highway Results | 58 |
| 3.4.4 | Highway with an Exit Results | 60 |
| 3.4.5 | GPS Drift Results | 62 |
| 3.5 | Chapter Remarks | 64 |
| 4 | Data Dissemination in Urban Vehicular Ad hoc Networks with Extreme Traffic Conditions | 67 |
| 4.1 | Introduction | 68 |
| 4.2 | Related Work | 70 |
| 4.3 | Proposed Protocol | 72 |
| 4.3.1 | Broadcast Suppression | 73 |
| 4.3.2 | Store-carry-forward | 77 |
| 4.3.3 | Delay Desynchronization | 79 |
| 4.3.4 | Rate Control | 82 |
| 4.4 | Performance Analysis | 83 |
| 4.4.1 | Simulation Setup | 84 |
| 4.4.2 | Evaluated Metrics | 85 |
| 4.4.3 | Manhattan Grid Results | 86 |
| 4.4.4 | Real City Results | 89 |
| 4.5 | Chapter Remarks | 91 |
| 5 | Final Remarks | 93 |
| 5.1 | Conclusions | 93 |
| 5.2 | Publications | 95 |
| 5.2.1 | Periodicals | 95 |
| 5.2.2 | Conferences | 95 |

Chapter 1

Introduction

1.1 Motivation

Wireless ad hoc networks have attracted the interest of the research and industrial communities in the last few years due to the intrinsic capability of these networks in enabling spontaneous networking [Feeney et al., 2001; Boukerche, 2005; Cesana et al., 2010; Marfia et al., 2013; Ruj et al., 2013]. Nodes can gather together and cooperate to spontaneously create a communication environment, independently of a pre-established or centralized management infrastructure (see Figure 1.1). Therefore, nodes in these networks do not behave only as traditional hosts, i.e., producing and consuming data, but they must also act as routers, collaboratively relaying data from other nodes. This general framework is attractive in a wide range of scenarios, especially those involving environment and industrial monitoring, and disaster recovery, since relying on a fixed infrastructure is almost impossible [Zhang and Lee, 2000; Zussman and Segall, 2003; Mainwaring et al., 2002; Toor et al., 2008].

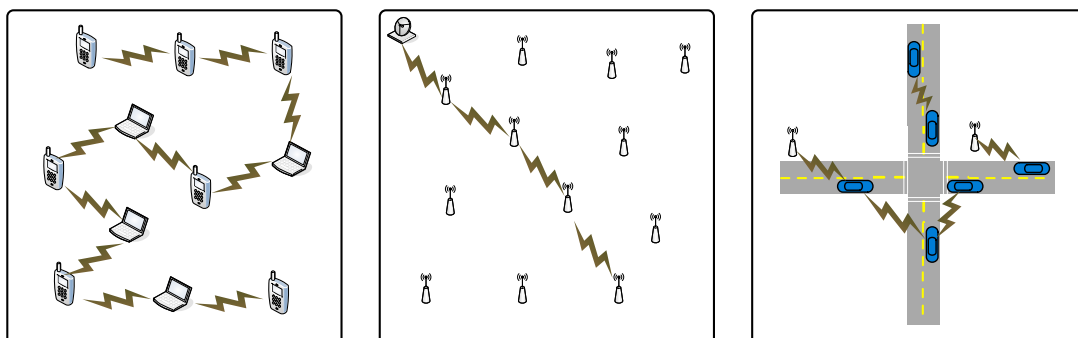


Figure 1.1: Examples of wireless ad hoc networks

Despite the benefits inherited from the absence of a pre-established and centralized infrastructure, this fact also imposes several challenges. For instance, nodes in these networks must have self-configuring and self-management capabilities to enable the deployment and operability of these networks without human intervention. Furthermore, to guarantee efficient and robust communication to all entities, nodes must cooperate in a distributed manner to route packets through the network and deliver them to the intended recipients. All this must be performed by using the available bandwidth effectively, otherwise, service disruption may happen, resulting in a non-functional network.

Depending on the network characteristics, such as the type of nodes (sensors, laptops, cell phones or cars) and node mobility (static, partially mobile or completely mobile), wireless ad hoc networks can be further classified into mobile ad hoc networks (MANETs) [Kieess and Mauve, 2007], wireless sensor networks (WSNs) [Boukerche, 2008; Yick et al., 2008], vehicular ad hoc networks (VANETs) [Hartenstein and Laberteaux, 2008], etc, each with its own active research community, research challenges and set of solutions and protocols.

The activity known as network wide broadcasting or data dissemination, in which a node transmits a packet to a subset or all nodes in the network, is very common in accomplishing many different tasks in all networks described above. For instance, many unicast and multicast routing protocols rely on data dissemination to establish routes between source and receivers in MANETs [Johnson et al., 2001; Perkins and Royer, 1999; Zhang and Jacob, 2003; Ko and Vaidya, 1998; Boukerche, 2004]. Moreover, data dissemination has been used for replicating data, reconfiguring, querying and reprogramming WSNs [Bar-Yossef et al., 2008; Viana et al., 2009; Vecchio et al., 2010; Whitehouse et al., 2006; Lin and Levis, 2008; Paek et al., 2010]. Finally, data dissemination has also been employed as a data communication procedure to notify drivers about events of interest in VANETs [Tonguz et al., 2010; Viriyasitavat et al., 2010; Schwartz et al., 2011; Ros et al., 2012].

The main challenge of any data dissemination solution is to *minimize the number of packet retransmissions while ensuring that the packet is delivered to the intended recipients*. Furthermore, depending on the purpose of the data dissemination and the type of wireless ad hoc network, such task (i) should be simple and inexpensive in terms of computing resources due to possible hardware constraints; (ii) should handle node failures, since the underlying hardware may be unreliable; (iii) should incur a low delay to deliver packets, since the application at hand may have time-strict requirements; (iv) should handle dynamic topologies, since nodes can move at very high speeds; and finally, (v) should treat network disconnections, since the density in the

network can vary in time and space. Therefore, the design of data dissemination solutions for wireless ad hoc networks can be very challenging. The simplest solution for data dissemination is the pure flooding algorithm, in which all nodes upon receiving a packet for the first time immediately retransmit it. Despite its simplicity, this solution produces a great number of redundant packets, which may cause many collisions and waste precious bandwidth, a problem known as broadcast storm problem [Williams and Camp, 2002; Wisitpongphan et al., 2007].

During a data dissemination process, once a node receives a packet it must decide (i) whether to retransmit it, (ii) when to retransmit it, and (iii) whether to store it locally, for instance, to be later retransmitted to another node or to be collected by a special node. These decisions can be hard-coded, such as on a simple flooding, or it can be made based on a counting procedure [Ni et al., 1999; Bar-Yossef et al., 2008], the distance to the transmitter [Tonguz et al., 2010; Schwartz et al., 2011], the location of the node that has received the packet [Ni et al., 1999; Viriyasitavat et al., 2010], an inferred probability [Viana et al., 2009; Vecchio et al., 2010; Bakhouya et al., 2011], or the network topology [Lim and Kim, 2000; Peng and Lu, 2000; Ros et al., 2012]. In this thesis, we propose data dissemination solutions where nodes make decisions based on probability (see Chapter 2), distance, location and network topology (see Chapters 3 and 4).

1.2 Goals

The general goal of this thesis is to propose new data dissemination solutions for wireless ad hoc networks that can accomplish different tasks in each one of them. Moreover, depending on the purpose of the data dissemination and the type of wireless ad hoc network, different research challenges must be tackled. Therefore, our first specific goal is to investigate data dissemination as a means for data replication and storage mechanism in WSNs. Solutions designed for this task must take into account the fact that nodes in these networks are static, resource constrained and often unreliable. Thereafter, as our second specific goal, we investigate data dissemination as a data communication procedure to report events to drivers who are inside a region of interest in VANETs. The nodes (vehicles) in these networks do not possess the same resource constraints as the ones in WSNs. However, due to node mobility at high speeds, these networks are much more dynamic when compared to WSNs.

1.3 Contributions

The main contributions of this thesis are the design and development of three data dissemination solutions, one for WSNs and two for VANETs. More specifically, we specify three algorithms as follows:

- **ProFlex: data dissemination for heterogeneous WSNs with mobile sinks:** is a data dissemination solution that relies on a small subset of powerful nodes to create replication structures and disseminate the sensed data to the nodes in the network. In this scheme, the sensed data is intelligently replicated and disseminated to sensor nodes in such a way that a mobile sink can later visit a small subset of nodes in order to collect the sensed data produced by the whole network. This scheme can be used as an alternative approach for the traditional data collection task in WSNs, where nodes route packets to a sink positioned at the border of the network. Simulation results show that ProFlex has the lowest overhead when compared to existing approaches, however it possesses a slightly worse dissemination and collection efficiency. Nevertheless, we show that by taking advantage of the data redundancy and correlation inherent to WSNs, it is possible to decrease the overhead of ProFlex even more and at the same time accomplish a dissemination and collection efficiency similar to existing approaches.
- **HyDi: data dissemination in highways for VANETs with extreme traffic conditions:** assuming that the broadcast data dissemination communication model will prevail over the unicast communication model in VANETs, we propose HyDi. Such solution works solely under highway scenarios and disseminates data to all vehicles located in a region of interest defined by an application. The key point of the protocol is that, while most existing solutions focus solely on dense and connected scenarios, HyDi guarantees that the packets are delivered to all intended recipients independently of the road traffic condition perceived, i.e., the protocol adapts to whether the network is dense or sparse. Moreover, the protocol does not rely on any fixed infrastructure and nodes make all decisions based on the knowledge of their one-hop neighbors. Simulation results show that, when compared to related protocols, HyDi is the protocol with the best delivery ratio under sparse traffic, and it has the lowest delay and lowest overhead under dense traffic. Finally, we show that our solution is robust to GPS errors.
- **ADVENT: data dissemination in urban VANETs with extreme traffic conditions:** just like HyDi, ADVENT also is a data dissemination solution

that guarantees packet delivery to all intended recipients independently of the perceived road traffic condition. However, contrarily to HyDi, ADVENT operates under urban scenarios and nodes make all decisions based only on the information about the transmitter of a packet. Moreover, it avoids the synchronization effects introduced by the IEEE 802.11p standard [Eckhoff et al., 2012]. It also adapts the rate at which a vehicle inserts data into the channel according to the perceived available bandwidth. Therefore, besides the traditional emphasis on the decisions of whether to retransmit, when to retransmit and whether to store the packet locally, we also investigate how fast should the transmission happen in order to avoid channel overloading. Simulation results show that ADVENT has the best delivery, the lowest delay and lowest overhead when compared to literature protocols.

1.4 Outline

This thesis is organized as follows. Chapter 2 describes the data collection problem in WSNs and presents ProFlex, our data dissemination solution that uses mobile sinks as an alternative approach for data collection. Chapter 3 outlines the problem of data dissemination to a group of vehicles in a highway and shows HyDi, our data dissemination proposal for varying road traffic conditions. In the following, Chapter 4 addresses the same problem, however under urban scenarios. In this chapter we present ADVENT, which adapts not only to the road traffic condition but also to the network traffic on the channel. Finally, Chapter 5 shows our final remarks, some possible future work and the publications related to this thesis.

Chapter 2

Data Dissemination in Heterogeneous Wireless Sensor Networks with Mobile Sinks

A fundamental activity in WSNs is the collection of the sensed data. A common adopted approach is for sensor nodes to route the sensed data by means of multi-hop communication to a special node called sink. However, in this scheme, nodes closer to the sink must relay packets from all other nodes in the network, thus depleting their batteries early on and harming the overall data collection process, an issue known as the energy-hole problem. Despite some alternatives to overcome this problem, such as increasing the node density in the area closer to the sink, an interesting approach is for sensor nodes to replicate and disseminate the sensed data within the network and then a mobile sink visits a small subset of the nodes to collect the whole sensed data. With this in mind, this chapter presents ProFlex, a data dissemination protocol for large-scale heterogeneous wireless sensor networks with mobile sinks. ProFlex guarantees robustness in data collection by intelligently managing data replication and dissemination among selected storage nodes in the network. Contrarily to related protocols, ProFlex considers the resource constraints of sensor nodes and constructs multiple data replication structures, which are managed by more powerful nodes. Additionally, ProFlex takes advantage of the higher communication range of such powerful nodes and uses the long-range links to improve the data distribution to storage nodes. When compared with related protocols, we show through simulation that ProFlex has an acceptable performance under message loss scenarios, decreases the overhead of transmitted messages, and decreases the occurrence of the energy hole problem. Moreover, we propose an improvement that takes advantage of the inherent data correlation and

redundancy of wireless sensor networks in order to decrease even further the protocol's overhead without affecting the quality of the data distribution to storage nodes.

2.1 Introduction

The deployment of large-scale Wireless Sensor Network (WSN) applications (e.g., environment sensing, patient monitoring, military surveillance, etc.), which operate unattended for long periods of time and generate a considerable amount of data, poses several challenges [Akyildiz et al., 2002; Yick et al., 2008]. One of them is *how to retrieve the sensed data*. This is usually performed by a special node called *sink*. Contrary to ordinary sensor nodes, the sink node has enhanced computational, storage and power capabilities since it is responsible for storing and processing the network sensed data. Moreover, the network may employ a static or a mobile sink. In the former case, the ordinary sensors need to route the sensed data to the sink, so connectivity to at least one sink in the network must be maintained to guarantee a good data collection (see Figure 2.1a). However, such an approach suffers from the energy hole problem in which nodes closer to the sink typically consume more energy due to data relaying from other nodes in the network [Liu et al., 2010; Wu et al., 2008; Li and Mohapatra, 2007]. Therefore, disconnections may happen in the network, especially in the region closer to the sink, compromising the overall data collection process (see Figure 2.1b). To alleviate this problem, some studies propose using mobile sinks [Luo et al., 2005; Song and Hatzinakos, 2007; Pazzi and Boukerche, 2008]. In this approach, a mobile sink has the flexibility of traversing the network to gather the sensed data directly from the sensor nodes, and, thus, no single node should suffer from the overhead of relaying data from all other nodes in the network.

In this context and for scalability reasons, it may be infeasible for the mobile sink to visit all the network nodes in order to collect the sensed data. Therefore, a key research challenge is *how can the sensed data be disseminated among sensor nodes* (Figure 2.2a), so it can be later gathered by a mobile sink without the need of visiting all sensor nodes in the network (Figure 2.2b). Depending on how the data is disseminated, the mobile sink (i) may have to follow a predefined trajectory in which it needs to visit specific storing nodes or locations in the network [Basagni et al., 2008; Luo et al., 2006; Yang et al., 2006; Urgaonkar and Krishnamachari, 2004; Chatzigiannakis et al., 2008; Shenker et al., 2003; Hamida and Chelius, 2008], or (ii) it can be free to follow an uncontrolled mobility pattern [Viana et al., 2009; Bar-Yossef et al., 2008; Vecchio et al., 2010]. Clearly, avoiding restrictions on the mobile sink trajectory is

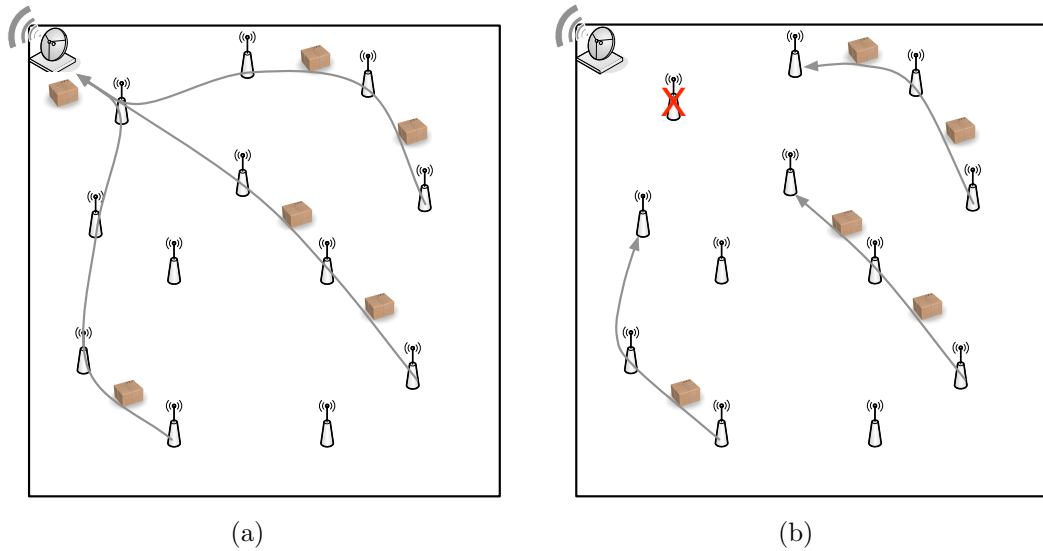


Figure 2.1: Traditional data collection strategy may lead to the energy hole problem, i.e., nodes closer to the sink deplete their batteries early on, thus hampering the data collection process

beneficial for both the mobile sink and sensor nodes, since the network is free to adapt to changing conditions. Hence, herein we keep our focus on how should the sensed data be disseminated to storing nodes in WSNs with one or more mobile sinks whose trajectories are unknown to the sensor nodes.

There are a few studies with the same aforementioned focus that use more powerful nodes in conjunction with ordinary sensor nodes to perform distributed data storage [Sheng et al., 2006, 2007; Anastasi et al., 2008]. In this scenario, only those powerful nodes are responsible for storing all sensed data, since the assumption is that they have no storage constraints. Moreover, this heterogeneous configuration does not present the same performance and scalability issues as homogeneous WSNs [Cavalcanti et al., 2004; Helmy, 2003; Sharma and Mazumdar, 2008]. Nevertheless, the use of more powerful nodes does not overcome the problem of data losses, since these nodes still can fail. To increase the network resilience to failures, a possible approach is to replicate the data and keep it at different storage nodes. Furthermore, it should be noticed that in the presence of a mobile sink, a good configuration on the number of data replicas and on the dissemination procedure to storing nodes might enable the sink to get a representative sample of the entire network data by visiting only a small percentage of the storing nodes [Bar-Yossef et al., 2008; Viana et al., 2010; Vecchio et al., 2010]. Hence, the use of *a suitable replication mechanism together with a well-thought decision of where to disseminate a specific data packet* are key elements for the effectiveness and

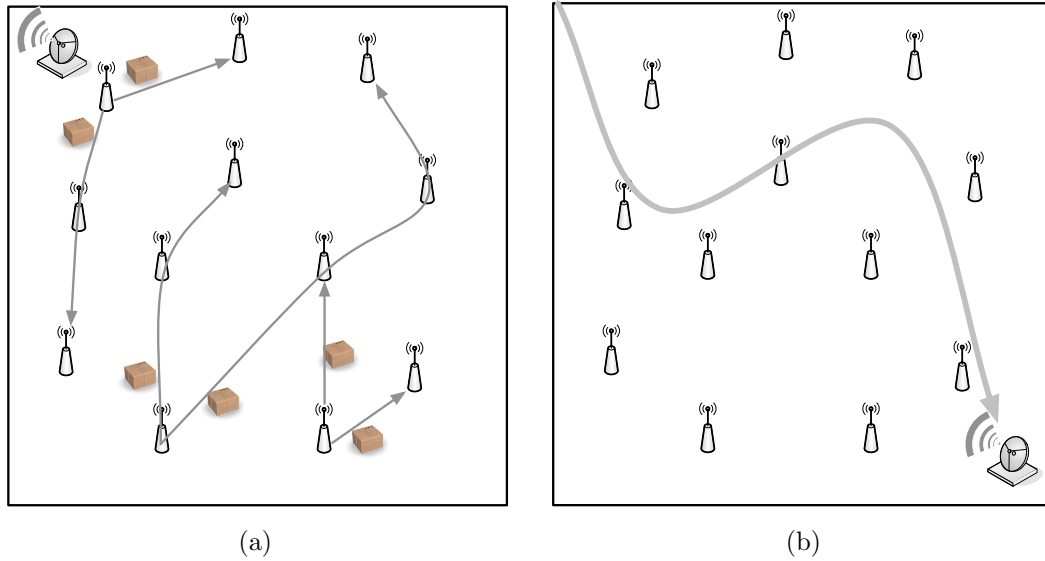


Figure 2.2: Data collection using a mobile sink. Depending on how the data is disseminated within the network, the mobile sink does not need to visit all sensor nodes in order to collect all the sensed data

efficiency of a data storage protocol.

Given these principles, in this work we propose a protocol named *ProFlex* that employs powerful nodes to perform distributed data storage in Heterogeneous Wireless Sensor Networks (HWSNs). However, instead of using the extra storage features of these nodes, we take advantage of their powerful communication capability and use the long-range links to improve data distribution. Thus, ProFlex is by design aware of the WSN heterogeneous topology. Simulation results show that by using a heterogeneous network topology, ProFlex has an acceptable performance under scenarios with message losses, decreases the overhead of transmitted messages and achieves good dissemination and collection (gathering) efficiency results. Moreover, ProFlex can exploit the data correlation and redundancy inherent to WSNs to decrease the overhead of transmitted messages even further without any negative effect on the data collection efficiency results.

2.2 Related Work

In the following, we present some of the proposals for data dissemination and distributed data storage in WSNs.

Sheng et al. [2006] study the data storage placement in WSNs to deal with the traditional problem of traffic overload and, consequently, high energy consumption of nodes closer to the sink. To overcome this problem, they propose two network models. The first one considers a tree topology rooted at the sink and a subset of nodes are selected as storage nodes, which are responsible for storing data collected by their descendants in the tree. In the second model, a tree topology is constructed after the planned deployment of the storage nodes, whose positions are obtained from a linear programming optimization. Nevertheless, node failures are not considered in both models, which might result in the loss of all data collected by storage nodes' descendants. Ratnasamy et al. [2002] propose GHT, a geographic hash table for data centric storage. In GHT, once a node generates data, it produces a packet and hashes it to a geographical position in the sensor field. Then, using the geographic routing protocol GPSR [Karp and Kung, 2000], the packet is routed to the closest node in the hashed geographical position. Such node will be responsible for storing the packet. Moreover, neighbors of this node also store the packet in order to handle node failures. However, such approach does not perform a uniform dissemination of the data produced by the network.

Bar-Yossef et al. [2008] propose a lightweight random membership service for ad hoc networks called RaWMS. The protocol provides each node with a partial uniformly chosen view of network nodes, i.e., each node in the network stores a subset of the data sensed by itself and by other nodes. The protocol is based on a reverse maximum degree random walk (RW) sampling technique. In RaWMS, every data producer node starts a reverse maximum degree RW, whose message carries the node's identifier and data. Each RW traverses the network for a predefined number of hops, so every message has an associated time-to-live (TTL) field that defines the length of the RW. The last node in the RW appears as if it was picked uniformly at random out of all network nodes and it will be responsible for storing the data carried by the RW. The authors prove that when the RW finishes, each node will have a uniform random view with data from the nodes in the network. In other words, RaWMS uniformly distributes the network data to the sensor nodes. Although the results of RaWMS to uniformly distribute the monitored data throughout the network are quite encouraging in terms of data gathering efficiency, as we shall present in our performance analysis (see Section 2.5), RaWMS incurs a high overhead, resulting in a short network lifetime.

Vecchio et al. [2010] propose Deep, a density-based proactive data dissemination protocol for WSNs with uncontrolled sink mobility. The goal is to obtain an effective uniform distribution of the sensed data at a reduced communication overhead. Deep combines a probabilistic flooding with a probabilistic storing scheme that allows the

sink to gather a representative view of the network’s sensed data by visiting any set of x out of n total nodes, where $x \ll n$. In Deep, when node i receives a message m for the first time, it rebroadcasts m with probability $p = \min(1, \frac{\beta}{|N(p)|})$, where $N(p)$ is the one hop neighborhood of i and β is the desired average number of retransmissions in the neighborhood. Moreover, if node i does not rebroadcast m and does not hear any other rebroadcast of m after a period of time, then i rebroadcasts m after all. Finally, for a partial view (local data sample) of size \sqrt{n} , where n is the number of nodes in the network, node i stores a new received message m with probability equal to $s = \frac{\sqrt{n}}{n}$. A node in Deep must keep track of all received messages during its operation, what might be impracticable in scenarios with a high number of produced messages. Moreover, Deep presents data gathering results comparable to RaWMS, although incurring a far lower message overhead. Nevertheless, such overhead is still high when compared with other approaches.

One such approach is the Supple protocol [Viana et al., 2010] — a flexible probabilistic data dissemination protocol for WSNs that considers static or mobile sinks. The Supple protocol has three phases: tree construction, weight distribution, and data replication. The first phase is a tree construction initiated by a central sensor node of the sensing area (e.g., the sink node). The central sensor node is responsible for receiving and replicating the collected data in the network. The second phase assigns weights to nodes, which represent the probability for a node storing data. Supple uses the hop distance of a node to the central node to calculate this probability. In the last phase, the sensor nodes send their data to the central node and this node replicates each data $r(v)$ times using the tree infrastructure and according to its storage probability. The value of $r(v)$ depends on the weights and on the amount of data each node is allowed to store. Viana et al. [2010] claim that a mobile sink visiting a small fraction of nodes, i.e., about $2.3\sqrt{n}$, for a total of n nodes, can retrieve all the generated data in the network. Moreover, due to the $r(v)$ replicas, there will be no data losses in case there is a failure of a small number of nodes. However, Supple does not consider the problems of finding a good positioned central node, energy consumption and traffic overload at nodes closer to the central node.

Notice that the main solutions described here rely on some kind of replication mechanism in order to increase the protocol’s resilience to node failures and message losses, and also as a support mechanism for the proper data dissemination among storing nodes. Moreover, the data dissemination from one node to another may rely on an always-up routing structure like the one employed by Supple or it may be based on other communication mechanisms like the gossip-based approach employed by Deep. Clearly, each existing proposal shares its strengths and weaknesses, thus our solution

proposes to benefit from the best techniques employed by those protocols combined with the main features inherent to HWSN designs. Furthermore, due to resource constraints of these networks we focus on (i) increasing the network resilience to failures with the use of multiple replication structures and (ii) reducing the overhead as much as possible at the cost of a small penalty in the data dissemination and collection efficiency.

2.3 System Model

The main assumptions adopted by our proposed solution are detailed hereafter.

Nodes: We consider that there is a large number n of sensor nodes scattered on a given geographic area for collecting data or monitoring events. All sensors are uniquely identified and can be of two types. The first one, named *L-sensor* for low-end sensor, is a node with limited resources, including processor, storage, communication and power resources. The second type, named *H-sensor* for high-end sensor, is a node with more sophisticated resources. Thus, *H-sensor* nodes have improved processing, storage, battery and communication power when compared with *L-sensor* nodes. A question that might arise is: *Why not designing a sensor network comprised of just H-sensor nodes?* Whereas *H-sensor* nodes are more powerful when compared with *L-sensor* nodes, the latter are much less expensive. Hence, it is assumed the network is composed of n_L *L-sensor* nodes, and n_H *H-sensor* nodes, where $n = n_L + n_H$ and $n_L \gg n_H$. Moreover, nodes later selected as storage nodes are provided with a *partial view* (local data sample) v with data from some other nodes and the data sensed by themselves. This *set of storing nodes* is defined as S . Therefore, each node may act as a storage node for some other nodes, but not for all of them. Due to the limited buffer of *L-sensor* nodes, power-aware compression [Marcelloni and Vecchio, 2008] and reduction algorithms [Aquino and Nakamura, 2009] may be employed. Finally, as an abuse of terminology, we also use the ID of a node as a reference to the data produced by the node. Hence, for the sake of presentation, the partial view v of a node i is a collection of IDs stored at node i .

Communication: We consider a connected network topology along the time. Given the expected network lifetime, we can estimate the amount of sensor nodes to achieve this goal. An *L-sensor* node i can communicate with another node j (*L-sensor* or *H-sensor*) that is inside its communication radius r_1 , i.e., the distance between i and j should be less than or equal to r_1 ($d(i, j) \leq r_1$). *H-sensor* nodes are equipped with two

radios, each one with a different frequency and a different communication radius (r_1 and r_2 , $r_2 \gg r_1$). It is also assumed that the radio frequencies used in both radios do not interfere with each other. An *H-sensor* node can communicate with both *L-sensor* and *H-sensor* nodes inside communication radius r_1 and r_2 , respectively.

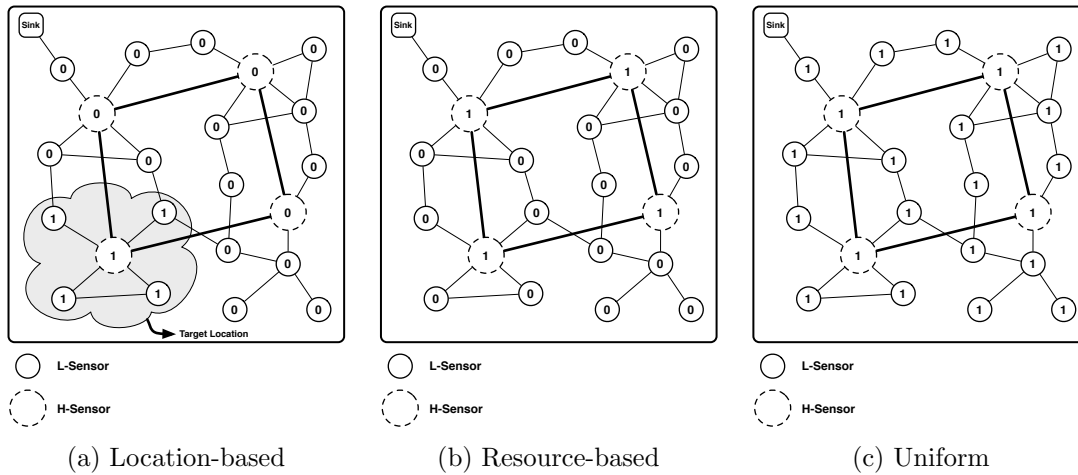


Figure 2.3: ProFlex enables different dissemination strategies depending on how the importance factor is assigned to each node in the network. However, in this thesis we will focus on a uniform dissemination strategy, therefore all nodes possess the same importance factor

Initial knowledge: Initially, a node i only knows its identity, which is unique, and a parameter $I(i)$ that defines its *importance factor* in the network ($I: S \rightarrow [0, 1]$, called the importance factor function). Importance factors are initially assigned to nodes based on an external criterion. It determines nodes in the network responsible for storing data. For instance, if the criterion is the sensor location, only nodes at the target location will be used as storing nodes and will have $I(i) > 0$, whereas nodes outside the target location will have $I(i) = 0$. In Figure 2.3a we show how the importance factor must be assigned if a location-based dissemination strategy is desired. On the other hand, it may be desired to choose as storing nodes only *H-sensor* nodes, since they are more powerful than *L-sensor* nodes. In this scenario, *H-sensor* nodes will have $I(i) > 0$ and *L-sensor* nodes will have $I(i) = 0$. Figure 2.3b shows the importance factor assignment for a resource-based dissemination strategy. Finally, if all nodes can be uniformly selected as storing nodes, i.e., equal storing load distribution, then all nodes in the network will have $I(i) = 1$. Figure 2.3c shows the importance factor assignment for a uniform dissemination strategy. For comparison reasons, this last scheme is used, hence for all *L-sensor* and *H-sensor* nodes, $I(i) = 1$. In summary,

the attribution of importance factors among nodes will define the *set of storing nodes* S .

2.4 Proposed Protocol

In this section, we present ProFlex, a Data Dissemination Protocol for Heterogeneous Wireless Sensor Networks with Mobile Sinks. The protocol is composed of three phases: tree construction, importance factor distribution, and data distribution. At the end of these three phases, ProFlex guarantees that a node in the set of storing nodes will store an amount of data proportional to its importance factor. For instance, if all nodes in the set of storing nodes have the same importance factor, then ProFlex guarantees a uniform distribution of network data among the nodes in the set of storing nodes. Algorithm 1 presents a general overview of the protocol.

Algorithm 1: General principle of ProFlex

- 1 Construction of trees T_h
 - 2 Propagation of importance factors and number of storing nodes at each tree T_h
 - 3 **foreach** *node* i **do**
 - 4 Send $data(i)$ to the root of T_h
 - 5 The root propagates each $data(i)$, $r(v)$ times according to the probabilities induced by the importance factors over the set of storing nodes
-

In summary, at first, it is assumed that for each sensor node in the network there is an assigned importance factor. This is used to define the dissemination strategy employed by the protocol. Since in this thesis our focus is on a uniform dissemination strategy, all nodes have the same importance factor (see Figure 2.4a). Then, each *H-sensor* node starts the construction of a tree rooted in itself (see Figure 2.4b). These trees are used in the last phase of the protocol to replicate and disseminate the data produced by the network. Thereafter, occurs the importance factor distribution, a process that begins at all leaf nodes and stops at the root of each tree. After this process, each node knows the value of the importance factor in each subtree (see Figure 2.4c). Then, when a node produces data, it generates a packet and forwards it to the *H-sensor* node at the root of its tree (see Figure 2.4d). When the *H-sensor* receives this packet, it replicates and forwards the packet to its own tree and to neighboring trees (see Figure 2.4e). Finally, when a node receives a replicated packet, it must decide whether to store the packet or to forward it to one of its child in the tree (see Figure 2.4f). In the

following sections a detailed description of these phases are presented. Moreover, for quick reference, Table 2.1 lists the main concepts of ProFlex.

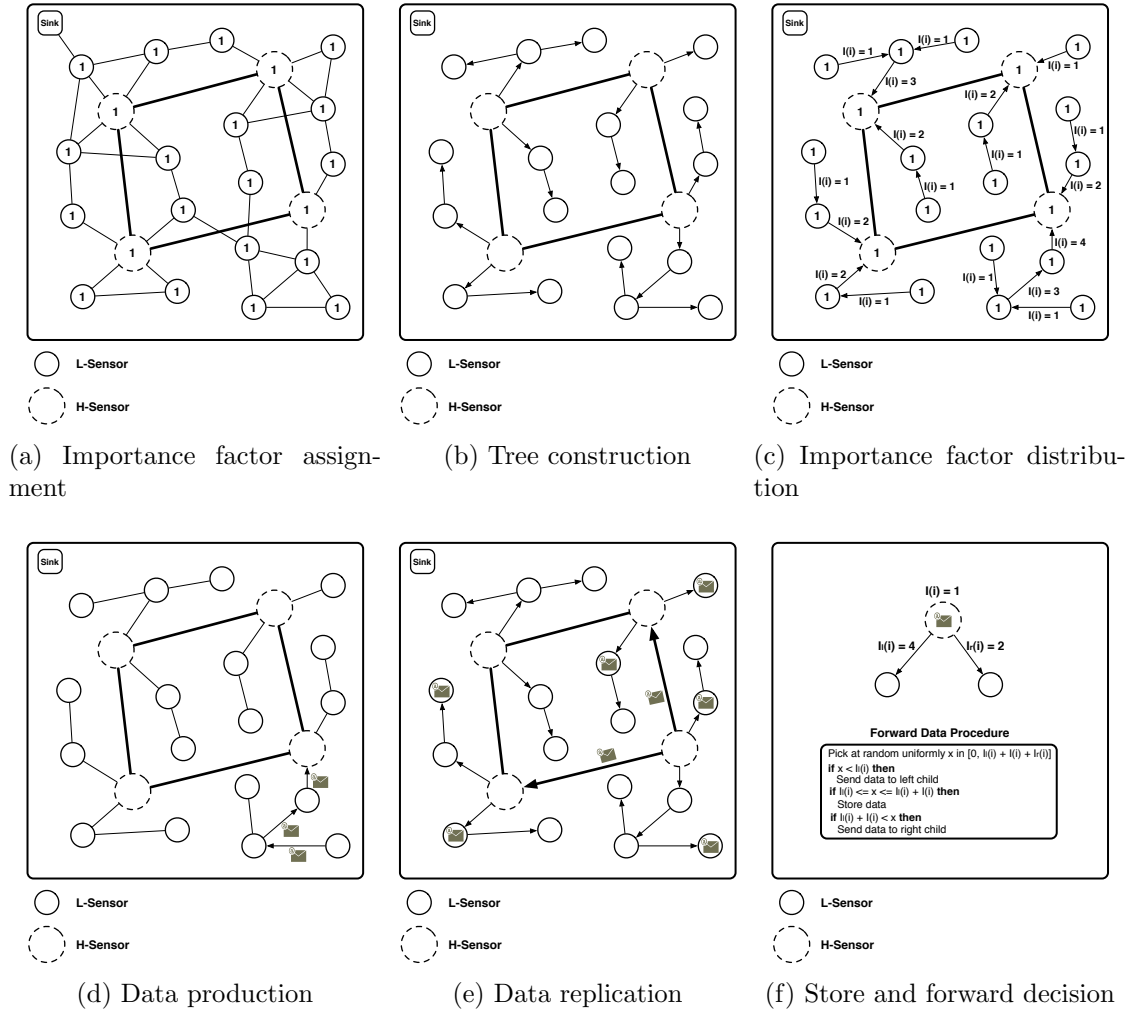


Figure 2.4: The main steps of ProFlex

2.4.1 Tree Construction

The first step of ProFlex is the tree construction initiated by all *H-sensor* nodes in the network. More specifically, contrarily to Supple [Viana et al., 2010], ProFlex deals with the problem of traffic overload at nodes closer to the tree root. For this end, multiple trees (i.e., replication structures) are constructed according to the number and positioning of the *H-sensor* nodes (Figure 2.5). These trees aggregate the shortest paths from each *L-sensor* node to the closest *H-sensor* node. In this work, the shortest path means the minimum number of hops between an *L-sensor* and its closest *H-sensor*

Table 2.1: Main concepts of ProFlex

| Concept | Description | Formula |
|----------------|---|---|
| n_L | Number of resource constrained sensors in the network (L-sensors) | |
| n_H | Number of powerful sensors in the network (H-sensors) | $n_L \gg n_H$ |
| r_1 | Range of the short-range radio. L-sensors just have a short range radio | |
| r_2 | Range of the long-range radio. H-sensors have both short and long-range radios | $r_2 \gg r_1$ |
| v | Partial view. It is the local buffer used to store packets | |
| S | The set of storing nodes | |
| $I(i)$ | Importance factor for node i . It determines the set of storing nodes (S) and influences the amount of data a node i will store. Here we use $I(i) = 1$ | $I: S \rightarrow [0, 1]$ |
| $I_l(i)$ | Importance factor of the left subtree of i , where such subtree is rooted at the left child j | $I_l(i) = I_l(j) + I_r(j)$ |
| $I_r(i)$ | Importance factor of the right subtree of i , where such subtree is rooted at the right child k of i | $I_r(i) = I_l(k) + I_r(k)$ |
| T_h | Binary tree rooted at H-sensor h | |
| $N(h)$ | All H-sensors that are neighbors of the H-sensor h | |
| $ S_h $ | Size of the set of storing nodes in the tree T_h | |
| $ S_{aggr}^h $ | Size of the aggregated set of storing nodes in the tree T_h and in the neighboring trees of H-sensor h | $ S_{aggr}^h = S_h + \sum_{j \in N(h)} S_j $ |
| $ v $ | Size of the partial view for all nodes in the tree of H-sensor h | $ v = \sqrt{ S_{aggr}^h }$ |
| $r(v)$ | Total number of replicas for a packet produced at a tree T_h | $r(v) = \sqrt{ S_{aggr}^h }$ |
| $r_k(v)$ | The number of replicas out of $r(v)$ that is sent to the tree of H-sensor k | $r_k(v) = \frac{ S_k }{ S_{aggr}^h } \times r(v)$ |

node, but any other metric can be used, i.e., delay, capacity, etc. Notice that, although each *H-sensor* node builds a tree rooted at itself, each *L-sensor* node will belong **only** to the tree rooted at the closest *H-sensor* node. Hence, during the tree construction, when an *L-sensor* node receives several *H-sensor*'s messages, it will update its local information and forward the message further only if the message is from a closer *H-sensor* node. In case there is a tie, i.e., there are two or more shortest paths, the *L-sensor* node will use a criterion (e.g., geographic location of the *H-sensor* node). Otherwise, the node will simply discard the message. For the sake of presentation and of comparison with Supple, we consider the use of a binary tree as a routing structure.

ProFlex supports, however, any other routing structure.

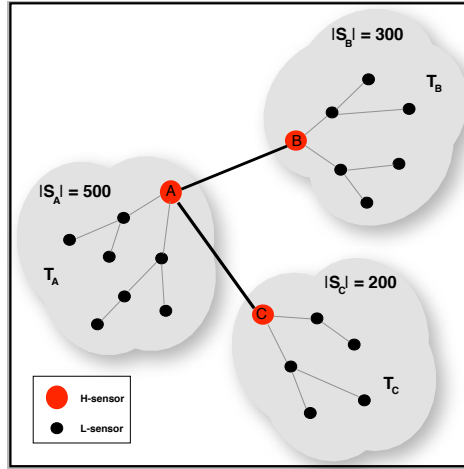


Figure 2.5: Network with 1000 sensor nodes of which 3 are *H-sensor* nodes

2.4.2 Importance Factor Distribution

In ProFlex, all nodes in the *set of storing nodes* have an *importance factor* assigned by the function $I: S \rightarrow [0, 1]$, which will be later used at the computation of node i 's storing probability. The importance factor assigned to a particular node i dictates whether node i will play the role of storing data for other nodes ($I(i) > 0$) or not ($I(i) = 0$). Furthermore, it dictates how much data a node should store, and its assignment may follow any distribution. For comparison reasons with the protocols described in Section 2.2, here we use the uniform distribution. Thus, for all nodes, $I(i) = 1$. This gives to all nodes in S equal chance of being assigned to the role of storing node and it also gives to all nodes equal chance of storing the same amount of data. Note that the importance factor of both *L-sensor* and *H-sensor* nodes are equal, thus they both have the same probability of storing data for other nodes. It is straightforward to notice that due to their better resource capabilities, *H-sensor* nodes could have a greater importance factor than *L-sensor* nodes. Therefore, the former would store more data than the latter, but we chose not to do so. Such a decision is supported by the fact that in this work, we only intend to leverage the communication characteristics inherent to heterogeneous WSNs and not the increased storage capacity of *H-sensor* nodes.

After defining the set of storing nodes, a key challenge is *how much data should a node store*. Stating another way, *what should be the partial view size $|v|$ of nodes in the set of storing nodes?* In existing protocols [Bar-Yossef et al., 2008; Viana et al.,

2010; Vecchio et al., 2010], the *partial view size* $|v|$ (i.e., maximum number of allowed stored packets at a given node) is a statically configured parameter. It is configured considering a uniform distribution of the data among the storing nodes and is based on the size of the set of storing nodes at a specific replication structure. By doing this, they ensure there will be enough space to store the data generated by the entire network. In particular, in Supple [Viana et al., 2010], a unique tree-based replication structure is considered. Therefore, if a uniform selection criterion is used, the size of the set of storing nodes will be equivalent to the number of nodes in the tree, i.e., n nodes. However, ProFlex uses several replication structures at the data distribution phase (see Section 2.4.3 and Figure 2.5), which are given by the multiple constructed trees. Thus, *each tree defines different storing nodes and sizes of set S* . This requires a dynamic configuration of the partial view size of nodes per replication structure since each structure will store a different amount of data.

In Supple [Viana et al., 2010], given a partial view of size $|v|$ and a network with n data producers, the set of storing nodes S must contain at least $\Theta(\frac{n}{v} \ln n)$ nodes in order to guarantee with high probability a good storage of all n collected data. On the other way round, the partial view size must be $v \geq \frac{n}{|S|} \ln n$. In fact, a partial view size $|v| = \sqrt{n}$ provides a good compromise between resilience and sensors' resource consumption when $|S| = n$, which is our case here [Bar-Yossef et al., 2008; Viana et al., 2010; Vecchio et al., 2010]. Notice that, the partial view size depends on the set size $|S|$ and the number of data producers n in the replication structure. Since ProFlex uses several replication structures rather than one, the partial view size $|v|$ of storing nodes will be different for each tree T . As will be discussed in the next section, an *H-sensor* node h stores in its tree T_h all data produced by nodes belonging to its own tree and by nodes belonging to *neighboring trees*. Neighboring trees are trees rooted at *H-sensor* neighbors of the *H-sensor* node h , denoted by $N(h)$. For instance, Figure 2.5 shows a network with three *H-sensor* nodes (A , B and C) and their respective trees. In that figure, the tree T_A rooted at *H-sensor* node A has 500 storing nodes ($|S_A| = 500$) and *H-sensor* node A has two *H-sensor* neighbors (B and C), hence it has two neighboring trees ($|N(A)| = 2$), i.e., T_B and T_C . Thus, later, *H-sensor* node A will store in T_A data produced by nodes in T_A and $T_{N(A)}$.

For the special case where there are n data producers, an *H-sensor* node only needs to know the number of storing nodes on its own tree and on the neighboring trees. This information may be piggybacked in packets during the importance factor distribution. Algorithm 2 shows how the size of the set of storing nodes and the importance factor are distributed. The main idea behind this algorithm is to initialize each node i with a tuple $(I_l(i), I(i), I_r(i))$, where $I_l(i)$ (similarly to the third component

$I_r(i)$ is the importance factor of the left (similarly to the right) subtree of i , and $I(i)$ is the importance factor of node i . Note that $I_l(i)$ (similarly to $I_r(i)$) is the sum of all importance factors of nodes in the left (right) subtree of node i .

Algorithm 2: Importance factor distribution algorithm

```

1 foreach node  $i$  do
2   create  $(I_l(i), I(i), I_r(i))$ 
3    $I_l(i) = I_r(i) = 0$ 
4 foreach node  $i$  in a depth-first search do
5   if  $j = \text{left child of } i$  then
6      $I_l(i) = I_l(j) + I(j) + I_r(j)$ 
7   if  $k = \text{right child of } i$  then
8      $I_r(i) = I_l(k) + I(k) + I_r(k)$ 
9 foreach  $H$ -sensor node  $h$  do
10   $|S_h| = \text{Size of } h\text{'s set of storing nodes}$ 
11  Send  $|S_h|$  to  $h$ 's  $H$ -sensor neighbors
12 foreach  $H$ -sensor node  $h$  do
13   $|S_{aggr}^h| = |S_h| + \sum_{j \in N(h)} |S_j|$ 
14 foreach  $H$ -sensor node  $h$  do
15   $|v| = \sqrt{|S_{aggr}^h|}$ 

```

When the H -sensor node h knows the size of the set of storing nodes $|S_h|$ in its tree, it will forward this value to its H -sensor neighbors. For the special case where $I(i) = 1$ for all nodes, then the sum $I_l(h) + I(h) + I_r(h)$ gives the size of the set of storing nodes $|S_h|$. Eventually, node h will also receive this value from its neighboring H -sensor nodes. Finally, node h calculates the size of the set of aggregated storing nodes $|S_{aggr}^h|$, i.e., its own set of storing nodes plus the set of storing nodes at its neighboring trees: $|S_{aggr}^h| = |S_h| + \sum_{j \in N(h)} |S_j|$. Based on this information, node h calculates the partial view size $|v| = \sqrt{|S_{aggr}^h|}$ for storing nodes on its tree and embeds this value on every data packet replicated on its own tree (see Section 2.4.3). Using this information, a node in the set of storing nodes knows the maximum volume of data it can store. Summarizing, an H -sensor h initially builds its tree T_h and computes the number of storing nodes $|S_h|$ in its tree. Then it forwards $|S_h|$ to H -sensor neighbors and eventually receives the number of storing nodes in the neighboring trees. Finally, using $|S_h|$ and the number of storing nodes in neighboring trees, H -sensor h computes the partial view size $|v|$ and embeds this value on every data packet replicated on its own tree.

For instance, in Figure 2.5, for the *H-sensor* node A , we have $|S_A| = 500$ and $\sum_{j \in N(A)} |S_j| = 300 + 200$, hence $|S_{aggr}^A| = 1000$ and $|v| = 31$. Thus, all nodes in A 's tree will have a partial view size $|v| = 31$. *H-sensor* nodes B and C will also execute these same steps to compute the value $|v|$ for their trees.

2.4.3 Data Distribution

The data distribution phase is at the heart of ProFlex and is responsible for properly disseminating the sensed data to the set of storing nodes. For scenarios in which equal importance factors are assigned to nodes, ProFlex ensures a uniform distribution of the network sensed data among the set of storing nodes. In fact, the partial view v of nodes is constructed due to the distribution of $r(v)$ replicas of each data packet. More specifically, the transmission of $r(v)$ replicas by each root h of each tree T_h will guarantee the storage of $|v| = \sqrt{|S_{aggr}^h|}$ data packets at each storing node of each tree.

Algorithm 3 shows the main steps a node must perform when it produces or receives a data packet. Initially, when an *L-sensor* node produces a data packet or receives a data packet from a child node, it just forwards the data to its parent until it reaches the *H-sensor* node that is at the root of its tree.

Algorithm 3: Data distribution algorithm

```

1 if L-sensor node  $i$  produces data then
2   | Send data to parent
3 if L-sensor node  $i$  receives data from child then
4   | Forward data to parent
5 if H-sensor node  $h$  produces data or receives data from L-sensor then
6   | Compute  $r_k(v)$  to each H-sensor  $k \in N(h) \cup \{h\}$ 
7   | Send  $r_k(v)$  data replicas to each H-sensor  $k$  in  $N(h)$ 
8   | Call ForwardData(data)  $r_h(v)$  times
9 if L-sensor node  $i$  receives data from parent then
10  | Call ForwardData(data)
11 if H-sensor node  $h$  receives data from another H-sensor neighbor then
12  | Call ForwardData(data)

```

When an *H-sensor* node h produces a data packet or receives one from a child *L-sensor* node, it first computes the number of replicas $r(v)$ for the packet to be forwarded to its children and *H-sensor* neighbors. Such computation ensures that nodes belonging to the set of storing nodes receive with high probability $|v|$ distinct data packets.

As shown in [Bar-Yossef et al., 2008; Viana et al., 2010], the number of replicas $r(v)$ is computed based on the desired partial view size of nodes in the set of storing nodes. For the special case where $|v| = \sqrt{|S_{aggr}^h|}$, then $r(v) \approx \sqrt{|S_{aggr}^h|}$. Hence, in order to compute the number of replicas for a data packet, an H -sensor h needs to know the number of storing nodes on its own tree and on neighboring trees. Such information was already computed in Algorithm 2 to determine $|v|$, thus $r(v) = \sqrt{|S_{aggr}^h|}$.

After computing the number of replicas $r(v)$ for a given data packet, H -sensor h determines how many replicas from $r(v)$ goes to its own tree and to each neighboring tree. This is proportional to the number of storing nodes in each tree with respect to the total number of storing nodes in S_{aggr}^h . Let $r_k(v)$ be the number of replicas for a tree T_k for $k \in N(h) \cup \{h\}$, thus $r_k(v) = \frac{|S_k|}{|S_{aggr}^h|} \times r(v)$. After determining the number of replicas, H -sensor h sends $r_k(v)$ data replicas to each H -sensor $k \in N(h)$.

For instance, in Figure 2.5, H -sensor A calculates $r(v) = 31$. Therefore node A sends $r_B(v) = \frac{300}{1000} \times 31 = 9$ replicas of each data to node B , $r_C(v) = \frac{200}{1000} \times 31 = 6$ replicas of each data to node C and $r_A(v) = \frac{500}{1000} \times 31 = 16$ to its own tree.

Finally, H -sensor h calls **ForwardData** (Algorithm 4) $r_h(v)$ times. The propagation by the H -sensor node is done according to the importance factor of its left and right subtrees and also to its own importance factor. To understand how is the **ForwardData** operation, in Figure 2.6 a node h receives a data packet and needs to make a decision whether the packet should be stored locally, or should be forwarded to the left or right subtree. Thus, node h sums the value of its own importance factor ($I(h) = 1$) and the values of the importance factor of the left ($I_l(h) = 12$) and right ($I_r(h) = 18$) subtrees. The total sum is equal to 31. Then, node h picks uniformly and randomly a value x in the interval $[0, 31]$. If $x < 12$, then it forwards data to left subtree. If $12 \leq x \leq 13$, then node h stores the packet. Otherwise, it forwards the packet to the right subtree.

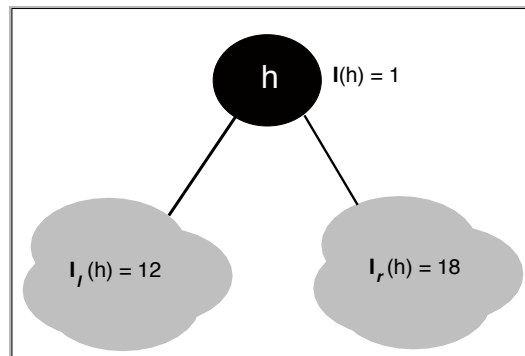


Figure 2.6: Forward data example

Algorithm 4: ForwardData(*data*) procedure

```

1 Pick at random uniformly  $x \in [0, I_l(i) + I(i) + I_r(i)]$ ;
2 if  $x < I_l(i)$  then
3   | Send data to left child
4 if  $I_l(i) \leq x \leq I_l(i) + I(i)$  then
5   | Store data in own view
6 if  $I(i) + I_l(i) < x$  then
7   | Send data to right child

```

Moreover, when an *L-sensor* node receives a data packet from its parent or an *H-sensor* receives a data packet from a neighboring *H-sensor* node, it also calls **ForwardData** to determine whether it will forward or store the packet. The algorithm stops when a leaf node in the set of storing nodes receives the message. At the end of the data distribution, all nodes of the set of storing nodes will have, with high probability, a partial view size $|v|$.

2.5 Performance Analysis

In order to assess the proposed protocol, we performed a series of simulations. As stated earlier, ProFlex operates under a heterogeneous WSN made up of two kinds of sensor nodes, *H-sensors* and *L-sensors*. Also, *H-sensor* nodes have an increased communication capacity when compared with *L-sensor* nodes. Therefore, the first step of our performance analysis was to find out what should be the number of *H-sensor* nodes (n_H) in the network and its communication radius (r_2) in order to achieve a good trade-off between data gathering efficiency and message overhead.

Thereafter, ProFlex was compared with the storage protocols described in Section 2.2, named Supple [Viana et al., 2010], RaWMS [Bar-Yossef et al., 2008], and Deep [Vecchio et al., 2010], under different simulation scenarios. Finally, we proposed and evaluated an improvement on ProFlex to decrease even further the protocol's overhead.

All simulations were performed using the Sinalgo [Group, 2008] simulator, version 0.75.3. Since this simulator does not implement the medium access control and physical layers, scenarios with message losses were simulated by defining a probability for each node losing a message. For instance, for a probability p , every time a node i receives a message m , i drops message m with probability p . Hereafter, all results are the arithmetic mean of a number of simulations necessary to accomplish a confidence

interval of 95%. Each simulation comprises a dissemination phase where each node produces a single data packet, and a gathering phase. In the first phase, each node executes a data storage protocol for performing data distribution over the network, according to each considered storage protocol. Then, in the gathering phase, a mobile sink collects data from the storing nodes. In particular, the sink performs as many visits as necessary to get a representative amount of data from the network, i.e., to get n different entries of storing node's views. The main adopted parameters are shown in Table 2.2. Notice that some of the presented parameters are exclusive for some of the literature's protocols.

Table 2.2: Simulation parameters

| Parameter | Value |
|--|------------------------------|
| Number of sensor nodes ($n_L + n_H$) | 1000 |
| Sensor field | $800 \times 800 \text{ m}^2$ |
| Network density | 20 |
| L-sensor communication range | 60 m |
| Importance Factor $I(i)$ | 1 |
| $ v $ - (Deep, RaWMS and Supple) | 31 |
| $r(v)$ - (RaWMS and Supple) | 31 |
| TTL - (RaWMS) | 125 |
| β - (Deep) | 5.4 |

2.5.1 ProFlex Assessment

A question someone using ProFlex might ask is how many H -sensor nodes (n_H) should be employed in the network and what should be the communication radius (r_2) between them. Moreover, what is the impact that these two parameters have on ProFlex's performance. Clearly, as stated in Section 2.3, the number of H -sensor nodes (n_H) should be much lower than the number of L -sensor nodes (n_L) or the cost to deploy the network will end up too high. Furthermore, the communication radius between H -sensor nodes should not exceed the technological limits imposed by current wireless communication radio interfaces.

With these issues in mind, for a given number of H -sensor nodes (n_H) and a given communication radius (r_2) between them, we evaluated ProFlex's data collection efficiency and message overhead. By data collection efficiency, we mean that, after the distribution of data to the network, a mobile sink placed at a random position visits the node at this position and then chooses the next position to visit, as described in [Friedman et al., 2008]. When visiting a node, the mobile sink gathers all data stored

at the partial view of this node. The protocol that collects 100% of the data by visiting the lowest number of nodes is the protocol with the best data gathering efficiency. Also, by message overhead, we mean the total number of transmitted messages necessary to distribute the network data to selected storage nodes.

Figure 2.7 shows the collection efficiency and the message overhead for different values of n_H (5, 10, 15, 20) and r_2 (120, 180, 240, 300, 360, 420, 480). It is worth noticing the values chosen for n_H satisfies the condition that $n_L \gg n_H$, i.e. for a network with a 1000 nodes, then at most 2% of the nodes are *H-sensor* nodes. In the figure, each curve represents a value for n_H and each marked point represents a value for r_2 . Also, the x-axis accounts for the number of nodes the mobile sink must visit in order to collect 90% of the network data, while the y-axis accounts for the total number of transmitted messages in order to distribute the network data to storing nodes. For instance, the triangle point of the dotted green line represents ProFlex with 5 *H-sensor* nodes and a communication radius between them equal to 480 m. Thus, with this configuration the mobile sink needs to visit about 100 nodes in order to collect 90% of the network data and ProFlex transmits almost 240000 messages in order to distribute the network data to the storage nodes. Looking at this figure it becomes clear that there is a trade-off between data collection efficiency and overhead, i.e., for a better collection efficiency, it is necessary to transmit more messages during the data distribution.

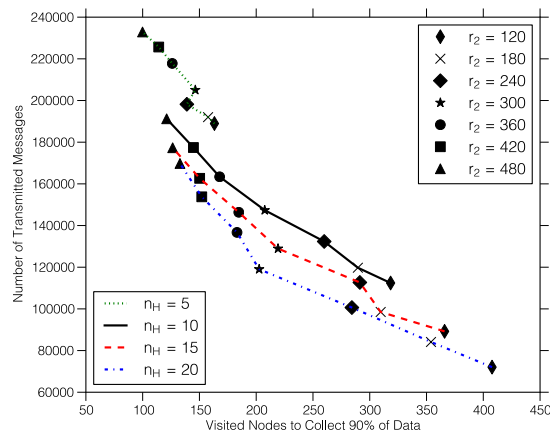


Figure 2.7: Trade-off between data gathering efficiency and messages overhead

Varying the number of *H-sensor* nodes: Looking at n_H in isolation, we conclude the following (fix on a symbol in the graph and then look for this symbol in the four different lines). By increasing n_H , we decrease the data collection efficiency (it

becomes worse) and also decrease the protocol's overhead. For instance, keeping $r_2 = 120$ m, when $n_H = 5$, ProFlex has a collection efficiency of about 170 nodes and an overhead of almost 190000 messages. But, when $n_H = 20$, ProFlex's collection efficiency becomes much worse, about 420 nodes, and the overhead also decreases to almost 70000 messages. The explanation for such a behavior is subtle and is related to the way ProFlex computes the number of replicas $r(v)$ for each data packet (cf. Section 2.4.3). For instance, consider the network in Figure 2.5. We know that $r_A(v) = \sqrt{1000} = 31$; $r_B(v) = \sqrt{800} = 28$; and $r_C(v) = \sqrt{700} = 26$. Hence, each *H-sensor* node replicates $r_A \times |S_A| = 15500$; $r_B \times |S_B| = 8400$; $r_C \times |S_C| = 5200$, and the total number of replicas in the network is equal to $15500 + 8400 + 5200 = 29100$ data replicas. Now, imagine the network in Figure 2.5 has only the *H-sensor* node *A*, with $|S_A| = 1000$. Hence, $r_A(v) = \sqrt{1000} = 31$, and $r_A \times |T_A| = 31 \times 1000 = 31000$ data replicas, or 1900 more data replicas than the former case, showing that an increase on n_H results in a decrease of the overhead. Finally, as stated earlier, with a decrease of the overhead there is also a decrease of the data collection efficiency.

Varying communication radius between *H-sensor* nodes: Now, looking at r_2 in isolation we conclude the following (fix on a line in the graph and then look for the 7 different symbols). By increasing r_2 , we increase the data collection efficiency and also increase the protocol's overhead. This behavior is due to the fact that with a greater r_2 , data packets can be disseminated farthest away from its origin point, enabling the data dissemination to more spots in the network and thus, improving the data gathering efficiency. However, in order to reach more spots in the network, each packet must be relayed more times until it finds its final destination, thus increasing ProFlex's message overhead.

Based on the above result, it becomes clear that a network designer using ProFlex must decide if the main requirement is a good data collection efficiency or a low overhead, i.e., accomplishing a competing data collection efficiency at the cost of a high overhead, or accomplishing a very low overhead at the cost of a worse data collection efficiency. Considering the nature of WSN applications, we argue that in most cases the focus on reducing the message overhead as much as possible is the prevailing one. Furthermore, a not so good data collection efficiency can be easily circumvented, for instance, by employing more than one mobile sink. Hence, since our focus is on reducing ProFlex's overhead and also accomplishing a competing data collection efficiency, hereafter, all simulations were performed using a heterogeneous network composed of $n_L = 990$ *L-sensor* nodes and $n_H = 10$ *H-sensor* nodes with communication radius between *H-sensor* nodes equal to $r_2 = 360$ m.

2.5.2 ProFlex vs. Literature Protocols

This section evaluates the ProFlex behavior compared with existing protocols in the literature, more specifically, Supple [Viana et al., 2010], Deep [Vecchio et al., 2010] and RaWMS [Bar-Yossef et al., 2008]. Those protocols were chosen due to their similarities with ProFlex or due to their first-class performance. It is worth noting that in our evaluation, the literature protocols also operate under a heterogeneous WSN with the same parameters (n_H and r_2) used by ProFlex even if they were not designed for this kind of network. This fact has no negative effect on the evaluated protocols. On the contrary, it leads to a fair comparison with ProFlex.

Data gathering efficiency: Figure 2.8 shows the data gathering efficiency for all protocols in a scenario with no message loss nor node failure. As can be observed, ProFlex has a slightly worse data gathering efficiency than the literature protocols. The explanation for such a result can be attributed to the fact that ProFlex transmits less messages than the other protocols (as will be shown later in this section) and, consequently, there are less data replicas throughout the network. Moreover, contrary to the literature protocols, a data packet produced in the tree of one *H-sensor* node is only replicated on its own tree and on its neighboring trees. This is associated to the fact that not every *H-sensor* node is connected to each other limits the regions in the network where a data packet will be replicated. Such a fact is presented in Figure 2.9. Here, we divided the network sensor field in 16 cells of equal size and computed for a data packet the number of cells it was found by the mobile sink. As can be observed, the literature protocols disseminate most of the data at about 14 cells in the network (i.e., 87,5% of network cells), while in ProFlex the number of cells a data packet can be found is about 8 (i.e., 50% of network cells). Such a difference clearly presents an impact on the data gathering efficiency of ProFlex. However, as discussed earlier, such a difference between ProFlex's data gathering efficiency when compared with existing protocols was anticipated and actually purposeful, since our main focus here was to keep ProFlex's overhead as low as possible. If the data gathering efficiency becomes an issue, a possible alternative is to employ more than one mobile sink to perform the data gathering or, in detriment of the energy, to increase the number of *H-sensor* nodes to be deployed (as discussed in the previous section).

Loss and failure robustness: Figure 2.10 shows the data gathering efficiency under a scenario with message loss. The protocol that was presenting the best results so far, Supple, does not perform as well under a message loss scenario as it did under a reliable scenario, but it still can recover 100% of the data. ProFlex still presents a worse

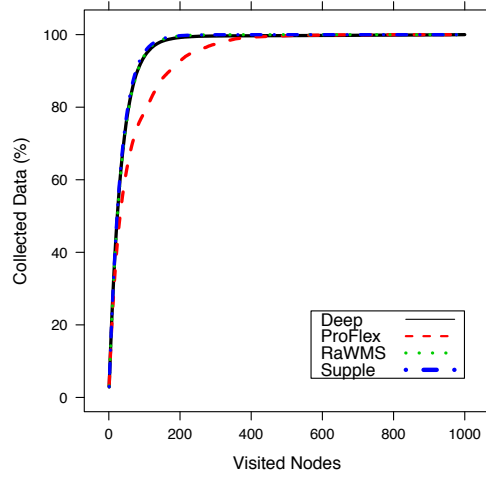


Figure 2.8: Data gathering efficiency for a scenario with no message loss nor node failure

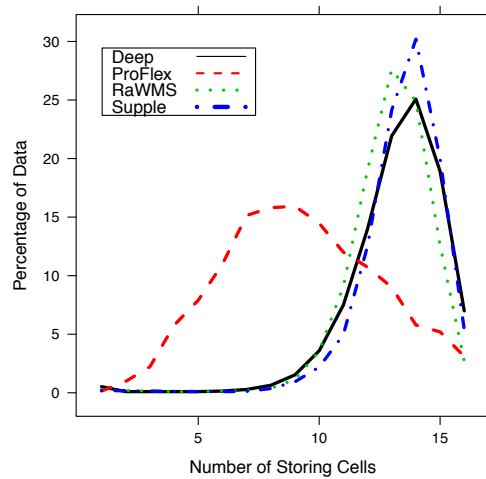


Figure 2.9: Data dissemination efficiency for a scenario with no message loss nor node failure

data gathering efficiency than Deep and RaWMS, but performs better than Supple. Actually, it is possible to see that when the message loss probability increases, the data gathering efficiency of ProFlex, Deep and RaWMS is almost unaffected. Hence, we can conclude that message loss does not have as much effect on ProFlex, Deep and RaWMS as it does on Supple.

Under a scenario with node failure the results change a little bit. As shown in Figure 2.11, both ProFlex and Supple are affected by node failures, whereas Deep and RaWMS are practically unaffected. The explanation for such a result can be attributed

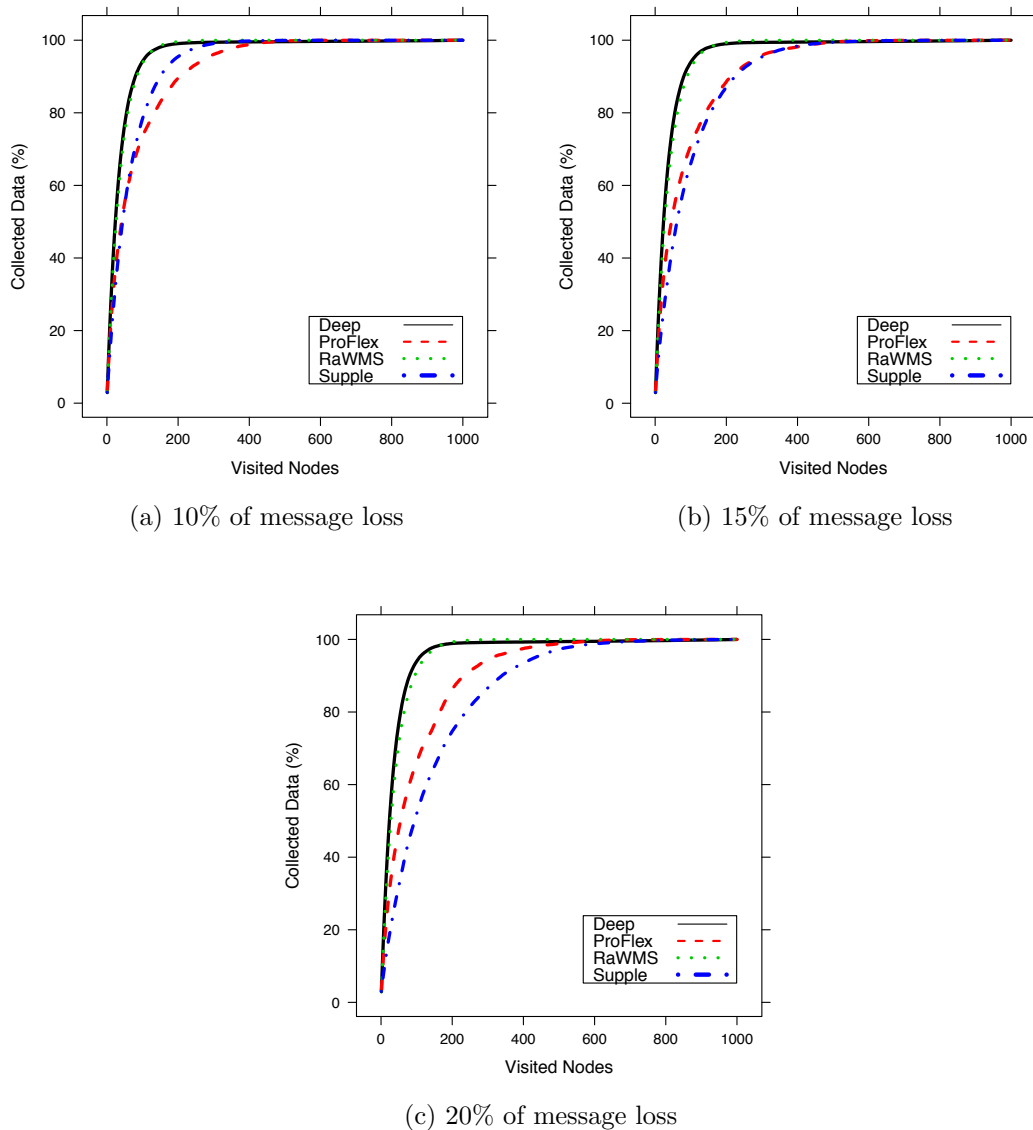
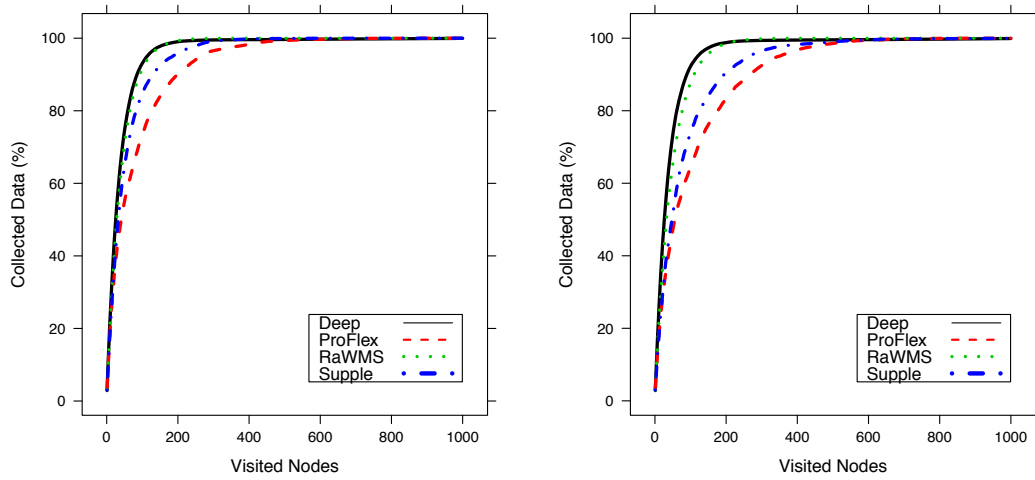


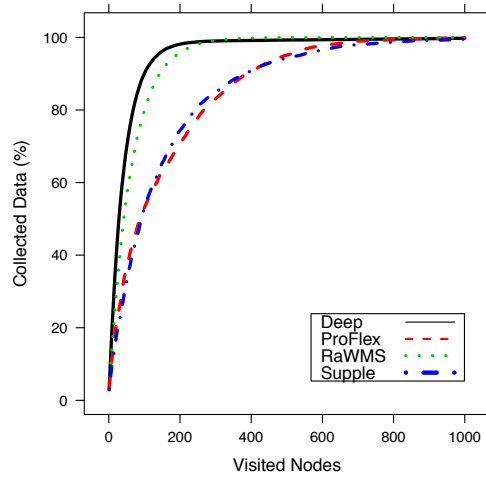
Figure 2.10: Data gathering efficiency for a scenario with message loss

to the fact that both ProFlex and Supple use a tree for doing data dissemination while Deep and RaWMS do not rely on any kind of predefined replication structure, i.e., RaWMS relies on random walks and Deep uses a gossip-based communication. Thus, when a node in the tree fails, all of its children are also compromised, preventing all of them from receiving any data packet. However, such a drawback can be easily circumvented by using some kind of salvation mechanism like the one employed by RaWMS, i.e., when a node forwards a message in the RW and does not receive an acknowledgement, then it forwards the message to another neighbor. Finally, it is worth noticing that although ProFlex's data gathering efficiency is still worse when



(a) 5% of nodes failing

(b) 10% of nodes failing



(c) 20% of nodes failing

Figure 2.11: Data gathering efficiency for a scenario with node failure

compared with Deep and RaWMS under loss and failures scenarios, ProFlex does not lose data, i.e., the mobile sink is able to gather 100% of the network data. Hence, the same alternatives to improve ProFlex’s data gathering efficiency under a reliable scenario may also be employed under loss and failure ones.

Energy hole vulnerability: A well-known problem in WSNs is the energy hole problem in which nodes closer to the sink tend to consume its energy resources faster than other nodes, since they have to route packets from all other nodes in the network [Li and Mohapatra, 2007; Liu et al., 2010; Wu et al., 2008]. In ProFlex and Supple (Deep

and RaWMS do not use trees for data dissemination), when a node has data to distribute to the network, it first sends the data to the root of the tree, and only then the root node will be responsible for distributing data to the network. Hence, nodes closer to the root node tend to route more packets than other nodes, resulting in the energy hole problem.

In order to evaluate the energy consumed by nodes according to their depth, Figure 2.12 shows the average number of data packets sent by a node as a function of its depth in the binary tree, in a scenario with no message loss or node failure. As expected, the closer the node is to the root, the more messages it has to send. However, in ProFlex, each *H-sensor* node is also a root node, and, consequently, more trees for data distribution are created in the network. The overhead is, thus, distributed among the trees, alleviating the impact of the energy hole problem. Despite Supple is also operating under a heterogeneous infrastructure, it is not tailored to take advantage of the features provided by this kind of network, opposed to ProFlex. For instance, when the node's depth is 1, ProFlex sends 91% less data packets than Supple. Therefore, we can conclude that increasing the number of *H-sensor* nodes, decreases the chances of the energy-hole problem.

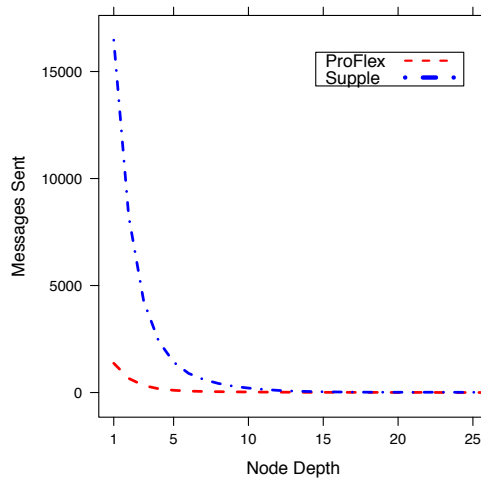


Figure 2.12: Average number of messages sent by a node as a function of its depth in the tree

Total communication overhead: Table 2.3 shows the number of transmitted messages for all evaluated protocols. As can be observed, besides mitigating the energy hole problem, ProFlex is the protocol with the lowest overhead. For instance, ProFlex transmits about 95% less messages than RaWMS. When compared with Supple and

Deep, the decrease is about 47% and 53%, respectively. Assuming that communication is the main activity responsible for the energy consumption in WSNs, these results are a strong indication that ProFlex will incur the lowest energy consumption for sensor nodes, thus increasing the network lifetime.

Table 2.3: Protocols overhead

| Protocol | Total Messages Transmitted |
|----------|----------------------------|
| ProFlex | 163390 |
| Supple | 310934 |
| Deep | 348046 |
| RaWMS | 3587087 |

2.5.3 Improving ProFlex

The results presented so far show that when compared with the literature protocols, ProFlex does not suffer from the energy hole problem as much as Supple, it is the protocol with the lowest overhead, its data gathering efficiency is competitive and can be easily managed by a network operator when deploying the *H-sensor* nodes (such flexibility is not allowed in the literature protocols), and, finally, performs well under scenarios with message loss and node failure. However, there is one characteristic inherent to WSNs that to the best of our knowledge was not leveraged by any existing state-of-the-art protocol, and that can decrease even further the overhead of ProFlex, if it is explored. Due to the nature of WSN applications and the highly dense deployments, the readings of sensor nodes can have a high degree of correlation and redundancy [Nakamura et al., 2007; Vuran et al., 2004; Akyildiz et al., 2004; Pattem et al., 2008]. Therefore, before starting the data distribution to storing nodes it is possible to apply some kind of summarization function to correlated data packets and only then start the data distribution.

To accomplish this, a small modification to Algorithm 3 is necessary. When a sensor node produces a data packet, it forwards the data to the root of the tree just like before. But when an *H-sensor* node receives a packet from its children, it does not replicate the packet immediately but rather it buffers the received packet locally and waits for a period t . After t expires, the *H-sensor* node summarizes the correlated packets that are locally buffered into a single packet, and replicates it onto its own tree and neighboring trees just like before. Here, we assume that a collection of buffered data packets are correlated if the distance between nodes that produced these packets is smaller or equal to a predefined value d . Hence, when a node produces a data packet,

it inserts its position into the packet's header. Thus, it is assumed that every sensor node knows its position. For instance, Figure 2.13 shows a network comprised of four nodes. Assuming $d = 10$, when an *H-sensor* node receives the data packets from these nodes it will figure out that only data packets belonging to the nodes A, B, and C are correlated since the distance between them is lower than or equal to d . Then, the *H-sensor* node summarizes the readings of nodes A, B and C using a summarization function like average, builds a packet with the summarized value and replicates to the network. Since the data from node D is not correlated to any other node, its value is replicated unchanged. This strategy reduces the total number of transmitted messages, and the question is to determine this amount.

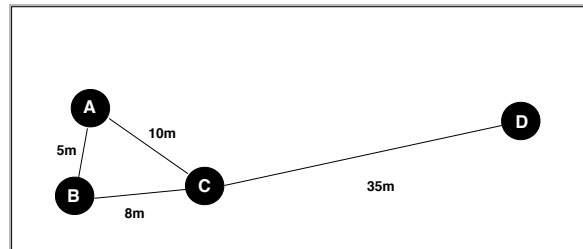


Figure 2.13: Example of a network with correlated data when $d = 10$

Table 2.4 shows the overhead of ProFlex and of four versions of ProFlex with data summarization. This result shows that by controlling the parameters t and d it is still possible to reduce ProFlex's overhead. For instance, when $d = 30$ and $t = 4$, the overhead is reduced by 21%. Moreover, Figure 2.14a shows that even transmitting less messages than the original ProFlex, the versions with data summarization still possess a data gathering efficiency comparable to the original version. This result is a bit surprising since it was thought that the decrease in the number of transmitted messages and the summarization itself would make the data dissemination a little worse, fact that did not prove itself as can be observed in Figure 2.14b.

Table 2.4: Overhead of some versions of ProFlex

| Protocol | Total Messages Transmitted |
|---------------------------|----------------------------|
| ProFlex | 163390 |
| ProFlex - $d = 10, t = 1$ | 161847 |
| ProFlex - $d = 10, t = 4$ | 157786 |
| ProFlex - $d = 30, t = 1$ | 151000 |
| ProFlex - $d = 30, t = 4$ | 127672 |

Figures 2.15 and 2.16 show the performance of ProFlex and its versions with data summarization in a scenario with message loss and node failures, respectively. These

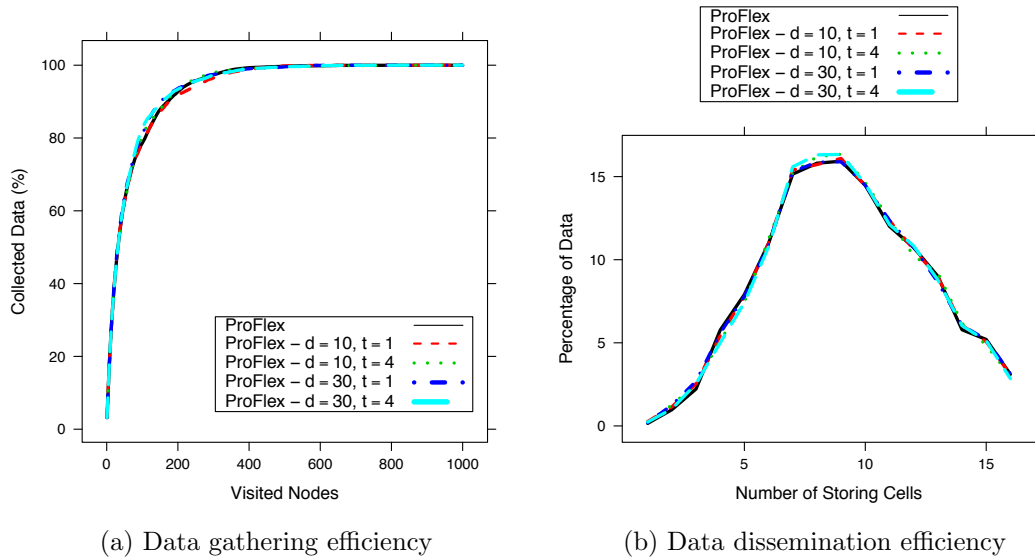


Figure 2.14: Data gathering efficiency and data dissemination efficiency for some versions of ProFlex under a reliable scenario

results show that the new versions of ProFlex perform exactly the same as the original version. When a summarized packet is lost, it is not the data of a single node that is being lost but rather the information of all nodes that were considered as being correlated to each other. However, the replication mechanism ensures that even if a packet is lost, it will still be possible to recover this same packet from some other part of the network.

From the aforementioned results, we conclude that, by employing data summarization, ProFlex’s overhead is reduced without affecting its data gathering efficiency. In fact, by choosing the right values for n_H , r_2 (cf. Section 2.5.1), d , and t , it is possible to come up with a version of ProFlex with a data gathering efficiency comparable to existing protocols, and still with a lower overhead. When using $n_H = 5$ and $r_2 = 480$ m, we showed in Figure 2.7 that ProFlex presents a high data gathering efficiency at the cost of a high overhead. Nevertheless, if the data summarization extension is applied in such configuration, for instance $d = 30$ m and $t = 4$ s, a lower overhead can be obtained. Figure 2.17 shows the data gathering efficiency for this version of ProFlex ($n_H = 5$, $r_2 = 480$ m, $d = 30$ m and $t = 4$ s) when compared with existing protocols, for a scenario with no message loss nor node failure, and Table 2.5 shows the resulting overhead. As can be observed, ProFlex now presents a comparable data gathering efficiency and sends about 43% less messages than Supple, the existing protocol with the lowest overhead. It is worth noticing, however, that the efficiency of the data correlation mechanism employed by ProFlex is subject to the precision of nodes’

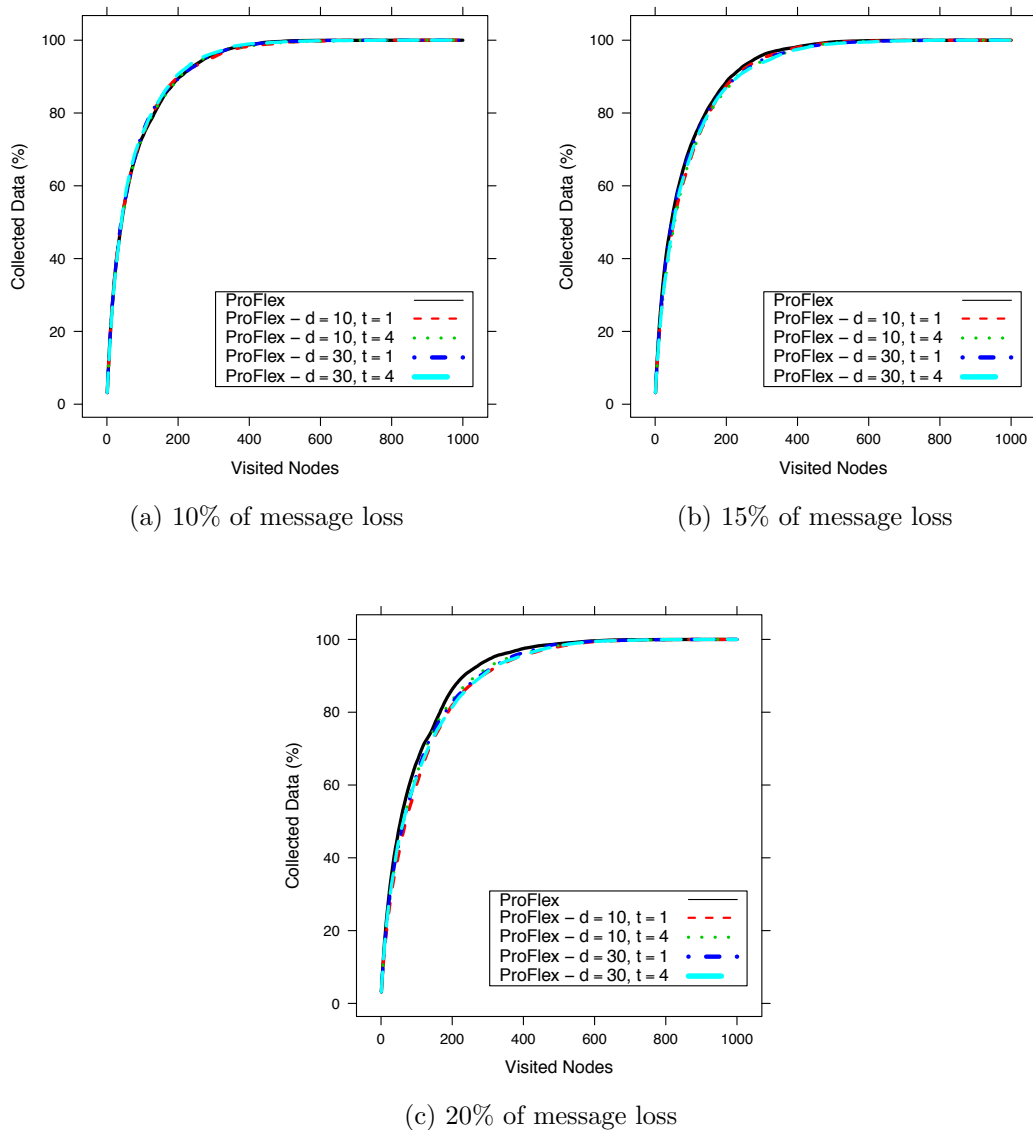
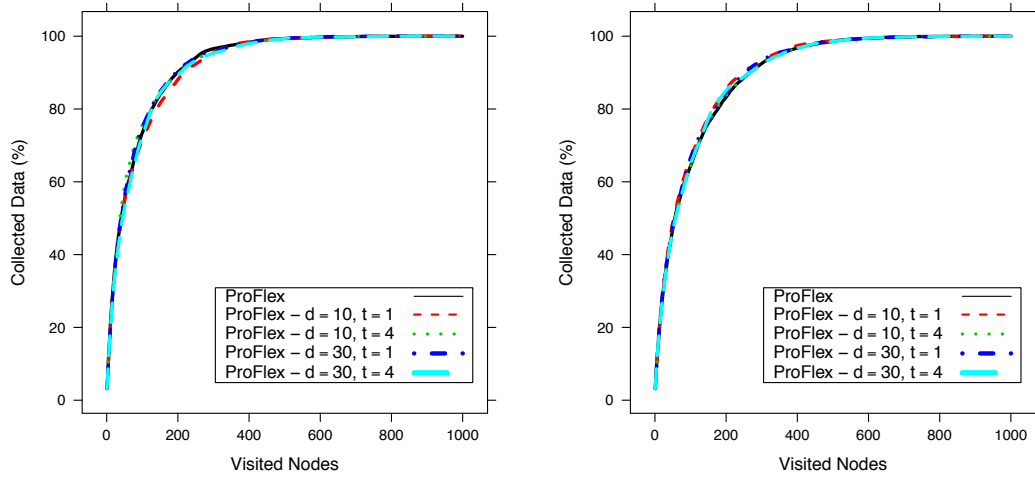


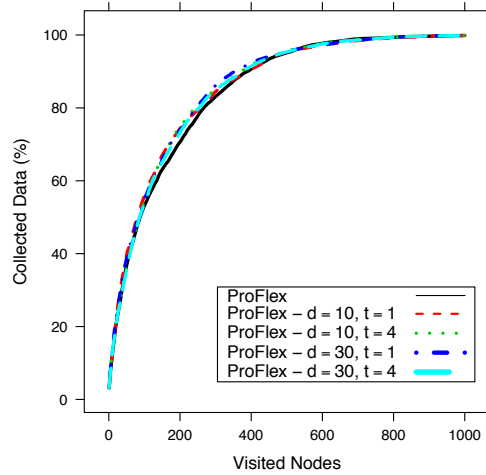
Figure 2.15: Data gathering efficiency for some versions of ProFlex under a scenario with message loss

positions. For instance, in the presence of localization errors, the mechanism may consider that data produced by some nodes are correlated (their reported positions lead the protocol to assume that the data are correlated), when in fact they are not (the actual nodes' positions show that the data are not correlated). The inverse may also happen, i.e., assume that data from some nodes are not correlated when in fact they are. An alternative approach to circumvent this problem is to look at the semantics of the data directly to decide whether they are correlated or not, and to not rely on the nodes' positions.



(a) 5% of nodes failing

(b) 10% of nodes failing



(c) 20% of nodes failing

Figure 2.16: Data gathering efficiency for some versions of ProFlex under a scenario with node failure

Table 2.5: Overhead in a network with $n_H = 5$ and $r_2 = 480$ m

| Protocol | Total Messages Transmitted |
|---------------------------|----------------------------|
| ProFlex - $d = 30, t = 4$ | 178975 |
| Supple | 316578 |
| Deep | 349197 |
| RaWMS | 3574989 |

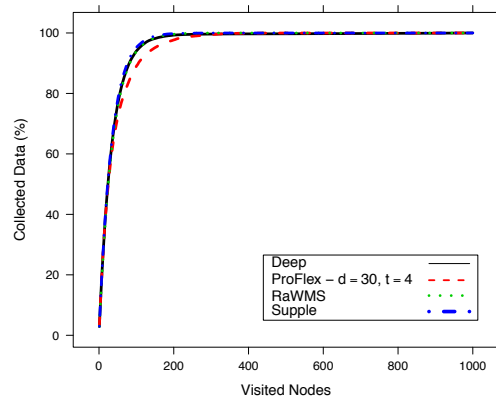


Figure 2.17: Data gathering efficiency for a scenario with no message loss nor node failure and a network with $n_H = 5$ and $r_2 = 480$ m

2.6 Chapter Remarks

This chapter presented ProFlex, a distributed data storage protocol for heterogeneous WSNs. We showed that the use of a heterogeneous infrastructure virtually reduces the occurrence of hot spots in the network. For instance, nodes closer to the root of the trees relay about 91% less messages when compared with Supple. Moreover, ProFlex is the protocol with the lowest overhead when compared with state-of-the-art protocols and is still able to accomplish a competitive data gathering efficiency. For instance, ProFlex transmits about 95% less messages than RaWMS and 47% less messages than Supple. Additionally, under scenarios with message loss, ProFlex performs much better than Supple, however under scenarios with node failures their performance are comparable.

We also proposed an improvement to ProFlex to leverage the inherent data correlation of WSN applications. Such an improvement was capable of reducing ProFlex's overhead by about 21% and maintain the data gathering efficiency of the original version. When compared with existing protocols, a version of ProFlex with the proposed improvement was capable of achieving similar data gathering efficiency as existing protocols with about 43% less messages' sending. Table 2.6 shows a summary of the main characteristics of ProFlex and related protocols studied in this chapter. As we can notice, ProFlex is the only solution that is able to leverage a heterogeneous network topology and data redundancy and correlation inherent to WSNs. This partially explains the improved performance of ProFlex when compared to related solutions.

So far, we investigated how data dissemination may be used to construct a data replication and distributed data storage mechanism for WSNs. As such, we focused on proposing a simple and computing inexpensive solution for a static network in which

Table 2.6: Comparison of protocols

| Protocol | Dissemination Strategy | Dissemination Procedure | Network Type | Exploits Redundancy |
|----------|------------------------|-------------------------|---------------|---------------------|
| ProFlex | Multiple | Tree-based | Heterogeneous | Yes |
| Supple | Multiple | Tree-based | Homogeneous | No |
| Deep | Uniform-only | Probabilistic flooding | Homogeneous | No |
| RaWMS | Uniform-only | Random walk | Homogeneous | No |

the nodes are likely to fail. Hereafter, we investigate how data dissemination may be used as a data communication procedure to notify drivers inside an area of interest about an event, which may be urgent or not, in VANETs. Therefore, we now focus on approaches that can tackle the topological dynamics inherent to these networks, while trying to guarantee a good delivery, low overhead and low delay.

Despite the apparent disparity between the content of this chapter and the remaining of this thesis, it is worth noticing that ProFlex is an interesting approach for the development of a distributed Traffic Information System (TIS) for VANETs [Lochert et al., 2008; Panichpapiboon and Pattara-Atikom, 2008; Sommer et al., 2011b]. For instance, to notify drivers about road traffic information in a city. In this scheme, roadside units (RSUs) would make the role of static sensor nodes (*L-sensors* and *H-sensors*), vehicles would make the role of mobile sinks and the sensed data would be the perceived road traffic by RSUs. Therefore, as a driver moves around in a city, he would have a complete view of the road traffic on his route by visiting and collecting the sensed data from a small number of RSUs. Furthermore, the communication technology employed in VANETs, i.e., IEEE 802.11p [IEEE, 2010], offers all elements required by ProFlex. For instance, the possibility to use different and non-interfering channels and the availability of long-range radios. Therefore, the solution proposed in this chapter is just an alternative approach to accomplish a task that the solutions described hereafter may also be employed.

Chapter 3

Data Dissemination in Highway Vehicular Ad hoc Networks with Extreme Traffic Conditions

Data dissemination to vehicles inside a region of interest is a fundamental service for many envisioned applications in VANETs. Despite being broadly studied, most existing solutions dealing with dissemination in such networks solely focus on either connected or intermittently connected topologies. However, the topologies on these networks can change dramatically depending on the geographical position or the time of day. Therefore, protocols capable of adapting themselves to the current road traffic condition is paramount. With this in mind, in this chapter we propose HyDi, a directional data dissemination protocol suited for highway scenarios. HyDi can seamlessly operate on dense networks by avoiding contention at the link layer, and also on intermittently connected networks by delivering messages even when there is no end-to-end communication path between the source and intended recipients. For that, we propose a new broadcast suppression technique that combines sender-based and receiver-based approaches to avoid excessive redundant retransmissions. Moreover, a new store-carry-forward mechanism that uses special vehicles in both moving directions is also presented. By means of simulation, we compare HyDi to two established solutions and we show that HyDi has a lower overhead, outperforms both protocols when considering the average delay to disseminate messages under heavy traffic scenarios and can deliver data to almost all intended recipients.

3.1 Introduction

Vehicular Ad hoc Networks (VANETs) have gained a lot of attention from the research and automotive communities in the last few years due to their potential in providing accident-free and intelligent transportation systems [Hartenstein and Laberteaux, 2008; Lee and Gerla, 2010; Marfia et al., 2013]. In these networks, vehicles are equipped with powerful processing units and wireless networking interfaces for communicating with passing vehicles and nearby roadside units (RSU), such as traffic signals and pavement sensors [Bai and Krishnamachari, 2010; Knorr et al., 2012]. For instance, in a collision-avoidance application, upon the detection of a dangerous situation, such as a sudden and hard brake, vehicles produce and disseminate warning messages in a time-critical fashion to approaching vehicles (see Figure 3.1). Therefore, drivers can become aware of it and act accordingly. Usually, the dissemination task performed by VANET applications is accomplished by means of uncoordinated broadcast messages, in which vehicles, after receiving a message, blindly rebroadcast it to further vehicles [Lu and Poellabauer, 2011]. Such process is commonly referred as Flooding.

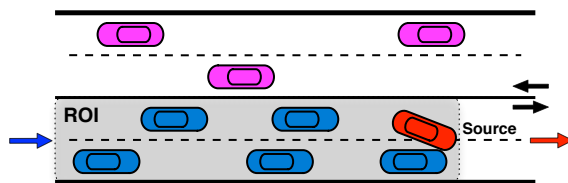


Figure 3.1: After a collision, a source vehicle produces and disseminates a warning to vehicles approaching the accident (vehicles moving to the east $[-\rightarrow]$ direction). The warning must be disseminated to all vehicles inside the region of interest (ROI) defined by the collision-avoidance application

The design of VANET dissemination protocols that rely on broadcast messages must consider two common problems. The first one, known as *the broadcast storm problem*, occurs in dense networks when multiple vehicles that are inside the communication range of one another blindly attempt to transmit at the same time. This may lead to severe contention at the link layer, packet collisions, inefficient use of bandwidth, caused by the high number of generated redundant messages, and probably service disruption [Ni et al., 1999; Williams and Camp, 2002]. One widely adopted approach to minimize or even avoid these undesired effects is to use broadcast suppression techniques [Wisitpongphan et al., 2007], i.e., mechanisms that select which nodes should rebroadcast and when the rebroadcast should be performed.

On the other extreme, there is a problem known as the *intermittently connected network problem*, which occurs in sparse networks when the number of vehicles in a given area is not sufficient to disseminate the message further. In such a scenario, if a vehicle is not aware that the network is disconnected, when it receives a message it will simply rebroadcast and then discard it, i.e., send the message to the application layer. Since there is no other vehicle inside the communication range to receive the message, it is simply lost forever, thus, resulting in a poor delivery and leaving some drivers unaware of the reported event. Therefore, one must rely on the *store-carry-forward* communication model, which takes advantage of the mobility of vehicles to opportunistically carry and transfer messages to vehicles that are geographically apart from one another [Jain et al., 2004; Spyropoulos et al., 2008].

It is worth noticing that both problems have been widely studied, especially by the mobile ad hoc network community (MANET) [Jain et al., 2004; Ni et al., 1999; Spyropoulos et al., 2008; Tseng et al., 2003; Williams and Camp, 2002]. Nevertheless, the existing solutions were developed either to cope with the broadcast storm or the intermittently connected network problem. We argue, though, that both problems should not be treated separately, since situations with both dense and sparse road traffic are very likely to coexist, especially when considering different time of the day and geographical positions in a highway or a city [Bai and Krishnamachari, 2009; Upoor and Fiore, 2012]. Moreover, to the best of our knowledge, we are aware of only two solutions in the VANET context that tackle both types of road traffic under highway scenarios [Tonguz et al., 2010; Schwartz et al., 2011], which motivated us to further investigate this problem. Therefore, we consider the problem of broadcast data dissemination in a specific direction of a highway that considers both scenarios mentioned above in a timely, efficient, and reliable manner using vehicle-to-vehicle communications.

In this chapter, we present HyDi, a data dissemination protocol for highway scenarios. We outline a new broadcast suppression mechanism that combines sender-based and receiver-based approaches to tackle the broadcast storm problem under high-traffic scenarios. In a sender-based approach, the transmitter chooses a priori the next vehicle to rebroadcast, whereas in a receiver-based approach the forwarding decisions are taken by receivers after processing the received messages, i.e., this is a posteriori decision. In summary, in our proposed sender-based mechanism, the transmitter always chooses as the next forwarder node some vehicle moving with its same direction, because their relative speed will be lower than choosing a vehicle moving in the opposite direction. Assuming that VANETs are highly dynamic networks, this increases the reliability on the choice performed by the transmitter. Moreover, in the receiver-based approach,

those vehicles that are intended recipients for a disseminated message possess a higher priority to rebroadcast. This decreases the number of vehicles contending to access the channel. Finally, we propose a completely new store-carry-forward scheme that selects special vehicles to buffer received messages as soon as network disconnections are perceived. These vehicles then forward these messages to neighbors capable of resuming the dissemination process. This enhances the delivery capability for our solution under intermittently connected networks.

By means of simulations we compare HyDi to two established solutions – SRD [Schwartz et al., 2011] and DV-CAST [Tonguz et al., 2010] – and we show that HyDi has a lower overhead when compared to these solutions, it has a delivery ratio close to 100% in almost all evaluated scenarios and outperforms both protocols when considering the average delay under heavy traffic scenarios. This is an indication that HyDi is a suitable approach for emergency warning applications, since it can deliver data to almost all vehicles in a given region of interest in a timely fashion way even under different traffic regimes.

3.2 Related Work

The broadcast storm and the intermittently connected network problems are topics that were already broadly investigated by the research community, though, separately from one another. In the MANET context, Ni et al. [1999] propose several threshold-based broadcast suppression techniques, such as distance-based, counter-based and geographic-based schemes. By using any of them, when a node receives a broadcast, it compares its local information to a predetermined threshold value. For instance, if the number of duplicate messages received is below the threshold or the relative distance to the sender is above the threshold, then the node decides to rebroadcast. Tseng et al. [2003] propose several techniques to dynamically adjust these threshold values according to the perceived local topology of a node. Despite some similarities, VANETs are intrinsically different from MANETs. For instance, in VANETs, the vehicles' movements are restricted by the roads and they can reach high speeds, whereas in MANETs, nodes are free to move in a certain area with much lower speeds. Therefore, solutions designed for MANETs cannot be directly applied to VANETs [Wisitpongphan et al., 2007].

In the VANET context, Korkmaz et al. [2004] propose the Urban Multi-hop Broadcast protocol (UMB), a broadcast suppression solution for urban scenarios that chooses the farthest vehicle from the transmitter to rebroadcast. Our proposed protocol

follows a similar approach, but with two substantial differences. First, rather than modifying the link layer, as in [Korkmaz et al., 2004], HyDi operates at the network layer and has as main goal to diminish the load submitted from the network layer to the link layer. We take such a decision based on the fact that VANET systems are confined to use the link and physical layers defined in the new WAVE standard [IEEE, 2010]. Therefore, we tried not to touch at the layers defined by the standard in order to make our solution compatible with future VANET systems deployments. Second, rather than using only a sender-based approach as in UMB, HyDi employs a hybrid solution based on both sender-based and receiver-based techniques in order to increase its robustness.

Wisitpongphan et al. [2007] propose three probabilistic and distance-based broadcast suppression techniques suited for highway scenarios. These are receiver-based approaches that do not rely on any neighbor information. When a vehicle receives a broadcast, it first calculates its distance to the transmitter. Then, using the computed distance and depending on the technique employed, the vehicle determines a delay or a probability to rebroadcast. According to the results presented in [Wisitpongphan et al., 2007], distance-based solutions reduce broadcast redundancy and message loss by up to 70%. Besides a sender-based broadcast suppression mechanism, here we also propose a receiver-based approach that uses the distance between nodes to determine whether and when a node should rebroadcast. This last scheme is used in scenarios in which the one-hop information is not timely available or the sender-based mechanism fails to disseminate the message further.

Concerning sparsely connected networks, data mules have been proposed to act as messengers carriers that collect data messages from one point, carry them for some time and then deliver them to another point in the network [Shah et al., 2003]. In Epidemic routing [Vahdat and Becker, 2000], nodes move through the network and as soon as new neighbors are discovered, they exchange data messages according to a pre-defined probability. In VIP delegation work [Barbera et al., 2011], data is propagated through a few previously selected socially important users in a mobile and intermittently connected network. Nevertheless, multi-hop opportunistic propagation is not considered. In a pioneering work, Chen et al. [2001] show the feasibility of using store-carry-forward to disseminate data messages in an intermittently connected VANET over a highway. The authors propose to use vehicles moving in the opposite direction to help in the dissemination process. We rely on some ideas presented in [Chen et al., 2001] to propose our store-carry-forward mechanism. Finally, while our work focuses only on highway scenarios, some recent studies have presented protocols that operate in densely and sparsely connected networks in urban scenarios [Korkmaz et al., 2006;

Tonguz et al., 2009; Viriyasitavat et al., 2010]. These approaches try to disseminate messages in different road directions when passing at an intersection and rely on either a fixed infrastructure or GPS and map information to determine who should disseminate the message.

To the best of our knowledge, there are only two solutions in the VANET literature that tackle the problem of data dissemination in highways under both dense and sparse networks. Tonguz et al. [2010] propose the Distributed Vehicular Broadcast protocol (DV-CAST) that combines two strategies: the distance-based *slotted 1-persistence* suppression technique [Wisitpongphan et al., 2007] and a store-carry-forward approach [Chen et al., 2001]. The results presented in [Tonguz et al., 2010] show that DV-CAST performs well in both extreme situations, but as shown later in our analysis and in [Schwartz et al., 2011], DV-CAST has a poor performance regarding the delivery ratio, overhead and delay. This behavior can probably be attributed to unforeseen situations, such as overtaking and vehicles leaving the highway by taking an exit, which were not considered by the authors in [Tonguz et al., 2010] and also because the broadcast suppression and store-carry-forward techniques employed by the protocol are not effective in avoiding the broadcast storm and intermittently connected network problems.

Schwartz et al. [2011] propose the Simple and Robust Dissemination protocol (SRD). The first improvement of SRD when compared to DV-CAST is a slightly modification to the *slotted 1-persistence* suppression technique employed in DV-CAST. First, in the traditional *slotted 1-persistence*, vehicles are assigned a time-slot to rebroadcast. However, a time-slot can be assigned to more than one vehicle, which may result in contention at the link layer and consequently packet collisions. Therefore, in SRD, each time-slot is further subdivided into micro-slots, which decreases the chance of more than one vehicle being assigned to the same time-slot. Schwartz et al. [2011] define this scheme as *micro-slotted 1-persistence*. Moreover, SRD gives a higher forwarding priority to vehicles that are following the source vehicle, while vehicles that are moving in the opposite direction are given lower priorities to rebroadcast. This last idea proved so efficient that we rely on a similar approach in the receiver-based broadcast suppression technique used in HyDi. Moreover, SRD uses many vehicles to act as data mules during the store-carry-forward process, contrarily to DV-CAST, which employs a single vehicle in each direction to perform such a task. Although the results presented in [Schwartz et al., 2011] show that SRD outperforms DV-CAST in the evaluated scenarios, we argue that SRD makes certain assumptions that might not be realistic. For instance, the authors assume all vehicles in the network have the same transmission range and adopt a predefined direction for message dissemination. Despite these two

existing solutions, we argue that this topic deserves a further investigation.

3.3 Proposed Protocol

In this section, we describe HyDi, a protocol designed to perform directional data dissemination in an efficient and effective way. Depending on the road traffic condition perceived, HyDi is able to adapt in order to guarantee a low overhead of messages transmitted, low delivery delay and a high delivery ratio. In this chapter, we focus on data dissemination under highway scenarios with traffic flowing in two opposite directions. The main application we envision for the use of HyDi is emergency warning, though any other application that relies on broadcast data dissemination may benefit from the protocol. In such an application, a source vehicle, upon the detection of a dangerous situation, produces a warning message and disseminates it to following vehicles. By means of a multi-hop vehicle-to-vehicle communication model, the warning message must reach all intended recipients as defined by the application, so drivers can become aware of the dangerous condition and act accordingly. Notice that we assume the message will be relevant only to vehicles moving in the same direction as the source vehicle and not to vehicles moving in the opposite direction. Nevertheless, vehicles moving in the opposite direction are still used by HyDi to enhance dissemination process. Finally, we assume the network is completely ad hoc, hence RSUs and repeaters are not available and only vehicles disseminate messages.

Designed to operate under extreme road traffic conditions, in HyDi, when a vehicle receives a data message, the decision of whether it should enter the broadcast suppression state or the store-carry-forward state is based only on its local topology, i.e., the vehicles in its vicinity. Therefore, we assume all vehicles are equipped with a GPS and they periodically broadcast beacons with their `<latitude, longitude, heading>`. Using this information, vehicles become aware of the position and direction of neighboring vehicles. It is worth noticing that the frequency with which vehicles disseminate beacons is a design parameter. A high frequency leads to a high accuracy on the one-hop neighborhood information of a vehicle, but may also lead to many beacons collisions. Furthermore, periodic beacons are also required by many other VANET applications, for instance, blind-spot detection. Thus, the local one-hop neighborhood connectivity is already a given piece of information.

The decisions made by the protocol are based on four pieces of information, which are easily obtained from the beacons received or are specified by the application at hand. These information are:

- **Message direction (MD):** assuming the highway is a straight multi-lane road in which vehicles move in both opposite roadways, the message direction is defined by the application. Without loss of generality, in this work we use the easterly (\rightarrow) and westerly (\leftarrow) directions. For instance, in an emergency warning application, when a source vehicle generates the warning, it is relevant only to vehicles following the source vehicle and not to the ones moving in the opposite direction. In Figure 3.2, the source vehicle produces a warning with MD = westerly.
- **Intended Recipient (IR):** it tells whether a vehicle is an intended recipient for the disseminated message. Then, IR = *true* for a given vehicle if its direction is the same as the source vehicle. Moreover, applications must specify a region of interest (ROI) or a time-to-live (TTL) for the generated messages. This way, if a vehicle moving in the same direction as the source vehicle receives a message, but it is outside the region of interest or the time-to-live has expired, then it immediately discard the message. For instance, in Figure 3.2, vehicles A and B have IR = *true*, since they are moving in the same direction as the source vehicle and they are also inside the ROI specified by the application. Although vehicle D is moving in the same direction as the source vehicle, it is outside the ROI, which was defined as 1 km by the application.
- **Message direction connectivity (MDC):** it depends on the MD and informs whether the vehicle has a next-hop neighbor with its same direction that can rebroadcast the message further (*true*), or whether a vehicle is the last one in a group of connected vehicles moving in the same direction (*false*). A vehicle with MDC = *true* should enter the broadcast suppression state, whereas a vehicle with MDC = *false* should enter the store-carry-forward state. For instance, in Figure 3.2, vehicle A has MDC = *true* since it is connected to vehicle B, which can continue with the dissemination process. Vehicle B has MDC = *false*, since it is not connected to any other vehicle that is moving in its same direction and that is also farther from the message origin. Vehicle C also has MDC = *false*, since it is not connected to any other vehicle that is moving in its same direction and that is also capable of disseminating the message further.
- **Opposite direction connectivity (ODC):** it indicates whether a vehicle has at least one neighbor moving in the opposite direction. For instance, if a vehicle is moving to the easterly direction and is connected to at least one vehicle moving to the westerly direction, then ODC = *true*, otherwise, ODC = *false*. In Figure 3.2,

vehicle C has $ODC = true$, since it is connected to vehicles A and B, while vehicle D has $ODC = false$ since it is not connected to any vehicle moving in the opposite direction.

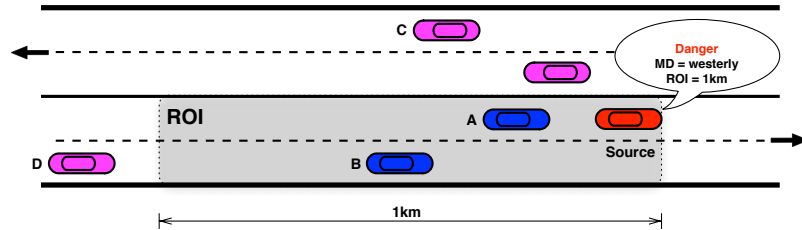


Figure 3.2: In this example, the source vehicle produces a warning message to be disseminated to following vehicles. Hence, the $MD = westerly$ and the intended recipients are vehicles A and B. Notice that, despite vehicle D is moving in the same direction as the source vehicle, it is outside the region of interest (ROI) defined by the application. Here, vehicle A has $\langle IR = true, MDC = true, ODC = true \rangle$, vehicle B has $\langle IR = true, MDC = false, ODC = true \rangle$, vehicle C has $\langle IR = false, MDC = false, ODC = true \rangle$ and vehicle D has $\langle IR = false, MDC = false, ODC = false \rangle$

By considering these four pieces of information, a vehicle using HyDi is able to determine in which of the two extreme cases it should operate. Basically, all vehicles with $MDC = true$ should enter a broadcast suppression regime, since for our purposes they are operating under a connected network topology. It is worth noticing that a vehicle may assume it is under a connected network even if the network is not completely connected. For instance, in Figure 3.2, vehicle A assumes it is under a connected network topology, which is not the case since vehicle D is disconnected from the rest of the network. Therefore, when we say that a vehicle with $MDC = true$ is under a connected network topology, we are referring to the local topology perceived by the vehicle and not the global topology of the network. The exactly behavior of a vehicle under this regime is described hereafter.

3.3.1 Broadcast Suppression

When a vehicle receives a message and perceives it is operating under a connected network, i.e., $MDC = true$, it must employ some kind of suppression mechanism to coordinate the rebroadcast of the message and consequently avoid redundant re-transmissions and high load on the channel. Several sender-based and receiver-based suppression mechanisms have been proposed in the literature [Ni et al., 1999; Tseng et al., 2003; Williams and Camp, 2002], some of them especially suited for VANETs

on highways [Wisitpongphan et al., 2007]. Nevertheless, in this chapter we propose a hybrid approach that relies on both sender-based and receiver-based mechanisms to increase the efficiency, efficacy and reliability of the protocol. Notice that, in a sender-based approach, the vehicle chooses a priori the next vehicle that should rebroadcast, whereas in a receiver-based approach the forwarding decision is taken by the receivers after processing the received message. In HyDi, the general idea is to employ first the sender-based approach and use the receiver-based part only when the first fails to propagate the message further.

At first, immediately before broadcasting a message, the vehicle selects the best vehicle to rebroadcast it further (sender-based). Algorithm 5 shows how this procedure works. Here, the adopted approach is to select the farthest neighbor with the same moving direction (lines 4 and 5). The idea for choosing vehicles with the same moving direction is that their relative speeds are generally smaller when compared to vehicles moving in opposite directions. Therefore, this increases the chance of choosing a vehicle as the next forwarder and it actually receiving the message. For instance, in Figure 3.3, assuming the MD as the westerly direction, immediately before vehicle *A* broadcasts a message, it chooses vehicle *C* as the next forwarder, since it is the farthest one-hop neighbor and they are both moving in the same direction. Then, *A* sets *C* as the next forwarder and broadcasts the message. When *C* receives it, it finds its address as the next forwarder, selects its own best forwarder and immediately rebroadcasts the message. This procedure continues until the message reaches the limit defined by ROI or the TTL expires. It is straightforward to notice that this procedure avoids a huge amount of unnecessary retransmissions, for instance, the retransmissions of vehicles *B*, *D* and *E* in Figure 3.3.

Algorithm 5: Choosing the next forwarder of vehicle *i*

input : The list of one-hop neighbors *N* of vehicle *i*
input : The message *msg* received by vehicle *i*
output: The address of the next forwarder

- 1 *greatestDistance* \leftarrow distance(*i.position*, *msg.originPosition*)
- 2 *nextForwarderAddress* \leftarrow broadcastAddress
- 3 **foreach** *n* \in *N* **do**
- 4 **if** distance(*n*, *msg.originPosition*) > *greatestDistance*
- 5 **and** *n.direction* = *i.direction* **then**
- 6 *greatestDistance* \leftarrow distance(*n.position*, *msg.originPosition*)
- 7 *nextForwarderAddress* \leftarrow *n.address*

8 **return** *nextForwarderAddress*

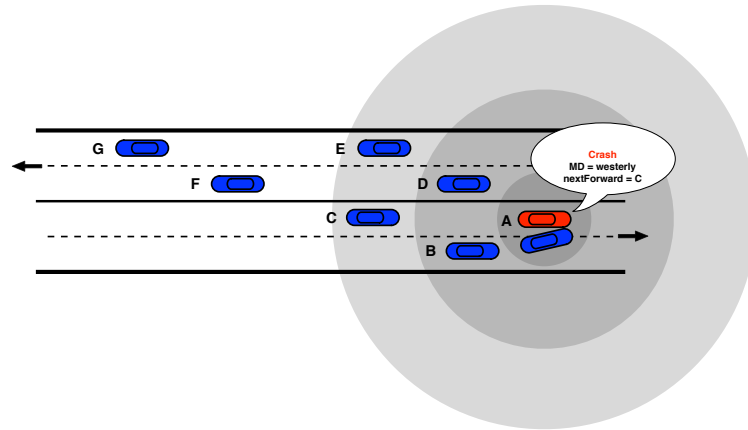


Figure 3.3: Broadcast suppression mechanism. Using the sender-based part of the algorithm, the transmitter chooses a priori the next forwarder, i.e., vehicle C. If this procedure fails, the receiver-based part guarantees that some other vehicle will rebroadcast, e.g., vehicles B, D and E

However, notice that a vehicle may not find a next forwarder vehicle. For instance, in Figure 3.3, when vehicle *C* receives the broadcast from *A* and looks for its own next forwarder, it does not find any, since there is no vehicle in its one-hop neighborhood that is farther than itself and is also moving in its same direction. In such a scenario, *C* simply rebroadcasts the message with the broadcast address in the field of the next forwarder. When a vehicle receives a message with the broadcast address or the address of another vehicle in the field of the next forwarder, it executes the receiver-based broadcast suppression mechanism part of the protocol, described hereafter.

Now, let's suppose that *C* does not receive the message from *A*. In the case of a sender-based approach only, the message is not rebroadcasted by any of the one-hop neighbors of *A* (e.g., *B*, *D* and *E*), since it has *C* as the responsible for rebroadcasting it. Hence, this would interrupt the dissemination process, resulting in a poor performance. Therefore, to overcome such situation, we combine the previously described sender-based approach with a distance-based broadcast suppression mechanism, which is our receiver-based strategy. Algorithm 6 shows all steps of this hybrid solution. At first, when a vehicle *j* receives a message from a vehicle *i*, it verifies whether it is a duplicate or not (Line 1). If it is not a duplicate, then *j* checks whether it has a neighbor able to continue the dissemination ($MDC = true$)(Line 2) or not ($MDC = false$). Assuming it has, *j* will look for its own address in the field of the next forwarder (Line 3). If *j* finds its address as the next forwarder, it will choose its own next forwarder, according to Algorithm 5, insert his own address as the sender of the message and immediately rebroadcast it (lines 4-6).

Algorithm 6: Broadcast suppression mechanism

```

input : The list of one-hop neighbors  $N$  of vehicle  $i$ 
input : The message  $msg$  received by vehicle  $i$ 

1 if  $msg$  is not a duplicate then
2   if  $MDC = true$  then
3     if  $msg.nextForwarder = i.address$  then
4        $msg.nextForwarder \leftarrow getNextForwarder(N, msg)$ 
5        $msg.sender \leftarrow i$ 
6        $broadcast(msg)$ 
7     else
8        $D \leftarrow distance(i.position, msg.sender.position)$ 
9        $percentageDistance \leftarrow \frac{\min(D, communicationRange)}{communicationRange}$  // (0, 1]
10      if  $IR = true$  then // High priority
11         $S \leftarrow \lfloor N_s \times (1 - percentageDistance) \rfloor$  // [0,  $N_s - 1$ ]
12      else // Low priority
13         $S \leftarrow \lfloor N_s \times (2 - percentageDistance) \rfloor$  // [ $N_s$ ,  $2N_s - 1$ ]
14       $T \leftarrow (S \times \tau) + \tau$ 
15       $scheduleBroadcast(msg, T)$ 
16    else
17      // Store-carry-forward (see Algorithm 7)
18  else
19    if  $isBroadcastScheduled(msg)$ 
20    and  $isSenderFartherAway(i.position, msg.sender.position)$  then
21       $cancelBroadcast(msg)$ 
22    // Store-carry-forward (see Algorithm 7)

```

However, if j does not find its own address in the field of the next forwarder, it will execute the receiver-based part of the protocol (lines 7-15). The idea is that vehicle j schedules and waits T seconds to rebroadcast the message at time slot S . To calculate the time slot S , j first determines the distance D to i (Line 8). Then, it calculates the percentage distance by dividing D by the communication range, which produces a number in the interval $(0, 1]$ (Line 9). Notice that, to get a number in the interval $(0, 1]$, we use the minimum value between the distance and the communication range, since the communication range is just an estimation. Therefore, the distance between the vehicles may be bigger than the estimated communication range. After calculating the percentage distance, j verifies whether it is an intended recipient ($IR = true$) or not ($IR = false$) (Line 10). Here, intended recipients have a higher priority to rebroadcast.

If j is an intended recipient, it calculates the time slot S according to the percentage distance and the number of available time slots for each priority (N_s), a predefined parameter to the protocol (Line 11). The result is a number in the interval $[0, N_s - 1]$. On the other hand, if j is not an intended recipient, its time slot S is a number in the interval $[N_s, 2N_s - 1]$. Finally, j calculates the delay T to rebroadcast according to its assigned time slot and a predefined delay τ , and schedule the rebroadcast (lines 14-15). Like N_s , the delay τ is also a parameter to the protocol. This delay must take into account the medium access delay and propagation delay. Notice the additional τ at Line 14. Such additional delay gives enough time for the vehicle chosen as the next forwarder (e.g., vehicle C in Figure 3.3) a chance to rebroadcast, thus suppressing the retransmission of all other vehicles that are under the receiver-based approach (lines 18-20), (e.g., vehicles B , D and E in Figure 3.3).

For instance, in Figure 3.3, the number of time slots for each priority is $N_s = 3$, represented by a circle with three shades of gray around vehicle A (in this example we assume the communication range is a perfect circle). When A transmits the message with vehicle C as the next forwarder and B receives it, then B enters into the receiver-based part of the protocol, where it is assigned a time slot in the interval $[0, 2]$, since B is an intended recipient. Indeed, the actual time slot assigned is $S = 1$. On the other hand, vehicles D and E are assigned a time slot in the interval $[3, 5]$, since they are not intended recipients. In this example, D is assigned the time slot $S = 4$ and E the time slot $S = 3$. Notice that, despite B being assigned an earlier time slot than E , when B transmits, it does not suppress the broadcast from E , since B is not father in the message direction than E , according to Line 19 of Algorithm 6. Otherwise, it may handicap the dissemination process, for instance, to vehicles F and G . Finally, if vehicle C receives the broadcast from A , since it is the next forwarder, it enters into the sender-based part of the protocol and immediately rebroadcasts, thus suppressing the unnecessary rebroadcasts from B , D and E .

3.3.2 Store-carry-forward

The other component of HyDi is its store-carry-forward mechanism, as presented in Algorithm 7. The idea is that when a vehicle receives a message and perceives that there is no other vehicle to continue the dissemination process, then it holds the message until a new connection is established with a vehicle able to restore the dissemination to its pre-established course, or until it leaves the ROI or the TTL expires, thus discarding the message. To accomplish this, HyDi relies on vehicles with $MDC = false$. The exact behavior of a vehicle under this regime differ depending on whether it is connected to

another vehicle moving in the opposite direction or not and also on whether the vehicle is an intended recipient.

Algorithm 7: Store-carry-forward mechanism

```

input : The list of one-hop neighbors  $N$  of vehicle  $i$ 
input : The message  $msg$  received by vehicle  $i$ 
1 if  $msg$  is not a duplicate then
2   if  $MDC = true$  then
3      $\lfloor$  // Broadcast suppression (see Algorithm 6)
4   else
5      $msg.sender \leftarrow i$ 
6      $msg.fromSCF \leftarrow true$ 
7     broadcast( $msg$ )
8     if  $ODC = false$  or  $IR = false$  then
9        $\lfloor$   $i.SCF \leftarrow true$ 
10  else
11    // Broadcast suppression cancellation (see Algorithm 6)
12    if  $msg.fromSCF = true$  then
13      if  $msg.sender.direction = i.direction$  then
14        if  $ODC = false$  or  $IR = false$  then
15           $\lfloor$   $i.SCF \leftarrow true$ 
16        else if  $MDC = false$  and  $ODC = true$  and  $IR = false$  then
17           $\lfloor$   $i.SCF \leftarrow true$ 

```

According to Algorithm 7, when vehicle i receives a message and perceives there is no other vehicle to resume the dissemination process ($MDC = false$), then it immediately broadcast the message with a flag $fromSCF$ set to true (lines 4-6). This suppresses the rebroadcast of vehicles that are under the receiver-based broadcast suppression, according to lines 18-20 of Algorithm 6. For instance, in Figure 3.4, when vehicles B and D receive the message and they perceive that there are no other neighbor to continue the dissemination, they immediately rebroadcast to suppress the rebroadcast from vehicles A and C . After rebroadcasting the message, i verifies whether it should actually enter the store-carry-forward state or not (Line 7). If i is not connected to a neighbor in the opposite direction ($ODC = false$) or it is not an intended recipient ($IR = false$), then it enter the store-carry-forward state (Line 8). For instance, in Figure 3.4, vehicle D goes to the store-carry-forward state, but B does not, since it is connected to a neighbor in the opposite direction and it is also an intended recipient. In this case, B

relies on vehicle D to store and carry the message until D meets a new neighbor that is able to resume the dissemination, e.g., vehicles E or F .

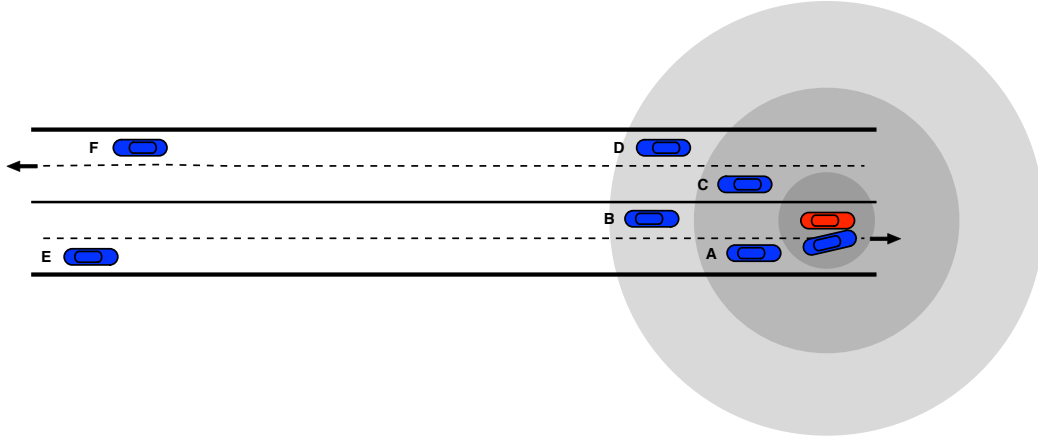


Figure 3.4: Store-carry-forward. HyDi uses the last vehicles in a group of connected vehicles to perform the store-carry-forward task, e.g., vehicles B and D. Moreover, it employs extra vehicles for this task to act as backup in case the first ones fail, e.g., vehicles A and C

As previously described, before entering the store-carry-forward state, vehicle i rebroadcasts the message with a flag `fromSCF` set to true. This way, i informs its neighbors, with its same moving direction, that it is about to enter a store-carry-forward state. Hence, i 's neighbors can decide whether they should also enter into a store-carry-forward state as backup vehicles, for instance, for the scenarios where i leaves the highway before resuming the dissemination process, or i is just overtaken by one of its faster neighbors. Such decision is taken at lines 10-15 of Algorithm 7. For instance, in Figure 3.4, when D broadcasts the message with the flag `fromSCF` set to true and C receives it, C decides to enter the store-carry-forward state. Therefore, even if D leaves the highway or C overtakes D , C still can meet a new neighbor to resume the dissemination.

Finally, the last part of the store-carry-forward mechanism is to restore the dissemination process, and the right opportunity for that is when a new neighbor is detected. Therefore, according to Algorithm 8, every time a vehicle receives a beacon, it checks whether it is from a new neighbor capable of resuming the dissemination. If it is, then it broadcasts the message with the flag `fromSCF` set to false and leaves the store-carry-forward state. Notice that, only vehicles with `MDC = false` broadcast the message when a new neighbor is detected. Therefore, in Figure 3.5, even if vehicles C and D are under the store-carry-forward state, when they meet a new neighbor able to restore the dissemination, e.g., vehicle E , only D will broadcast. Furthermore,

even after leaving the store-carry-forward state, a vehicle may still return to such state later on, according to lines 10-15 of Algorithm 7. Consider the example in Figure 3.5, where D meets E and the dissemination resumes just momentarily, because E does not have any neighbor to continue the dissemination ($MDC = false$). Therefore, E would broadcast the message with the flag `fromSCF` set to true, and D would receive it and go back to the store-carry-forward state (lines 14-15 of Algorithm 7). This scenario is not considered by any of the existing solutions.

Algorithm 8: Resuming the dissemination when a beacon is received

input : The list of one-hop neighbors N of vehicle i
input : The beacon b received by vehicle i

```

1 if  $MDC = false$  then
2   updateNeighborhoodStatus( $b, N$ )
3   if  $i.SCF = true$  and ( $ODC = true$  or  $MDC = true$ ) then
4      $msg.sender \leftarrow i$ 
5      $msg.fromSCF \leftarrow false$ 
6     broadcast( $msg$ )
7      $i.SCF \leftarrow false$ 

```

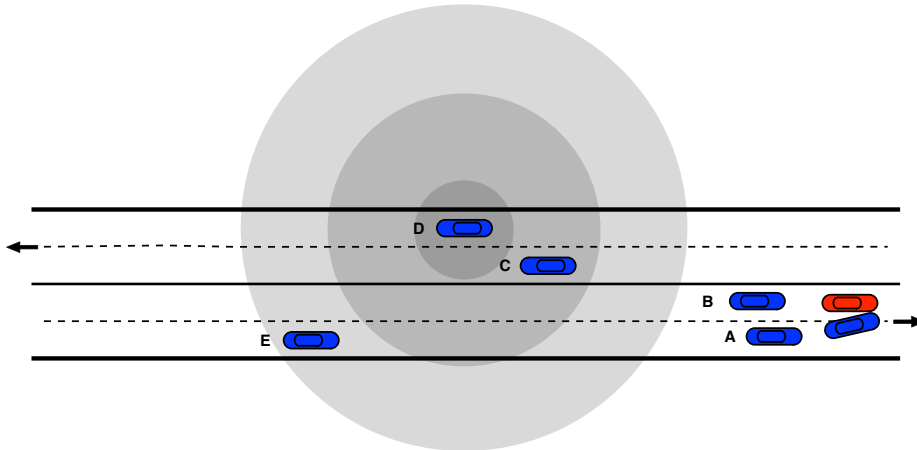


Figure 3.5: In HyDi, multiple vehicles store-carry-forward a message, for instance, vehicles C and D. However, when they meet a new neighbor able to resume the dissemination process, e.g., vehicle E, only the vehicle with $MDC = false$ will broadcast the message, in this example, vehicle D

3.4 Performance Analysis

In this section, we assess the performance of HyDi by means of simulation using the OMNeT++ 4.2.2 simulator¹ [Varga and Hornig, 2008]. We compare HyDi with two existing protocols that also focus on directional dissemination under both sparse and dense highways – SRD [Schwartz et al., 2011] and DV-CAST [Tonguz et al., 2010] – and with a traditional Flooding. We evaluate all protocols under low, normal and high traffic scenarios.

To increase the accuracy of our analysis, we use the Veins 2.1² [Sommer et al., 2011a] simulation framework, which implements the link and physical layers defined in the IEEE 1609.4 WAVE Multichannel Operation and IEEE 802.11p standards [IEEE, 2010]. In the MAC layer, we set the bit rate to 18 Mbit/s, and use the IEEE 1609.4 WAVE Control Channel (CCH) for beacon transmissions and the Service Channel (SCH) for data dissemination. Therefore, beacons do not interfere in data messages, i.e., beacons do not contend to access the medium at the same time as data messages. Moreover, we use the two-ray ground propagation model [Sommer et al., 2012] and we set the transmission power to 1.6 mW, thus achieving a communication range of approximately 250 m. These parameters were chosen according to envisioned deployments for VANETs [Hartenstein and Laberteaux, 2008; Marfia et al., 2013; Eckhoff et al., 2012]

For the data dissemination protocols, we set the beacon message size to 32 bytes and the data message size to 2048 bytes. These values are within the limits imposed by the WAVE standard [IEEE, 2010]. Beacon messages are generated with a frequency of 2 Hz. While real deployments will require a much higher beacon frequency, e.g., 10 Hz, a value of 2 Hz is enough for our purposes and it has been a common value elsewhere [Tonguz et al., 2010; Schwartz et al., 2011]. Based on a small subset of experiments, we set the number of time slots (N_s) used by our broadcast suppression technique to 6 and the delay (τ) to 60 ms (see Algorithm 6). For all presented results, each point in the graphs represents the mean of 50 replications with a confidence interval of 95%. Table 3.1 shows a summary of the main simulation parameters used in our performance analysis.

3.4.1 Evaluated Scenario

HyDi is a protocol designed for directional data dissemination, hence the scenario considered here consists of a straight four-lane highway of 4.5 km of extension with

¹<http://www.omnetpp.org>

²<http://veins.car2x.org>

Table 3.1: Simulation parameters.

| Parameter | Value |
|----------------------------------|------------|
| Transmission power | 1.6 mW |
| Transmission range (R) | 250 m |
| Bit rate | 18 Mbit/s |
| Number of time slots (N_s) | 6 |
| Delay (τ) | 60 ms |
| Highway length | 4.5 km |
| Beacon size | 32 bytes |
| Beacon frequency | 2 Hz |
| Data message size | 2048 bytes |
| Data message frequency | 1 Hz |
| Number of data messages produced | 100 |
| Data message ROI | 4 km |
| Data message TTL | 100 s |
| Number of runs | 50 |
| Confidence interval | 95% |

vehicles moving in both easterly and westerly directions (see Figure 3.6). To simulate different road traffic regimes, we use the SUMO 0.17.0³ [Behrisch et al., 2011] mobility simulator to build the highway and produce the vehicles' movements. There are two vehicle production flows, one at each opposite edge of the highway, which insert vehicles into the network at a constant rate. In order to ensure overtaking, both flows insert into the network two types of vehicles. The first type consists of vehicles capable of reaching a maximum speed of 120 km/h, while the second type consists of vehicles capable of reaching a maximum speed of 55 km/h. Therefore, this can be thought of as a network composed by passengers cars and heavy trucks. Notice the amount of vehicles for each type is equal throughout the simulation time.

Furthermore, the first flow consists of vehicles moving toward the easterly direction and the second one consists of vehicles moving toward the westerly direction. Each flow corresponds to vehicles entering the network at rates of 200, 400, 500, 600 and 700 vehicles/hour (low traffic scenarios); 800 and 1000 vehicles/hour (normal traffic scenarios); and finally 1200, 1400 and 1600 vehicles/hour (high traffic scenarios). Our goal is to assess how the considered protocols behave under intermittently connected (low traffic), connected (normal traffic) and highly connected networks (high traffic). A RSU positioned 250 m from the east edge of the highway produces 100 data messages at a frequency of 1 Hz. All of them have a ROI of 4 km and a TTL of 100 s. Here, data messages correspond to an emergency warning that is being disseminated

³<http://sumo.sourceforge.net>

to the westerly direction to notify drivers who are approaching the RSU. Therefore, the goal is to disseminate the warnings to the westerly direction in a timely fashion way, delivering them to as many vehicles moving to the easterly direction as possible by means of multi-hop communication. It is worth noticing the RSU is used solely for the purpose of generating the emergency warnings and it is not used to help in the dissemination process.

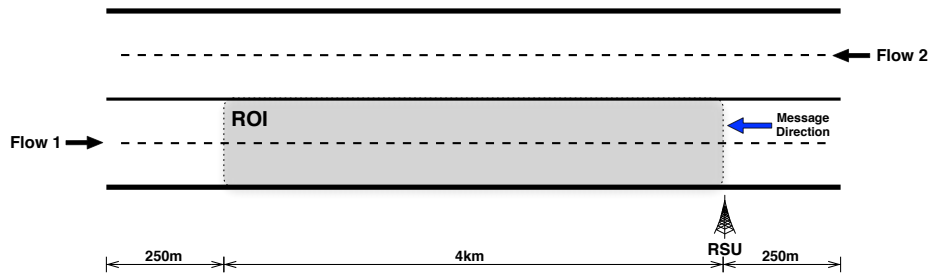


Figure 3.6: The base highway scenario considered in our performance analysis

3.4.2 Evaluated Metrics

Our proposed protocol has as main requirements the data dissemination in a reliable, scalable and efficient way. Therefore, we evaluate the following metrics to assess whether the protocol meets these design requirements:

- **Delivery ratio:** the percentage of data messages generated by the RSU that is actually delivered to intended recipients. It is expected that dissemination protocols must achieve a delivery ratio close to 100%.
- **Messages transmitted:** the total number of data messages transmitted by all vehicles during the dissemination process. A high number of message transmissions indicates that network bandwidth is being wasted due to a high number of duplicated messages, which may cause service disruption, especially under high traffic situations.
- **Delay:** the average time it takes for a data message to travel from the RSU to intended recipients. This metric is particularly relevant for time critical information that must be disseminated as quickly as possible.

3.4.3 Highway Results

A primary requirement for any dissemination protocol is a guarantee that it delivers data messages to all intended recipients. Figure 3.7 shows the delivery ratio for all evaluated protocols. As can be observed, HyDi is the solution that delivers more messages under all considered road traffic scenarios. Under normal and high traffic, in which the network is always connected, all protocols guarantee 100% delivery. On the other hand, under low traffic, where the store-carry-forward communication model prevails, the literature protocols do not perform as expected. For instance, the lowest delivery ratio for HyDi is about 90% for a traffic of 200 vehicles/hour, while for all other protocols it is below 50%. The lower delivery for DV-CAST under these scenarios can be attributed to the fact that this protocol uses just one vehicle in each road direction to perform the store-carry-forward task. Therefore, if these vehicles fail to deliver the message, there are no other vehicle to act as backup. Recall from Section 3.3.2 that HyDi uses many vehicles for this task. The lower delivery for SRD under low traffic can be attributed to short-lived connections and disconnections among vehicles. This handicaps the proper functioning of the mechanism used by SRD to determine whether a vehicle should perform the store-carry-forward task or not. Notice that, the lower delivery for Flooding under low traffic scenarios is due to the fact it does not perform store-carry-forward. Finally, as the traffic increases, the delivery ratio for all protocols improves, since use of the store-carry forward communication model becomes less frequent.

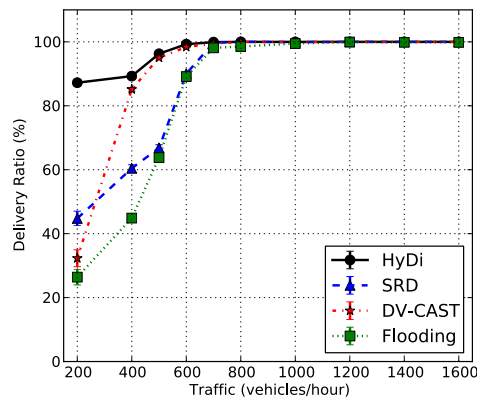


Figure 3.7: The delivery ratio for all road traffic scenarios. Notice that, HyDi has the best delivery ratio, especially under low traffic scenarios, where the store-carry-forward communication model prevails

Figure 3.8 shows the total number of data messages transmitted by all vehicles during the dissemination process. As can be observed, for low traffic scenarios, HyDi

has the highest overhead. This result is expected and can be explained by two facts. First, HyDi delivers more messages to intended recipients than the other protocols under those scenarios, hence it requires more transmissions. Second, HyDi employs many vehicles in the store-carry-forward task. Therefore, when those moving to the westerly direction encounter new neighbors moving to the easterly direction, they cannot know whether these new neighbors have already received the disseminated messages. Hence, they simply forward the messages to these new neighbors. This increases the reliability of the protocol, as previously shown, at the cost of an increase in overhead. Notice the broadcast storm problem is not an issue under those scenarios. When the network becomes connected, and the broadcast storm is an issue, HyDi transmits less messages than the other protocols. Such result shows that the broadcast suppression mechanism proposed here is effective in avoiding redundant retransmissions. It also shows that HyDi scales well with the increase in traffic, since the number of messages transmitted is not much affected, starting at 800 vehicles/hour.

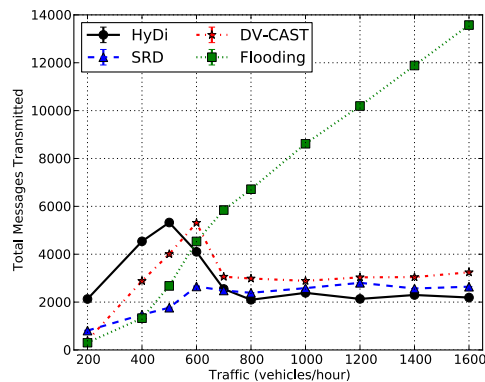


Figure 3.8: The total number of data messages transmitted by all vehicles during the dissemination process. HyDi transmits more messages under low traffic scenarios because it delivers more messages than the other protocols. Moreover, it employs many vehicles to the store-carry-forward task, which increases the reliability of the protocol at the cost of an increase in the overhead. As the traffic increases, HyDi transmits less messages than the other protocols

Figure 3.9 shows the average delay to disseminate data messages to intended recipients. As can be observed, HyDi has the highest delay for low traffic scenarios. Essentially, the explanation for such a fact is the same as to the previous result, i.e., HyDi delivers more messages than the other protocols. Notice that, the high delay for HyDi, SRD and DV-CAST under low traffic scenarios is due to the fact that vehicles need to store and carry the messages for longer distances in order to deliver them. Moreover, the low delay for DV-CAST at a traffic of 200 vehicles/hour and for Flooding

under low traffic scenarios in general is explained by the fact that these protocols deliver the messages only to vehicles that are close to the source, as can be inferred from the result shown in Figure 3.7. As the traffic increases and storing and carrying the messages becomes less common, the delay decreases. When the network is connected, HyDi's delay is only greater than Flooding's, as shown in the inset plot. This is due to the fact that, in HyDi, the sender-based broadcast suppression mechanism prevails most of the time, hence vehicles can rebroadcast the message immediately after receiving it.

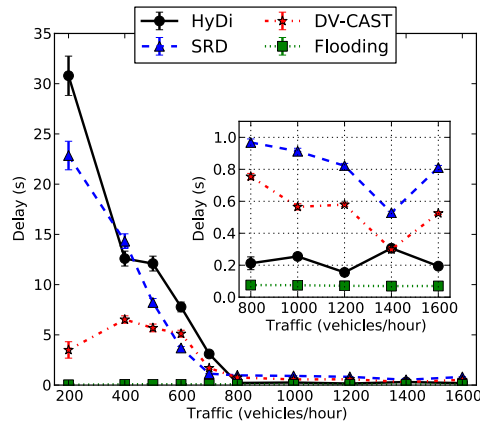


Figure 3.9: The average delay to deliver data messages to intended recipients. HyDi has a higher delay under low traffic scenarios because it delivers more messages to intended recipients. As the traffic increases, HyDi's delay is only higher than Flooding's

3.4.4 Highway with an Exit Results

The performance of dissemination protocols should not be disturbed if some vehicles leave the highway during the dissemination process. To assess the behavior of our solution under such circumstance, we introduce a small modification to the previous scenario. In this new scenario, vehicles moving to the westerly direction have the option of taking an exit located 2250 m from the east edge of the highway, i.e., in the middle of the highway. Indeed, half the vehicles take the exit, while the other half proceeds to the end of the highway. Vehicles that take the exit are removed from the network. Figure 3.10 shows the delivery ratio for this considered scenario. As we can see, the delivery ratio for HyDi is not affected by vehicles leaving the network before the end of the dissemination process, especially under low traffic scenarios. It continues to be the protocol that delivers more messages to intended recipients. The protocol that is most affected is DV-CAST. Recall that, this protocol employs only a single vehicle in each direction to the store-carry-forward task. Therefore, if one of the vehicles performing

such task takes the exit, then the delivery for this protocol is completely compromised, as evidenced by this result.

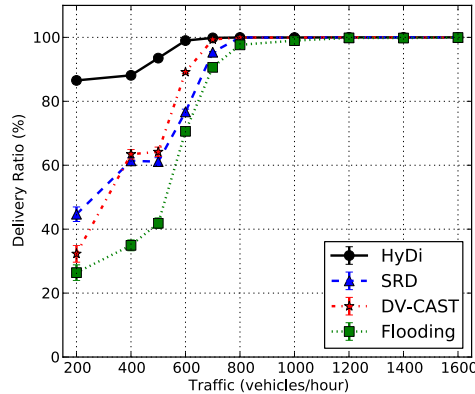


Figure 3.10: The delivery ratio for a highway with an exit. Notice that, the delivery ratio for HyDi is not affected by vehicles leaving the network

Figure 3.11 shows the total number of data messages transmitted. As we can see, for low traffic scenarios, as the traffic increases, so does the number of messages transmitted by HyDi, with a peak at 800 vehicles/hour. This happens because as vehicles take the exit, the remaining ones in the store-carry-forward state become the last vehicle in a group of connected vehicles ($MDC = false$). Therefore, they will immediately forward the messages when they encounter new neighbors (see Algorithm 8). Moreover, recall that HyDi delivers much more messages than the other protocols under low traffic. Notice that, as the network becomes connected, HyDi transmits less messages than the other protocols.

Figure 3.12 shows the average delay to deliver data messages to intended recipients. As in the scenario with no exit, HyDi is the protocol with the highest delay under low traffic scenarios. Since HyDi delivers more messages under these scenarios, it means that vehicles must store and carry the messages for longer distances to reach more intended recipients, hence justifying the higher delay. At a traffic of 1200 vehicles/hour, HyDi's delay behaves as in the scenario of a highway without an exit. In summary, we conclude that HyDi's performance is not much affected by vehicles leaving the network during the dissemination process. For instance, it guarantees an acceptable message delivery even under intermittently connected networks and avoids the broadcast storm under high traffic densities.

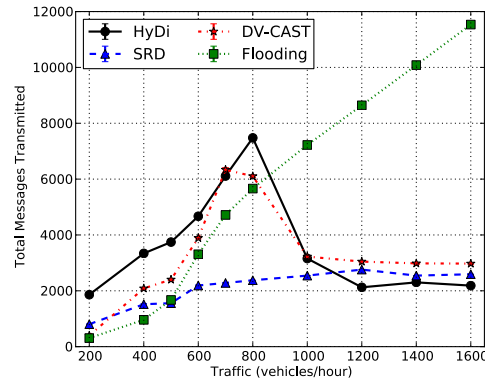


Figure 3.11: The total number of data messages transmitted for a highway with an exit. Under low traffic, HyDi is the protocol with the highest overhead. However, as the network becomes connected, HyDi transmits less messages than related protocols

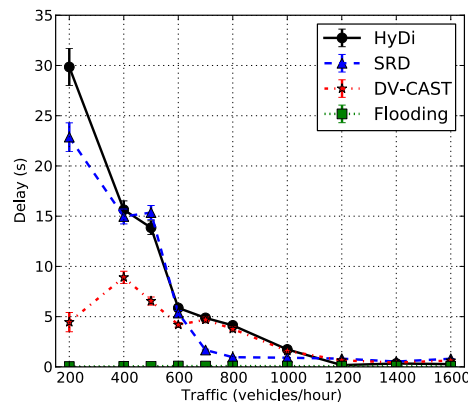


Figure 3.12: The average delay to deliver data messages to intended recipients in a highway with an exit. HyDi has a higher delay under low traffic scenarios because it delivers more messages to intended recipients. As the traffic increases, HyDi's delay is only higher than Flooding's. In general, HyDi's delay is not much affected by the fact that vehicles are leaving the network during the dissemination process

3.4.5 GPS Drift Results

For the effective functioning of our proposed protocol, it is fundamental the availability of accurate information about the neighborhood of a vehicle. Therefore, to show the resilience of the protocol to outdated or wrong information, we simulated HyDi in a highway without exit, but now with the presence of GPS drift. For that, when a vehicle broadcasts its position to its neighbors through periodic beacons, we add an error, chosen uniformly in the interval $[0, \text{MAX_ERROR}]$, to the reported position. We assessed HyDi for values of MAX_ERROR of 10 m, 30 m and 50 m.

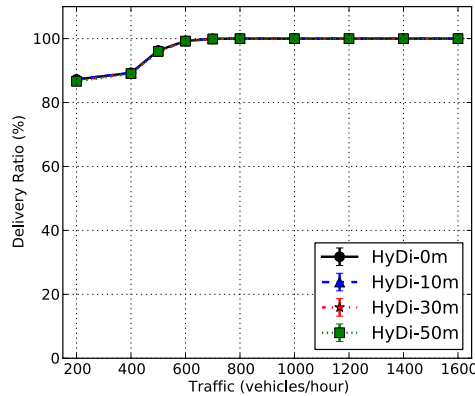


Figure 3.13: The delivery ratio for HyDi in the presence of GPS errors. Essentially, HyDi’s delivery is not affected by wrong reported positions

Figure 3.13 shows the delivery ratio. As can be observed, HyDi is slightly affected only for the scenarios with 200 and 400 vehicles/hour when the GPS error is at most 50 m. This is expected since it is under the store-carry-forward mechanism that accurate information is required to determine whether a node should store and carry a message or not. For normal and high traffic scenarios, the part of the protocol that suffer the most with the lack of accurate information is the sender-based broadcast suppression mechanism. However, as described in Section 3.3.1, when this mechanism fails, there still is the receiver-based part to keep up with the dissemination process.

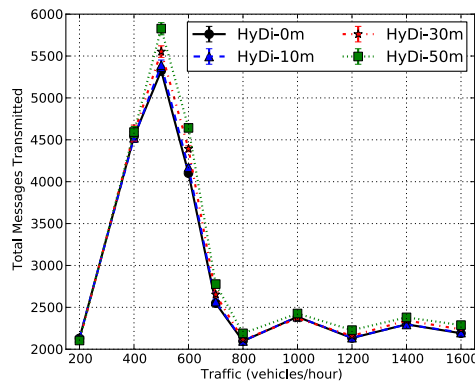


Figure 3.14: The total number of data messages transmitted for HyDi in the presence of GPS errors. Its is possible to notice a increase at traffics of 500 and 600 vehicles/hour

Figure 3.14 shows the total number of data messages transmitted. Once again, it is under low traffic scenarios that HyDi suffers the most, in particular at 500 and 600 vehicles/hour. Recall from Section 3.3.2 that in HyDi many vehicles perform the store-carry-forward task. However, only vehicles with $MDC = false$ actually transmit

the message to uniformed vehicles. Therefore, due to the wrong reported positions, many vehicles believe they are the last vehicle in a group of connected vehicles ($MDC = false$), thus transmitting the message indiscriminately (see Algorithm 8). Under normal and high traffic scenarios, the number of messages transmitted increases only slightly. This means that vehicles chosen as next forwarder by the sender-based mechanism are failing to rebroadcast. However, our receiver-based mechanism still is able to avoid redundant retransmissions.

Finally, Figure 3.15 shows the average delay to disseminate data messages to intended recipients. As can be observed, the delay of the protocol is not affected by inaccurate information, apart for the scenario with 800 vehicles/hour. Notice that, for low traffic scenarios, the fact that many vehicles redundantly retransmit the message does not negatively impact the delay, since the message still is transmitted every time a new neighbor is detected, thus promptly resuming the dissemination process. Moreover, under normal and high traffic scenarios, even if the sender-based approach fails, the receiver-based approach resumes the process with a delay of about τ seconds, according to Line 14 of Algorithm 6. Since in our simulations $\tau = 60$ ms, the increase in delay is not noticeable. These results show that our proposed solution is quite resistant to the presence of inaccurate information.

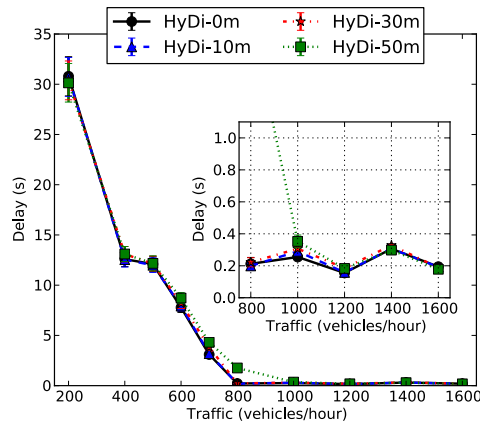


Figure 3.15: The average delay for all traffic scenarios

3.5 Chapter Remarks

In this chapter we proposed HyDi, a data dissemination protocol suitable for highway scenarios that seamlessly operate under both connected and intermittently connected vehicular ad hoc networks. Under connected networks, HyDi applies a combination of

sender-based and receiver-based techniques for information dissemination in order to avoid the so-called broadcast storm problem, which is characterized by excessive packet losses, contention and delay at the link layer. When HyDi detects that the network changed from a connected to a disconnected state, the protocol applies store-carry-forward techniques until the flow of dissemination is restored.

By means of simulation, we compared HyDi to the only two existing protocols in the literature that also focus on directional broadcasting in both dense and sparse highways, DV-CAST and SRD, and also to a simple flooding. We showed that HyDi has a very low overhead under high traffic scenarios, which is a strong indication of the high efficiency of the broadcast suppression mechanism employed by HyDi in avoiding redundant retransmissions. Moreover, HyDi outperforms all protocols when considering the average delay in dense network scenarios. This result is an indication that the contention at the link layer for HyDi is lower when compared with other protocols. Therefore, vehicles using HyDi can access the medium as soon as possible to deliver the messages to other vehicles. Finally, HyDi has the best delivery ratio under intermittently connected networks and has a delivery ratio of 100% under both normal and high traffic regimes. This result shows that HyDi is a suitable solution for safety applications that possess strict requirements regarding message delivery.

A limitation of HyDi is that it is only functional on highway scenarios. Therefore, in the next chapter we outline a solution for data dissemination that operates under urban scenarios with extreme road traffic conditions. Moreover, such solution avoids the synchronization effects introduced by the WAVE standard and it also controls the rate at which vehicles insert data into the channel. Therefore, this protocol adapts not only to the perceived road traffic condition, but also to the network traffic on the communication channel.

Chapter 4

Data Dissemination in Urban Vehicular Ad hoc Networks with Extreme Traffic Conditions

In the previous chapter, we outlined HyDi, a directional data dissemination protocol that works solely on highway scenarios. Due to the narrow applicability of such solution, in this chapter we propose ADVENT, a data dissemination protocol for urban VANETs. Contrarily to HyDi, which uses one-hop neighbor information to make decisions about which vehicles should rebroadcast, ADVENT relies exclusively on the position and heading information of the transmitters to deliver messages under dense and sparse networks. In dense scenarios, ADVENT selects vehicles inside a forwarding zone to rebroadcast messages to further vehicles. Moreover, the protocol employs implicit acknowledgements to guarantee robustness in message delivery under sparse scenarios. We also show that due to the channel hopping mechanism introduced by the new wireless communication technology for vehicular environments, even when messages at neighboring vehicles are assigned different delays to rebroadcast, they can still be transmitted at the same time, thus leading to channel contention and probably message collisions. Therefore, we outline a technique to avoid this resynchronization effect. Finally, vehicles in ADVENT adapt the rate at which they insert data into the communication channel. Therefore, ADVENT seamlessly adapts not only to the perceived road traffic condition but also to the available bandwidth. When compared to related protocols, simulation results for both Manhattan grid and real city street scenarios show that ADVENT decreases both the latency to disseminate messages and the network overhead, and also guarantees message delivery to all vehicles in the region of interest.

4.1 Introduction

Unlike traditional networks, such as the Internet, in which the unicast communication model is prevalent, applications developed for VANETs usually rely on the exchange of broadcast messages to deliver data to a group of vehicles located in a region of interest (see Figure 4.1) [Cesana et al., 2010; Marfia et al., 2013]. This process is commonly referred to as *data dissemination*. Consider, for instance, a traffic information system (TIS) application that disseminates a message to vehicles located in the downtown region of a city to notify the drivers about heavy traffic at a main exit, thus they can take alternative routes. For applications like this, especially the safety-related ones, the broadcast messages should reach all vehicles inside the region of interest in a time critical fashion without incurring a high load into the network.

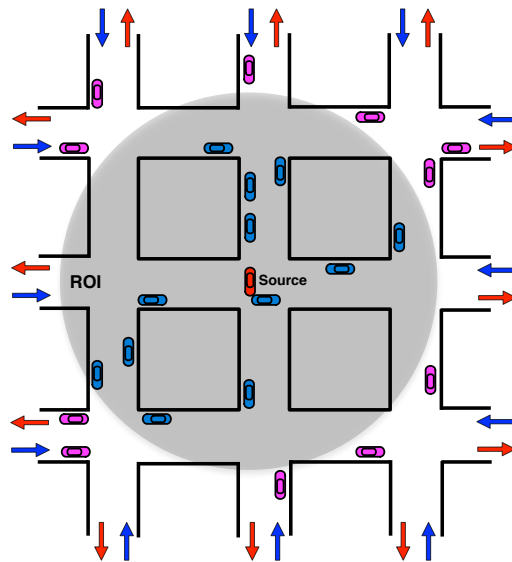


Figure 4.1: After a collision, a source vehicle produces and disseminates a warning to all vehicles inside the region of interest (ROI) defined by the application

In VANETs, due to the dynamics of the network, interesting road traffic patterns may arise. For instance, while the network may experience high traffic conditions at intersections or at rush hours, the traffic may be smooth or sparse at other regions of the city or after rush hours. Many data dissemination protocols suited for high traffic scenarios have been proposed in the literature to tackle the *broadcast storm problem* inherent to these scenarios [Wisitpongphan et al., 2007]. Moreover, some solutions for sparse traffic scenarios that deal with the *intermittently connected network problem* have also been proposed [Chen et al., 2001]. However, it is reasonable to assume that diverse traffic conditions will coexist under realistic scenarios [Bai and Krishnamachari,

2009; Uppoor and Fiore, 2012]. Therefore, data dissemination protocols for these networks should perceive the traffic condition at hand and seamlessly adapt to act accordingly. Surprisingly, most solutions for urban scenarios designed so far either focus on the broadcast storm problem or the intermittently connected network problem. Furthermore, solutions that do focus on both problems either use special infrastructure or a combination of Global Positioning System (GPS) and mapping information, such as the layout of roads [Zhao et al., 2007; Viriyasitavat et al., 2010].

Therefore, in this chapter we take a step further by proposing ADVENT, a data dissemination protocol designed to operate under diverse road traffic conditions in urban scenarios. By using only information about the transmitters, we propose a novel broadcast suppression mechanism that selects vehicles inside a *forwarding zone* to rebroadcast the messages. Thereafter, we combine this mechanism with a distance-based approach to determine a waiting delay for the vehicles to perform the rebroadcast. The idea is that vehicles inside the forwarding zone are assigned a lower waiting delay to transmit. Moreover, the identifiers of the last messages received by a vehicle are piggybacked in periodic beacons exchanged among neighboring vehicles. Therefore, when a vehicle receives a new message, it stores and carries the message around until a new encounter with an *uninformed vehicle* is made, i.e., until it encounters a vehicle that has not received such message. When such moment actually happens, the *informed vehicle* relies on the aforementioned distance-based approach to schedule a rebroadcast to the uninformed vehicle.

Nevertheless, we show that even when our distance-based mechanism assigns different waiting delays to vehicles to perform a rebroadcast, in the end, they may contend to access the channel at the same time, which may lead to message collisions. This happens due to a resynchronization effect caused by the Multichannel Operation of the IEEE 802.11p standard [IEEE, 2010]. Indeed, all data dissemination protocols for VANETs that assign different waiting delays to rebroadcast in an attempt to avoid the broadcast storm problem are vulnerable to this resynchronization effect. Hence, we propose a mechanism to avoid such resynchronization. Finally, vehicles in ADVENT adapt the rate at which they insert data into the communication channel. Therefore, ADVENT seamlessly adapts not only to the perceived road traffic condition but also to the traffic on the communication channel.

Simulation results show that, when compared to related protocols – UV-CAST [Viriyasitavat et al., 2010], ABSM [Ros et al., 2012] and Flooding – under both Manhattan grid and real city street scenarios, ADVENT is the solution that can deliver the messages to more intended recipients by incurring the lowest delay and also the lowest number of redundant retransmissions.

4.2 Related Work

Data dissemination in VANETs is a topic that has been investigated for many years. However, most solutions designed so far either focus on dense or sparse network scenarios. Furthermore, the solutions that do focus on both scenarios either employ some kind of infrastructure, like RSU or repeaters at intersections, or a combination of GPS and mapping information. Tonguz et al. [2010] and Schwartz et al. [2011] propose the Distributed Vehicular Broadcast (DV-CAST) and the Simple and Robust Dissemination (SRD), respectively. Both protocols were conceived to operate under diverse road traffic scenarios. Moreover, they rely exclusively on local one-hop neighbor information and do not employ any special infrastructure. However, these protocols were designed to perform directional data dissemination and they are used solely for highway environments. As shown in [Ros et al., 2012], data dissemination protocols designed to operate in highways do not perform well when deployed in urban environments.

Korkmaz et al. [2004] propose the Urban Multi-hop Broadcast protocol (UMB), a medium access layer (MAC) solution designed to avoid the broadcast storm and hidden terminal problems. UMB defines a Request to Broadcast/Clear to Broadcast handshaking procedure in which the farthest vehicle from the sender is selected to acknowledge the reception of the broadcast and also to rebroadcast the message to further vehicles. Furthermore, UMB employs repeaters at intersections to disseminate messages to other road directions. However, it assumes the network is always connected. Bakhouya et al. [2011] propose the Adaptive Information Dissemination (AID), a distributed statistical-based broadcast suppression protocol for VANETs. Based on the inter-arrival time between message receptions, a vehicle decides whether to rebroadcast or not. For instance, in a high density traffic scenario, after receiving some redundant retransmissions for a given message, a vehicle may decide to not rebroadcast it by assuming it was already transmitted by many other vehicles. The protocol does not use any neighbor information or any kind of infrastructure. However, it assumes an always-connected scenario.

Zhao et al. [2007] propose the Data Pouring protocol to disseminate messages along roads in an urban environment. Despite working under both dense and sparse networks, the protocol requires repeaters at intersections for buffering and forwarding messages along the intersecting roads. Yi et al. [2010] propose the StreetCast protocol, which, analogously to UMB, is a MAC layer solution. Under this scheme, RSUs are deployed at intersections to select the best relay vehicles to forward the messages. Moreover, the protocol employs a beacon control mechanism to avoid excessive periodic beacons at crowded intersections. Nevertheless, StreetCast operates only under dense

networks. Wisitpongphan et al. [2007] propose three probabilistic and distance-based broadcast suppression techniques – weighted-p-persistence, slotted-1-persistence and slotted-p-persistence – that do not require any neighbor information or special infrastructure. For instance, in the slotted-1-persistence technique, vehicles decide based on the distance to the sender when to rebroadcast a message. Vehicles farther from the sender transmit first, thus suppressing the transmission of other vehicles trying to rebroadcast. However, all three techniques assume a dense network environment.

Among the solutions that guarantee message delivery under both dense and sparse urban network scenarios and that do not employ any kind of special infrastructure, Ros et al. [2012] propose the ABSM, which uses the Connected Dominating Set (MCDS) concept. ABSM relies on the fact that the Minimum Connected Dominating Set (MCDS) provides the best-case solution for the broadcast data dissemination problem in a connected network topology. The MCDS is the smallest set of rebroadcasting vehicles that are connected to one another and all vehicles that are not in MCDS are connected to at least one vehicle in the MCDS. Therefore, assuming a connected network, if all vehicles in the MCDS broadcast a message, all vehicles in the network will be covered. However, calculating the MCDS is a NP-Hard problem. Hence, ABSM employs a heuristic that uses local one-hop or two-hop neighbor information to determine whether vehicles belong to the CDS or not. Vehicles in the CDS are scheduled first to rebroadcast the messages to other vehicles, thus suppressing the transmissions of vehicles that are not in the CDS. Moreover, reception acknowledgements are piggybacked in periodic beacons to guarantee message delivery under intermittently connected networks. In ABSM, when a new message is received by a vehicle, it waits for implicit acknowledgements from its neighbors to compute its waiting time to rebroadcast. Therefore, the latency to deliver messages depends on the periodic beacon frequency. ABSM proposes to achieve a high message delivery ratio without a great overhead, however at the expense of an increase in the delay to deliver messages.

Viriyasitavat et al. [2010] propose the Urban Vehicular Broadcast (UV-CAST) protocol for both dense and sparse networks. In UV-CAST, when a vehicle receives a new message, it uses local one-hop neighbor information to determine whether it should operate under a broadcast suppression regime or under a store-carry-forward regime. If the vehicle determines that it should operate under a broadcast suppression regime, it uses mapping information to verify whether it is at an intersection or not to properly calculate a waiting time to rebroadcast. On the other hand, if the vehicle perceives that it should operate under a store-carry-forward regime, it verifies whether it is a boundary vehicle or not (see Figure 4.2). The protocol assumes that boundary vehicles have a greater probability of encountering new neighbors. Therefore, these vehicles

store and carry the message around until they encounter uninformed neighbors. UV-CAST also uses implicit acknowledgements piggybacked in periodic beacons to identify uninformed vehicles. Notice that, only boundary vehicles are responsible for storing, carrying and forwarding messages to other vehicles. Moreover, when a vehicle receives a beacon from an uninformed neighbor, it immediately rebroadcasts the message without any explicit or implicit coordination with other vehicles in the neighborhood. This lack of coordination leads to an increase in the number of redundant retransmissions, especially under dense network scenarios, as shown in our performance analysis (c.f. Section 4.4).

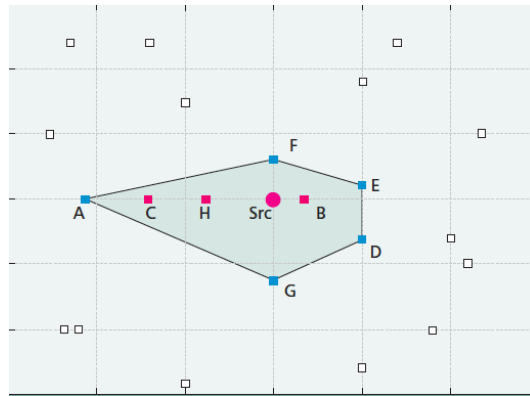


Figure 4.2: In UV-CAST, vehicles selected as border vehicles should go to the store-carry-forward state. In this figure, vehicles A , D , E , F and G are considered as border vehicles with respect to the Src vehicle. Therefore, they store-carry-forward messages received from the Src . Image Source: [Viriyasitavat et al., 2010]

4.3 Proposed Protocol

ADVENT is a lightweight and completely distributed data dissemination protocol that does not rely on any infrastructure support. Its main goal is to perform data dissemination within a region of interest (ROI) with a low overhead, short delay and high delivery ratio under both dense and sparse networks. Assuming that VANETs are confined to use the link and physical layers defined by the IEEE 802.11p standard [IEEE, 2010], ADVENT operates at the network layer to be compatible with future VANET deployments. It avoids redundant retransmissions by selecting a small subset of vehicles to disseminate the messages to all other vehicles in the ROI. Essentially, the fundamental tasks performed by ADVENT are threefold. First, it determines *what* should be this small subset of vehicles. Second, it determines *when* the vehicles in this subset should perform the dissemination. Third, it determines *how fast* the ve-

hicles should disseminate. For that, ADVENT assumes that all vehicles are equipped with a GPS and they embed their position and heading information in all transmitted data messages. Moreover, the IDs of the last messages received by a vehicle are piggybacked in periodic beacons. Therefore, periodic beacons are used as implicit acknowledgements for the messages received by the vehicles. Notice that, fundamental applications envisioned for VANETs require position information periodically, such as blind-spot detection and Emergency Electronic Brake Light (EEBL) [Hartenstein and Laberteaux, 2008]. Therefore, periodic beacons exchange is not a strong assumption. Finally, vehicles account the number of data messages lost in order to adapt the rate at which messages are inserted into the communication channel.

Figure 4.3 shows the main components of our proposed solution. It is worth noticing that ADVENT does not keep any control state of whether the protocol should operate under a broadcast suppression regime or a store-carry-forward regime. This simplifies the implementation and understanding of the protocol. ADVENT always tries to avoid unnecessary retransmissions, independently of the perceived road traffic condition. We now describe how a vehicle behaves under the proposed protocol.

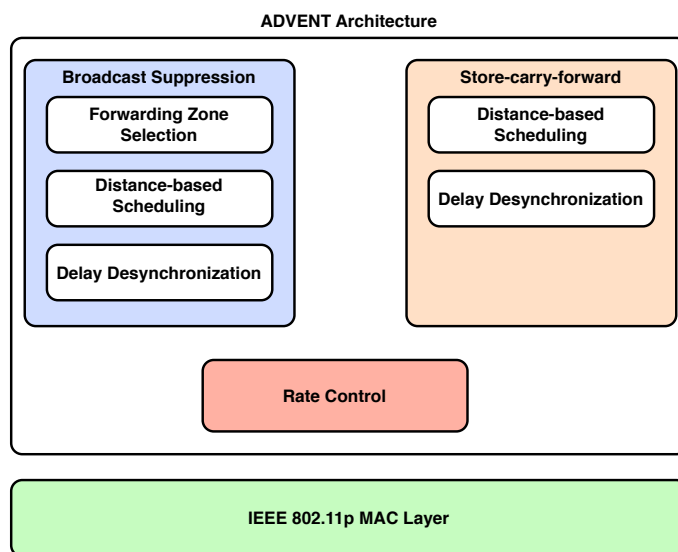


Figure 4.3: The main components of the ADVENT protocol

4.3.1 Broadcast Suppression

In ADVENT, when a vehicle receives a message for the first time, it first verifies whether it is inside the *forwarding zone* of the sender of the message. Vehicles inside the forwarding zone are given a higher priority to rebroadcast than vehicles outside it.

Hereafter, we define what a forwarding zone is and how it is used to limit the number of vehicles selected to broadcast a message.

Forwarding zone: It is defined according to the direction of the sender, as outlined in Figure 4.4. Let the direction of a vehicle be a value in the interval $[0, 2\pi]$. Moreover, let the size of the forwarding zone be controlled by an input parameter θ that lies in the interval $[0, \frac{\pi}{2}]$. When θ is 0, there is no forwarding zone, and when it is $\frac{\pi}{2}$, all neighbors of the sender are inside the forwarding zone. Therefore, using the position and direction of the sender and its own position, a vehicle is said to be inside the forwarding zone of the sender if it lies in any of the four intervals $[0 - \theta, 0 + \theta]$, $[\frac{\pi}{2} - \theta, \frac{\pi}{2} + \theta]$, $[\pi - \theta, \pi + \theta]$ or $[\frac{3\pi}{2} - \theta, \frac{3\pi}{2} + \theta]$. The rationale is that vehicles inside these regions have a higher priority over vehicles outside it, i.e., vehicles inside the forwarding zone have a lower waiting time to rebroadcast. Notice that, the information required for a vehicle to determine whether it is inside the forwarding zone of the sender or not is embedded in received data messages. For instance, in Figure 4.4, vehicles *A* and *B* are inside the forwarding zone of the *Sender*, since they lie in the interval $[0 - \theta, 0 + \theta]$. Vehicles *C* and *D* also are inside it, since they lie in the intervals $[\frac{\pi}{2} - \theta, \frac{\pi}{2} + \theta]$ and $[\pi - \theta, \pi + \theta]$, respectively. However, vehicles *E* and *F* are outside the forwarding zone of the *Sender*, since they do not lie in any of the four previously defined intervals.

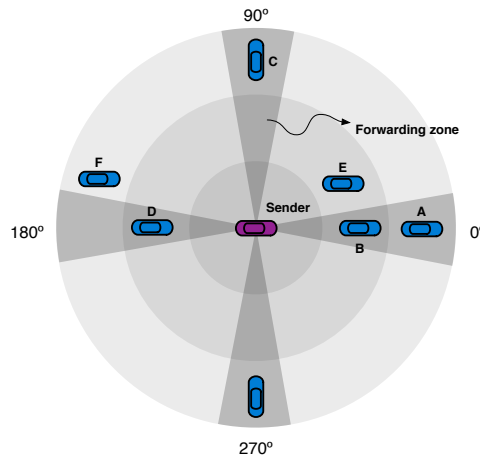


Figure 4.4: The general idea of the forwarding zone

The rationale for defining a forwarding zone is to try to limit the number of vehicles to rebroadcast a message. Moreover, vehicles lying in different intervals have a greater chance of not interfering in one another. For instance, if vehicles *A* and *C* broadcast at the same time, there is a greater chance of one not interfering in the other. However, if vehicles in the same interval broadcast a message, a collision will probably happen. Consider, for instance, vehicles *A* and *B*. Therefore, ADVENT combines

the forwarding zone concept with a modification of the distance-based broadcast suppression mechanism *slotted-1-persistence*. In this scheme, vehicles are given a waiting delay to broadcast based on the distance to the sender of the message. The higher the distance to the sender, the lower the waiting delay to transmit. Moreover, a vehicle inside the forwarding zone always is assigned a lower delay than a vehicle outside it, thus given the former a higher priority to broadcast. Hereafter, we describe how this combined approach works.

Algorithm 9: The broadcast suppression algorithm used in ADVENT

```

1 Initialize
2  $N_s \leftarrow$  number of available time-slots;
3  $R \leftarrow$  communication radius;
4  $\theta \leftarrow$  size of the forwarding zone;
5  $t \leftarrow$  base delay;

6 Event data message  $m$  received from neighbor  $s$ 
7 if vehicle is outside the region of interest specified in  $m$  or the time-to-live of  $m$  expired then
8    $\lfloor$  discard  $m$ ;
9 if  $m$  is not a duplicate then
10   add message to the list of received messages;
11   insert  $m$  ID in subsequent beacons;
12    $D \leftarrow$  distance to  $s$ ;
13    $percentageDistance \leftarrow \frac{\min(D, R)}{R}$ ;
14   if vehicle is inside forwarding zone of  $s$  then
15      $\lfloor S \leftarrow \lfloor N_s \times (1 - percentageDistance) \rfloor$ ;
16   else
17      $\lfloor S \leftarrow \lfloor N_s \times (2 - percentageDistance) \rfloor$ ;
18    $T \leftarrow S \times t$ ;
19    $T_d \leftarrow \text{desynchronize}(T)$ ;
20   schedule rebroadcast_timer for  $m$  to fire up at  $currentTime + T_d$ ;
21 else
22   if rebroadcast_timer for  $m$  is scheduled then
23      $\lfloor$  cancel rebroadcast_timer for  $m$ ;

24 Event scheduled rebroadcast_timer for  $m$  expires
25 add  $m$  to output_queue;
```

Algorithm 9 shows the main steps of the broadcast suppression mechanism employed by ADVENT. At first, when a vehicle i receives a new message m , it first verifies whether it is inside the ROI defined by m , and the time-to-live (TTL) for m shows it is

still a valid message. The ROI and TTL are defined by the application and their values are embedded in data messages. If i is outside the ROI or m is not valid anymore, then i simply discards m (lines 6–8). Otherwise, assuming that m is not a duplicate, vehicle i enters the broadcast suppression mechanism (lines 9–20). First, i adds m to the list of received messages. This way, the next periodic beacons sent by i will piggyback the ID of m and all other messages previously received that are still valid (lines 10–11). Then, i calculates the distance D to the sender of m , for instance, vehicle s (Line 12). It is worth noticing that s may not be the source for the message, just a next hop vehicle in the dissemination process. Then, i calculates the percentage distance by dividing D by the communication range R (Line 13). Notice that, we use the minimum value between D and R , since the communication range is just an estimation. Therefore, the distance between the vehicles may be bigger than the estimated communication range. After calculating the percentage distance, i verifies whether it is inside the forwarding zone of the sender s . Here, vehicles inside the forwarding zone have a higher priority to broadcast. If i is inside the forwarding zone, it calculates a time slot S according to the percentage distance and the number of available time slots for each priority (N_s), a predefined parameter to the protocol (lines 14–15). The result is a number in the interval $[0, N_s - 1]$. On the other hand, if i is not inside the forwarding zone of s , its time slot S is a number in the interval $[N_s, 2N_s - 1]$ (lines 16–17). Then, i calculates the delay T to broadcast according to its assigned time slot S and a predefined base delay t (Line 18). Like N_s , the base delay t is also a parameter to the protocol. This delay must take into account the medium access delay and propagation delay.

If it was not for the synchronization effects introduced by IEEE 802.11p, vehicle i could schedule to broadcast with a waiting delay T . However, as will be discussed in Section 4.3.3, even when two neighboring vehicles attempting to broadcast choose different values for T , the actual broadcast may happen at the same time, which may lead to contention and probably a message collision. Hence, vehicle i passes the value T to a special function called `desynchronize`, which receives the waiting delay T and returns a new waiting delay T_d (Line 19). As will be shown, the delay T_d is calculated considering the channel hopping mechanism at the MAC layer. Finally, i schedules a timer `rebroadcast_timer` with a delay T_d seconds (Line 20). When such timer finally expires, vehicle i inserts the message into an output queue (lines 24–25), which is controlled by the rate control mechanism discussed in Section 4.3.4. It is this mechanism that will determine when the message will be sent down to the MAC layer to be properly broadcasted. Finally, notice that, while the `rebroadcast_timer` for m is scheduled, if vehicle i receives a duplicate of m , it cancel the `rebroadcast_timer`, thus suppressing its own transmission (lines 21–23).

For instance, in Figure 4.4, $N_s = 3$, identified by the three different shades of grey. Therefore, vehicles inside the forwarding zone choose a time-slot in the interval $[0, 2]$. Vehicle A will rebroadcast at time-slot $S_{Sender,A} = 0$, whereas vehicle B will rebroadcast at time-slot $S_{Sender,B} = 1$. Since A is assigned an earlier time-slot, when B overhears the rebroadcast of A , it immediately cancels its own rebroadcast, thus avoiding a redundant retransmission. On the other hand, vehicles outside the forwarding zone choose a time-slot in the interval $[3, 5]$. Vehicle F will rebroadcast at time-slot $S_{Sender,F} = 3$. Since vehicle D is inside the forwarding zone and is assigned to an earlier time-slot, when F overhears the rebroadcast of D , it cancels its own rebroadcast, once again, avoiding a redundant retransmission.

4.3.2 Store-carry-forward

The aforementioned mechanism is sufficient to deliver messages to intended recipients in a reliable and efficient way under a dense network regime. However, VANETs are intermittently connected networks in which varying road traffic conditions will coexist in practice. Therefore, relying exclusively on direct relaying through multi-hop transmissions certainly results in poor delivery under sparse network regimes. To overcome such an issue, ADVENT also relies on indirect relaying through the store-carry-forward communication model. In ADVENT, after receiving a new message, vehicles store and carry the message until they leave the ROI defined by the message or the time-to-live (TTL) of the message expires, thus discarding it. Moreover, vehicles place the IDs of the last messages received in periodic beacons. Again, a vehicle places the ID of a message in periodic beacons until it leaves the ROI or the TTL of the message expires. Thereby, when a vehicle receives a periodic beacon from a neighbor that does not acknowledge the receipt of a message (*uninformed neighbor*), then the vehicle assumes the neighbor has not received the message and it schedules a rebroadcast using a distance-based mechanism that is similar to the one employed by the broadcast suppression part of the protocol. Therefore, vehicles in ADVENT always perform broadcast suppression, even when transmitting a message in the store-carry-forward state.

Algorithm 10 shows how the store-carry-forward mechanism works. When a vehicle i receives a beacon b from a neighbor s , i verifies for each message m stored on its local buffer, which ones are not acknowledged by s in b , i.e., what are the messages m on the local buffer of i for which it could not find an ID in the beacon b sent by s (lines 5–13). Then, i determines when it should broadcast these not acknowledged messages m to s (lines 7–13). The idea is to calculate a waiting delay to broadcast based on

Algorithm 10: The store-carry-forward algorithm used in ADVENT

```

1 Initialize
2  $N_s \leftarrow$  number of available time-slots;
3  $R \leftarrow$  communication radius;
4  $t \leftarrow$  base delay;
5 Event beacon b received from neighbor s
6 foreach message m in the list of received messages do
7   if m is not acknowledged in b then
8      $D \leftarrow$  distance to  $s$ ;
9      $percentageDistance \leftarrow \frac{\min(D, R)}{R}$ ;
10     $S \leftarrow \lfloor N_s \times percentageDistance \rfloor$ ;
11     $T \leftarrow S \times t$ ;
12     $T_d \leftarrow desynchronize(T)$ ;
13    schedule rebroadcast_timer for  $m$  to fire up at  $currentTime + T_d$ ;
14 Event data message m received from neighbor s
15 if m is a duplicate then
16   if rebroadcast_timer for m is scheduled then
17     cancel rebroadcast_timer for  $m$ ;
18 Event scheduled rebroadcast_timer for m expires
19 add  $m$  to output_queue;
```

the distance to the sender of the beacon. The principle is similar to the mechanism employed in the broadcast suppression procedure, however with two slightly differences. First, the forwarding zone concept is not used here. Hence, all vehicles have the same priority to broadcast. Second, vehicles closer to the sender of the beacon have a lower waiting delay to broadcast. With this in mind, vehicle i calculates the distance D to the source of the beacon, i.e., s (Line 8). Then, i calculates the percentage distance by dividing D by the communication range R (Line 9). Thereafter, i calculates the time slot S to broadcast (Line 10). Such calculation depends on the percentage distance and the number of available time slots (N_s), a predefined parameter to the protocol. The result is a number in the interval $[0, N_s - 1]$. Then, after determining the time slot to broadcast, i multiplies this value by a predefined base delay t to calculate the waiting delay to broadcast T (Line 11). Notice that the store-carry-forward also desynchronizes the delay T using the mechanism described in Section 4.3.3 (Line 12). Using the desynchronized delay T_d , i schedules the *rebroadcast_timer* for each message m not acknowledged (Line 13). Just like the broadcast suppression mechanism, when the *rebroadcast_timer* for a message m expires, i inserts it on the output queue (lines 18–19). Notice that, if i receives a duplicate for m while it is scheduled to broadcast,

i cancels its transmission (lines 14–17).

4.3.3 Delay Desynchronization

The mechanisms described so far, i.e., broadcast suppression and store-carry-forward, are adequate for data dissemination under dense and sparse wireless ad hoc networks that employ the traditional IEEE 802.11 technology. However, in VANETs, due to the use of the new Wireless Access in Vehicular Environments (WAVE) standard [IEEE, 2010], a third mechanism must be employed. WAVE was proposed to enable the reliable operation of safety applications. For that, WAVE relies on the use of multiple communication channels. The U.S. Federal Communications Commission, for instance, reserved seven non-overlapping channels in the 5.85 GHz frequency range just for vehicular communication (Figure 4.5). One of these channels is designated as the Control Channel (CCH) and the remaining six as the Service Channel (SCH). Since the standard does not mandate the use of several antennas, a channel hopping scheme is proposed. Under such scheme, every T_c seconds the transceiver is allowed to go from the CCH to the SCH, and then, after additional T_c seconds, go from the SCH back to the CCH and so on. The standard defines $T_c = 50$ ms. As can be observed in Figure 4.6, when the MAC layer receives a message from the upper layer to be sent on the SCH, but the CCH is currently active, then the message must wait for the SCH to become active in order to be transmitted. At the beginning of each channel operation, there is a guard interval period of 5 ms in which the channel is treated as busy. It is assumed that periodic beacons, mostly used by safety applications, will use the CCH, while general applications will use one of the SCHs. Every message sent to the MAC layer from an upper layer must specify in which channel it should be transmitted.

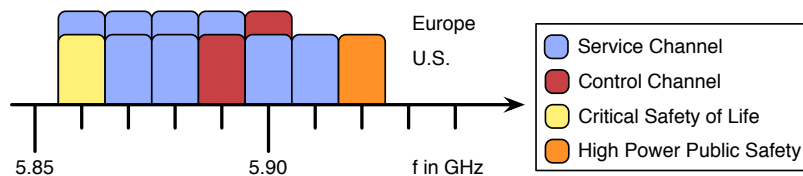


Figure 4.5: The channels in the 5.85 GHz frequency range allocated for vehicular communication

The channel hopping scheme introduces an undesirable synchronization effect. Consider the example shown in Figure 4.7. In this figure, there are two vehicles trying to rebroadcast a message they have received from a common neighbor. At time T_1 , both vehicles schedule the message to be sent down to the MAC layer. Notice that,

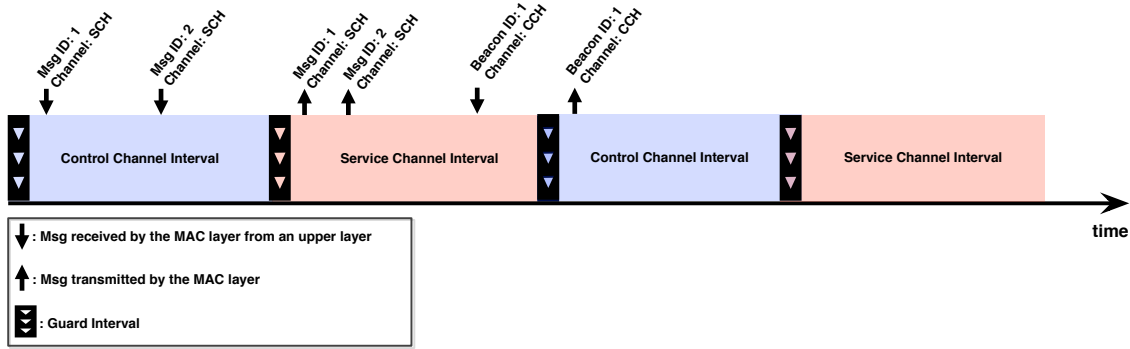


Figure 4.6: The channel hopping mechanism used in vehicular communication. When a message that must be sent on a SCH arrives at the MAC layer, but the CCH is currently active, then such message must wait until the SCH becomes active in order to be transmitted. For instance, the messages with IDs 1 and 2 are assigned for transmission on a SCH, but when they arrive at the MAC layer from an upper layer, the CCH is currently active. Then, the MAC layer buffers them until the SCH becomes active, when it finally proceeds to transmit them. Notice that, the same may happen with a message assigned for the CCH when the SCH is currently active

vehicle A schedules to send the message down to the MAC with a waiting delay of 10 ms, while vehicle B assigns a waiting delay of 25 ms. Without the channel hopping mechanism, A would transmit first, B would overhear the transmission of A and would suppress its own transmission, thus avoiding a redundant retransmission. However, this is not what actually happens in a channel hopping mechanism. At time T_2 , i.e., 10 ms after T_1 , vehicle A sends the message down to the MAC layer for transmission on the SCH. Notice, however, that the CCH is currently active. Hence, the MAC layer at vehicle A buffers the message until the SCH becomes active. At time T_3 , i.e., 25 ms after T_1 , it's the moment for vehicle B to send the message down to the MAC layer, also for transmission on the SCH. As in A , the MAC layer at vehicle B also buffers the message, since the CCH is currently active. Finally, at time T_4 , when the SCH becomes active, the MAC layer at both vehicles listens the medium and discover it is not busy. Therefore, they transmit the message at the same time, thus leading to a collision. Notice that, despite the fact that the broadcast suppression algorithm did its proper job of assigning different waiting delays for the vehicles to transmit (desynchronization), in the end, the transmissions occurred on the same moment (resynchronization). Hence, we argue that a broadcast suppression solution alone is not enough to avoid the broadcast storm problem in dense VANETs, especially for data burst dissemination [Eckhoff et al., 2012].

To overcome the aforementioned issue, we propose a mechanism that receives a

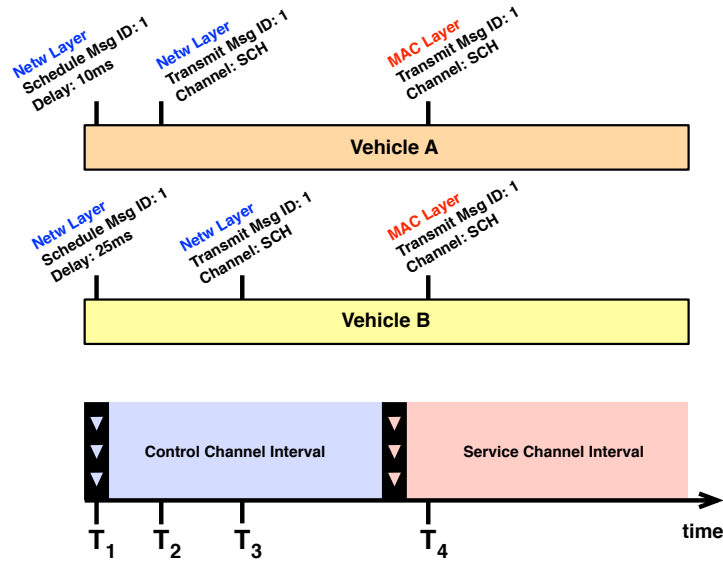


Figure 4.7: The synchronization effect introduced by the channel hopping mechanism used in the IEEE 802.11p MAC layer. Here, vehicles A and B schedules with different waiting delays a message to be sent down to the MAC layer. Indeed, the messages are sent down to the MAC layer of each vehicle at different moments in time. However, since the CCH was active when the MAC layer receives the messages and they are assigned for transmission on the SCH, the MAC layer buffers them and waits for the SCH to become active. When the SCH finally becomes active, both vehicles transmit the messages at the same time, thus leading to a collision

waiting delay T calculated by any broadcast suppression approach and, if necessary, recalculates a new waiting delay T_d according to the channel hopping regime. Algorithm 11 shows how this mechanism works. It is assumed that the new waiting delay T_d calculated by this solution is for a message to be transmitted on the SCH. However, the algorithm can be easily extended to include the calculation of waiting delays for messages intended for transmission on the CCH. Recall that, in algorithms 9 and 10, after calculating the waiting delay T for a vehicle to broadcast a message m (i.e., the delay to send the message down to the MAC layer), they call the function `desynchronize`, which returns a new waiting delay T_d to broadcast.

The general idea of Algorithm 11 is to add to the original waiting delay T to transmit a message m , an additional delay T_a . Such additional delay T_a is the amount of time that m would perceive the CCH as active if it was immediately sent to the MAC layer and the MAC layer had waited T seconds before trying to transmit m on the SCH. For instance, in Figure 4.8, a vehicle receives a message m at T_1 and its broadcast suppression protocol calculates a waiting delay $T = 60$ ms to broadcast. Notice that, at T_1 , the CCH channel is active and will remain so for more $T_s = 5$ ms. In

Algorithm 11: desynchronize function

```

1 Initialize
2  $T \leftarrow$  waiting delay passed to the function;
3  $T_c \leftarrow$  channel time, i.e., for how long each channel can be active (50 ms);
4  $T_s \leftarrow$  remaining time until a channel switch;
5 if CCH is currently active then
6    $cch\_cycles \leftarrow \left\lfloor \frac{T}{T_c} \right\rfloor$ ;
7    $T_a \leftarrow T_s + (cch\_cycles \times T_c)$ ;
8    $T_d \leftarrow T + T_a$ ;
9 else
10   $T_d \leftarrow T$ ;
11   $T_{tmp} \leftarrow T - T_s$ ;
12  if  $T_{tmp} > 0$  then
13     $cch\_cycles \leftarrow \left\lfloor \frac{T_{tmp}}{T_c} \right\rfloor$ ;
14     $T_a \leftarrow cch\_cycles \times T_c$ ;
15     $T_d \leftarrow T + T_a$ ;
16 return  $T_d$ 

```

this example, Algorithm 11 calculates $T_a = 55$ ms, because if the network layer sends m to the MAC layer at T_1 , and the MAC layer waits $T = 60$ ms before trying to transmit m on the SCH, then m would perceive the CCH as active for 55 ms. It is worth noticing that, if the network layer does not use the desynchronization mechanism proposed here, it will schedule m with a delay T . Therefore, at T_2 , it sends m down to the MAC layer for transmission in the SCH. However, it cannot be immediately transmitted because the CCH is current active at T_2 , which may lead to a resynchronization effect. On the other hand, if the network layer does use Algorithm 11, it schedules m with a waiting delay $T_d = T + T_a$. Therefore, at T_3 , the network layer sends m down to the MAC layer and it can be immediately transmitted, because the SCH is currently active. Indeed, by using a waiting delay $T_d = T + T_a$ guarantees that when the network layer sends m down to the MAC layer, the SCH always will be active, thus avoiding waiting delays resynchronization. In summary, relying on Algorithm 11 produces the same result as using the normal waiting delay T with a clock that works only when the SCH is active, i.e., time advances only when the SCH is active.

4.3.4 Rate Control

The last component of ADVENT is its rate control mechanism. When the waiting delays defined by the broadcast suppression and store-carry-forward mechanisms for

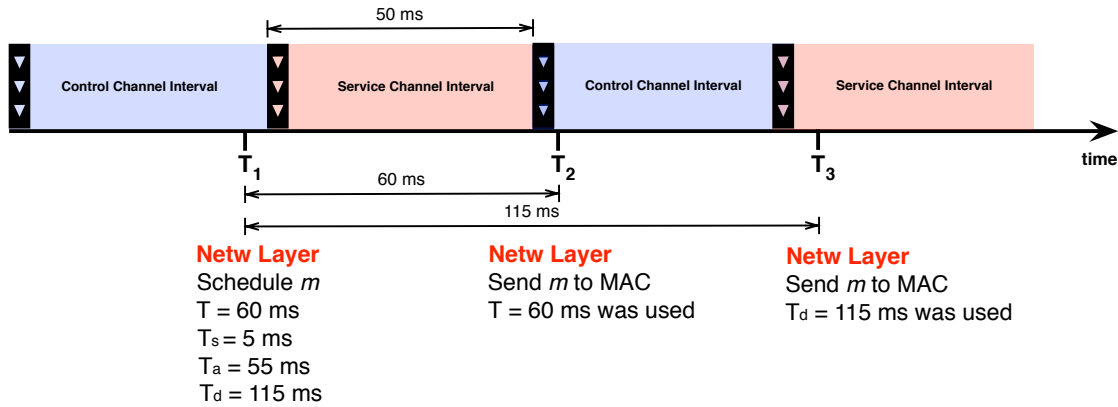


Figure 4.8: Example showing how the desynchronization mechanism works

a message finally expires, such message is inserted into an output queue controlled by the rate control mechanism (see algorithms 9 and 10). It is the rate control that determines how fast messages waiting in the output queue are sent down to the MAC layer for proper transmission. In summary, the rate control removes messages from the output queue at the same rate that the MAC layer operates. However, as messages are lost, such rate starts to diminish. Algorithm 12 shows the main steps taken by this component. Initially, we define the data rate for this component to the bit rate of the MAC layer (Line 2). Moreover, we assume it is possible to retrieve from the MAC layer the number of messages lost since the SCH became active (Line 3). When the timer used by the rate control component expires (Line 4), a message m is removed from the front of the output queue (Line 5). Then, it is immediately sent down to the MAC layer (Line 6). The rate control mechanism must decide now when the next message in the output queue will be sent down to the MAC. Therefore, we first calculate the new operational data rate, which depends on the number of messages lost (Line 7). Then, using this operational data rate, we calculate the transmission delay for m (Line 8). Such delay is used to set up a timer for the next transmission (Line 9). By using this simple mechanism, ADVENT is able to adapt not only to varying road traffic conditions, but also to the amount of data traffic on the communication channel.

4.4 Performance Analysis

To assess the performance of ADVENT, we performed a series of simulations using the OMNeT++ 4.2.2 simulator¹ and compared it to three related protocols – UV-CAST [Viriyasitavat et al., 2010], ABSM [Ros et al., 2012] and Flooding. As discussed

¹<http://www.omnetpp.org>

Algorithm 12: Rate control mechanism

```

1 Initialize
2  $datarate \leftarrow$  MAC layer bit-rate;
3  $lost\_messages \leftarrow$  lost messages since SCH become active;
4 Event scheduled timer send_data expires
5  $m \leftarrow output\_queue.front()$ ;
6 send  $m$  down to MAC layer;
7  $op\_datarate \leftarrow \frac{datarate}{1+lost\_messages}$ ;
8  $transmission\_delay \leftarrow \frac{m.length}{op\_datarate}$ ;
9 schedule send_data to fire up at  $currentTime + transmission\_delay$ ;

```

in Section 4.2, UV-CAST requires a combination of GPS and mapping information to identify whether a vehicle is at a crowded intersection or not. Moreover, only a subset of vehicles go to the store-carry-forward state. ABSM is a protocol that relies on the connected dominating set concept to choose the vehicles to rebroadcast. Finally, Flooding is a simple implementation of a dissemination protocol in which all vehicles immediately rebroadcast received messages exactly once.

4.4.1 Simulation Setup

We evaluated all protocols under low, normal and high traffic conditions in both a Manhattan grid as well as in a real city street scenario. The Manhattan grid scenario is comprised of ten evenly-spaced vertical and horizontal double-lane streets in an area of 1 km². We also consider signal attenuation effects caused by buildings. For that, we assume that each block in the grid contains a 80 m x 80 m obstacle, representing a high-rise building. To assess different road traffic conditions, we vary the density in the network from 20 vehicles/km² to 400 vehicles/km². We relied on the SUMO 0.17.0 mobility simulator [Behrisch et al., 2011] to simulate realistic vehicle movements. A vehicle positioned approximately at the center of the network generates 100 messages of 2048 bytes, which are disseminated at a rate of 1.5 Mbit/s to all vehicles in the network.

The real city street scenario is represented by a two hour mobility dataset that covers a 400 km² area of the city of Cologne, Germany [Uppoor and Fiore, 2012]. Such dataset is realistic from both macroscopic and microscopic viewpoints [Uppoor and Fiore, 2011]. Besides an accurate mobility trace, informations like different types of roads, buildings and road signalization make this scenario much more realistic than a Manhattan grid. To assess the behavior of the protocols under different road traffic conditions, we performed a data dissemination at five different times of the day (06:30,

06:45, 07:00, 07:15 and 07:30 am). Notice that, as the time of the day increases, so does the road traffic. The dissemination is initiated by a vehicle positioned approximately at the center of the network. Such source vehicle generates 100 messages of 2048 bytes, which are disseminated at a rate of 1.5 Mbit/s to all vehicles located in a region of interest (ROI) with a radius of 2 km.

Moreover, we used the Veins 2.1² [Sommer et al., 2011a] network model to increase the accuracy of our results, since it implements both an obstacle model for signal attenuation effects and the WAVE standard for vehicle communications. As main parameters, we set the bit rate at the MAC layer to 18 Mbit/s and the transmission power to 0.98 mW, resulting in a transmission range (R) of about 200 m under the two-ray ground propagation model. Moreover, we set the beacon message size to 32 bytes. Periodic beacons are generated every 1 s. Regarding ADVENT's specific parameters, we set the forwarding zone angle (θ) to $\frac{\pi}{6}$, the number of time slots (N_s) to 5 and the configured delay (t) to 50 ms. The results represent the mean of 50 executions for each evaluated scenario with a confidence interval of 95%. Table 4.1 shows the main parameters used in our assessment.

Table 4.1: Simulation parameters used in our assessment

| Parameter | Value |
|------------------------------------|-----------------|
| Transmission power | 0.98 mW |
| Transmission range (R) | 200 m |
| MAC bit rate | 18 Mbit/s |
| Forwarding zone angle (θ) | $\frac{\pi}{6}$ |
| Number of time slots (N_s) | 5 |
| Delay (t) | 50 ms |
| Beacon size | 32 bytes |
| Beacon frequency | 1 Hz |
| Data message size | 2048 bytes |
| Number of data messages produced | 100 |
| Number of runs | 50 |
| Confidence interval | 95% |

4.4.2 Evaluated Metrics

The metrics used to evaluate the reliability, scalability and efficiency of ADVENT are:

²<http://veins.car2x.org>

- **Delivery ratio:** the percentage of data messages generated by the source vehicle that is actually delivered to intended recipients. It is expected that dissemination protocols must achieve a delivery ratio of 100%.
- **Transmitted messages:** total number of data messages transmitted by all vehicles in the network. This metric is a strong indication of whether a protocol avoids redundant retransmissions, which may cause the broadcast storm problem under extreme road traffic conditions.
- **Delay:** the average time it takes for a data message to travel from the source vehicle to intended recipients. This metric is important for time-constrained messages that must be disseminated as quickly as possible (e.g., accident warnings).

4.4.3 Manhattan Grid Results

Figure 4.9 shows the delivery ratio for all protocols under varying traffic densities. As we can see, overall, ADVENT is the most reliable solution. For instance, for a traffic density of 20 vehicles/km², ADVENT delivers almost 10% more messages than ABSM, and 20% more than UV-CAST. Recall that, in both ADVENT and ABSM, all vehicles store and carry the message around until it expires or the vehicle leaves the ROI. In UV-CAST, only a subset of vehicles are responsible for doing the store-carry-forward. Therefore, ADVENT and ABSM always perform better than UV-CAST when the traffic is sparse. Flooding does not rely on the store-carry-forward communication model, thus explaining its poor performance under sparse scenarios. As the traffic density increases, so does the delivery ratio for all protocols. However, as the traffic density keeps increasing (starting at 200 vehicles/km²), ADVENT guarantees a delivery ratio of 100%, while the delivery for ABSM and UV-CAST starts to deteriorate. This probably happens as a result of the broadcast storm under high traffic, since these protocols are not avoiding message collisions and consequently message losses. This result shows the effectiveness of the proposed mechanisms used by ADVENT (broadcast suppression, delay desynchronization and rate control mechanisms) in avoiding the broadcast storm problem. Therefore, we can argue that ADVENT is a reliable solution for both dense and sparse VANETs.

Figure 4.10 shows the total number of data messages transmitted by all vehicles under different traffic densities. When the network is sparse, ADVENT is the protocol with the highest overhead. There are two possible explanations for such a fact. First, ADVENT is the protocol that delivers more messages to intended recipients when the traffic is sparse, thus requiring more message transmissions. Second, despite the fact

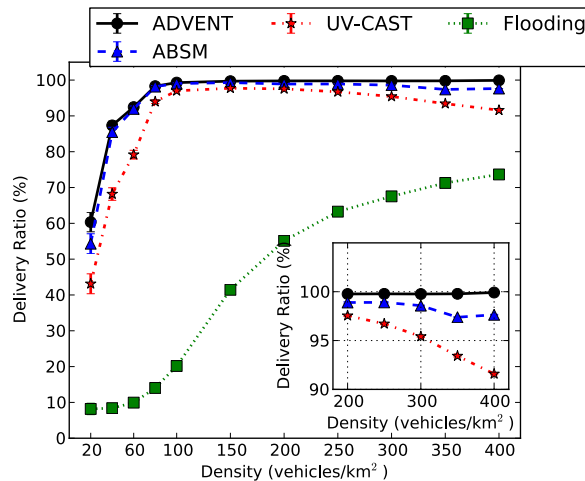


Figure 4.9: Delivery ratio for the Manhattan grid scenario

that the store-carry-forward mechanism employed in ADVENT guarantees good delivery results, it does not avoid redundant retransmissions. As the real city results will show, this second explanation is more plausible. As the road traffic increases and the network becomes denser, ADVENT transmits less messages than all the other protocols. For instance, at a density of 400 vehicles/km², ADVENT transmits about 25% less messages than ABSM, 50% less than Flooding and 65% less than UV-CAST. It is at high densities that the broadcast storm problem is an issue and ADVENT is avoiding redundant retransmissions at these scenarios. It is worth noticing that, when the traffic is high, UV-CAST transmits much more messages than Flooding. This happens because when some vehicles do not receive the messages being disseminated, e.g., due to collisions, the *informed vehicles* in the store-carry-forward state tries to deliver them once again to these *uninformed vehicles*. Recall, however, that vehicles in the store-carry-forward do not employ any kind of coordination to transmit a message to an uninformed vehicle. Therefore, this leads to more message collisions and consequently an explosion in the number of message transmissions.

Figure 4.11 shows the average delay to disseminate the data messages to all intended recipients under different traffic densities. Notice that, the high delay for ADVENT, ABSM and UV-CAST in sparse scenarios is due to the fact that these protocols perform store-carry-forward, thus vehicles carry the messages around until they encounter an uninformed neighbor. When the network is sparse, ADVENT is the protocol with the lowest delay, which is quite interesting since ADVENT delivers the messages to more intended recipients under these scenarios. We were expecting that to deliver the messages to more vehicles under sparse scenarios, it would be necessary to carry them for longer times, thus increasing the delay. However, this result shows

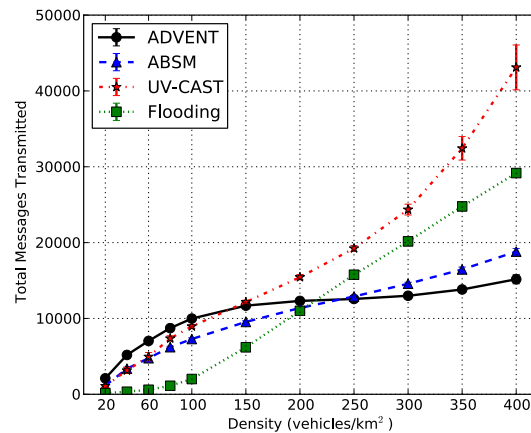


Figure 4.10: Total number of data messages transmitted for the Manhattan grid scenario

that in both ABSM and UV-CAST, informed vehicles are encountering uninformed neighbors, however they are missing the opportunities to disseminate the messages further. As the network becomes denser, ADVENT still takes less time to deliver the messages than ABSM and UV-CAST, as shown in the inset plot. This result shows that ADVENT is a suitable solution for time-strict applications, such as warning message dissemination.

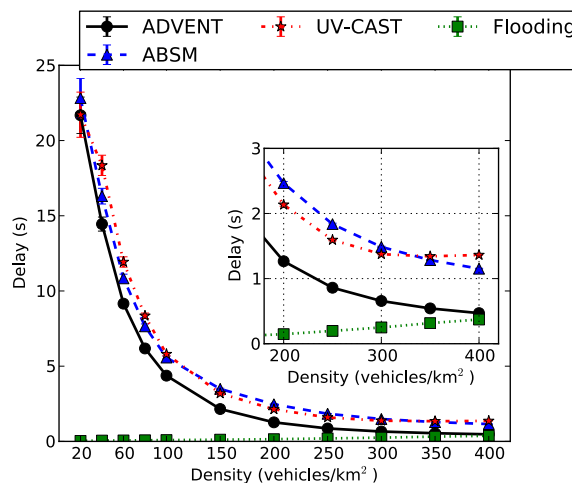


Figure 4.11: Average delay to disseminate the data messages to all intended recipients in the Manhattan grid scenario

4.4.4 Real City Results

In this section we assess the performance of all protocols under a real city scenario. Notice that, in this scenario, the street layouts are not as regular as in the Manhattan grid scenario and the traffic flows are accurately represented. Moreover, we investigate different road traffic conditions by performing the dissemination at different times of the day. Figure 4.12 shows the delivery ratio for all protocols. As we can observe, also in a real city scenario, ADVENT is the solution that can deliver the messages to more intended recipients, thus showing its reliability. For instance, for the dissemination at 07:30 am, ADVENT delivers to intended recipient about 10% more messages than ABSM. When compared to UV-CAST, the difference is about 50%. Furthermore, it is possible to notice that as the time of the day increases, i.e., as the road traffic increases, the delivery ratio for ABSM and UV-CAST starts to deteriorate, just like in the Manhattan grid scenario.

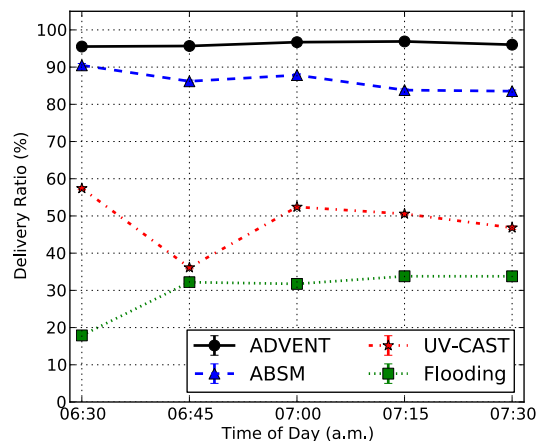


Figure 4.12: Delivery ratio for the real city scenario

Figure 4.13 shows the total number of data messages transmitted. ADVENT is the protocol with the highest number of data messages transmitted. As will become clear in the next result, in this real city scenario, most messages are delivered using the store-carry-forward mechanism. We have already shown that ADVENT is the solution that produces the highest amount of transmissions when the store-carry-forward communication model prevails. Therefore, this explains why ADVENT transmits so much under this real city scenario. Notice, however, that even by transmitting more messages and consequently increasing the communication channel use, ADVENT still can deliver more messages than the related protocols. Hence, the increase in the number of transmitted messages in this scenario is not leading to the broadcast storm prob-

lem. Nonetheless, this result shows that a better store-carry-forward mechanism for ADVENT is necessary.

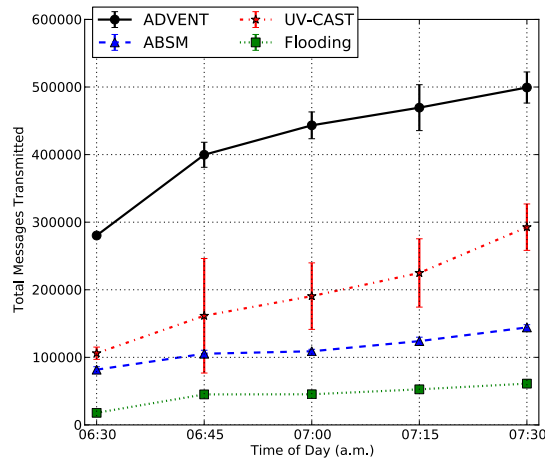


Figure 4.13: Total number of data messages transmitted for the real city scenario

Finally, Figure 4.14 shows the average delay to deliver the data messages to intended recipients. Notice the high delivery delay for ADVENT, ABSM and UV-CAST. This shows that most messages are being delivered by using the store-carry-forward communication model, thus confirming the last result. Moreover, the higher delay for ADVENT when compared to ABSM and UV-CAST is due to the fact that ADVENT delivers the messages to more intended recipients, as previously shown.

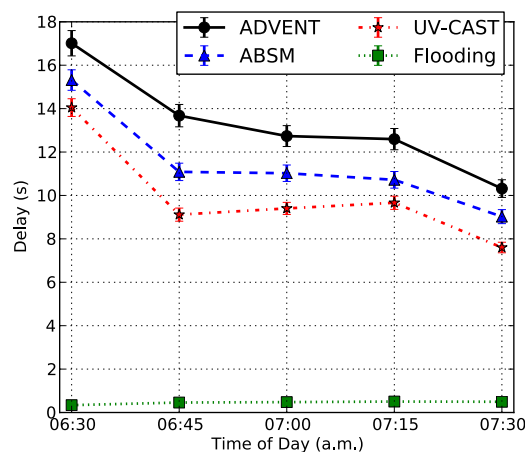


Figure 4.14: Average delay to disseminate the data messages to all intended recipients in the real city scenario

4.5 Chapter Remarks

This chapter presented ADVENT, a data dissemination protocol for urban VANETs with diverse road traffic conditions. We proposed a broadcast suppression mechanism in which vehicles determine whether they are inside a forwarding zone. Then, using such information, vehicles decide when to broadcast. Essentially, vehicles inside such forwarding zone are assigned shorter delays to broadcast when compared to vehicles outside it. Moreover, a vehicle in ADVENT embeds the IDs of the last messages received. This way, other vehicles are able to verify whether it has received all messages that have been disseminated so far, thus increasing the robustness of the protocol for both sparse and dense network scenarios. Due to the synchronization problem introduced by the channel hopping mechanism used in the new standard for vehicular communication, we proposed a technique to overcome such an issue. Finally, we outlined a mechanism that controls the rate at which vehicles insert data messages into the channel based on the available bandwidth.

When compared to three related protocols – ABSM, UV-CAST and Flooding –, simulation results showed that the proposed solution has the best delivery results under both dense and sparse scenarios. Regarding the number of transmitted messages, we showed that ADVENT has the lowest overhead under dense scenarios, thus showing the effectiveness of the proposed mechanisms in avoiding the broadcast storm problem. However, under sparse scenarios, its store-carry-forward mechanism produces a great number of redundant retransmissions. Finally, Table 4.2 presents a summary of the main characteristics of the data dissemination solutions for VANETs studied in the last two chapters.

Table 4.2: Comparison of protocols studied in chapters 3 and 4

| Protocol | Scenarios | Network Regimes | Infrastructure Required | Rate Control |
|-----------------|------------------|------------------------|--------------------------------|---------------------|
| DV-CAST | Highway | Both | No | No |
| SRD | Highway | Both | No | No |
| HyDi | Highway | Both | No | No |
| UMB | Urban | Connected | Yes | No |
| AID | Urban | Connected | No | No |
| StreetCast | Urban | Connected | No | No |
| Data Pouring | Urban | Both | Yes | No |
| UV-CAST | Urban | Both | No | No |
| ABSM | Urban | Both | No | No |
| ADVENT | Urban | Both | No | Yes |

Chapter 5

Final Remarks

5.1 Conclusions

In this thesis we showed that the activity known as data dissemination is very common in accomplishing many different tasks in wireless ad hoc networks. For instance, it has been used by unicast and multicast routing protocols in MANETs to establish routes between transmitter and receivers; to replicate data, reconfigure, query and reprogram WSNs; or as a data communication procedure to notify drivers about events of interest in VANETs.

Given the importance of such activity, we first investigated how it may be used in the design of a data replication and storage mechanism for WSNs. Therefore, we proposed ProFlex, a data dissemination solution that relies on a small subset of powerful nodes to create replication structures and disseminate the sensed data to the nodes in the network. In this scheme, the sensed data is intelligently replicated and disseminated to sensor nodes in such a way that a mobile sink can later visit a small subset of nodes in order to collect the sensed data produced by the whole network. We showed that such solution may be used as an alternative approach for the traditional data collection task in WSNs, where nodes route packets to a sink positioned at the border of the network. Simulation results showed that ProFlex has the lowest overhead when compared to existing approaches, however it possesses a slightly worse dissemination and collection efficiency. Hence, we showed that by taking advantage of the data redundancy and correlation inherent to WSNs, it is possible to decrease the overhead of ProFlex even more and at the same time accomplish a dissemination and collection efficiency similar to existing approaches. Finally, we discussed how such solution may be used to design a distributed Traffic Information System (TIS) for VANETs. Regarding the research opportunities, it would be interesting to actually design this TIS based

on ProFlex and assess its performance when compared to other approaches. Moreover, a salvation mechanism for scenarios with node failures would be extremely useful to improve the performance of ProFlex. An interesting enhancement to the protocol is to consider other requirements, e.g., battery usage, memory availability, link quality, etc., to determine where to store replicated data packets.

Thereafter, we investigated how data dissemination may be used as a communication procedure to report events to drives that are inside a region of interest in VANETs. Hence, we proposed HyDi, a data dissemination solution for highway VANETs that operates under varying road traffic conditions, i.e., dense and sparse networks. Contrarily to most related protocols, such solution does not rely on any fixed infrastructure and vehicles make all decisions based on the knowledge of their one-hop neighbors. Simulation results showed that, when compared to related protocols, HyDi is the protocol with the best delivery ratio under sparse traffic, and it has the lowest delay and lowest overhead under dense traffic. Finally, we showed that our solution is robust to GPS errors. However, a limitation of such protocol is that it works solely under highway scenarios.

Therefore, we proposed ADVENT, a data dissemination solution for urban VANETs with extreme traffic conditions. Contrarily to HyDi, vehicles in ADVENT make all decisions based only on the information about the transmitter of a packet. Moreover, we outlined a technique to avoid the synchronization effects introduced by the IEEE 802.11p standard. We also proposed a mechanism that enables ADVENT to adapt the rate at which a vehicle inserts data into the channel according to the perceived available bandwidth. Therefore, such solution is able to adapt not only to the perceived road traffic condition, but also to the network traffic on the communication channel. Simulation results showed that ADVENT has the best delivery, the lowest delay and lowest overhead under dense traffic when compared to literature protocols. However, under sparse networks, the store-carry-forward mechanism employed by ADVENT does not avoid redundant retransmission. Hence, a direct research opportunity is to investigate new methods for data dissemination under sparse VANETs. Moreover, both HyDi and ADVENT possess many parameters that have a great effect on the performance of these protocols. In this thesis, we set those parameters to values that we believed would lead to good performance results. These choices were based on a shallow parameter study. However, it would be interesting to make a parameter study in the same way as we performed for ProFlex in Chapter 2, and also propose mechanisms to automatically adapt these parameters according to the perceived road traffic condition, link quality or available bandwidth.

5.2 Publications

In the following sections, we present a list of publications produced during the PhD. The list is divided into periodical and conference papers. Those identified with a (★) are directly related to this work.

5.2.1 Periodicals

★ Maia, Guilherme; Guidoni, Daniel L.; Viana, Aline C.; Aquino, Andre L.L.; Mini, Raquel A.F.; Loureiro, Antonio A.F. (2013). A distributed data storage protocol for heterogeneous wireless sensor networks with mobile sinks. *Ad Hoc Networks*, 11(5):1588-1602.

Maia, Guilherme; Aquino, Andre L.L.; Guidoni, Daniel L.; Loureiro, Antonio A.F. (2013). A multicast reprogramming protocol for wireless sensor networks based on small world concepts. *Journal of Parallel and Distributed Computing*, 73(9):1277-1291.

Maia, Guilherme; Vaz De Melo, Pedro O. S.; Guidoni, Daniel L.; Souza, Fernanda S.H.; Silva, Thiago H.; Almeida, Jussara M.; Loureiro, Antonio A.F. (2013). On the analysis of the collaboration network of the Brazilian symposium on computer networks and distributed systems. *Journal of the Brazilian Computer Society*, 19(3):361-382.

5.2.2 Conferences

★ Maia, Guilherme; Boukerche, A.; Aquino, A. L. L.; Viana, A. C.; Loureiro, A. A. F. (2013). A Data Dissemination Protocol for Urban Vehicular Ad hoc Networks with Extreme Traffic Conditions. In *IEEE International Conference on Communications (ICC '13)*, 1-5.

★ Maia, Guilherme; Villas, Leandro A.; Boukerche, A.; Viana, A. C.; Aquino, A. L. L.; Loureiro, A. A. F. (2013). Data Dissemination in Urban Vehicular Ad hoc Networks with Diverse Traffic Conditions. In *IEEE Symposium on Computers and Communications (ISCC '13)*, 1-6.

★ Maia, Guilherme; Rezende, Cristiano; Villas, Leandro A.; Boukerche, A.; Viana, A. C.; Aquino, A. L. L.; Loureiro, A. A. F. (2013). Traffic Aware Video

Dissemination Over Vehicular Ad hoc Networks. In *ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '13)*, 1-8.

Villas, Leandro A.; Boukerche, A.; Guidoni, D. L.; Maia, Guilherme; Loureiro, A. A. F. (2013). A Joint 3D Localization and Synchronization Solution for Wireless Sensor Networks Using UAV. In *IEEE Conference on Local Computer Networks (LCN '13)*, 1-4.

Boukerche, A.; Villas, Leandro A.; Guidoni, D. L.; Maia, Guilherme; Cunha, F. D.; Ueyama, Jo; Loureiro, A. A. F. (2013). A New Solution for the Time-Space Localization Problem in Wireless Sensor Network using UAV. In *ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet '13)*, 1-8.

★ Maia, Guilherme; Guidoni, D. L.; Viana, A. C.; Aquino, A. L. L.; Mini, R. A. F.; Loureiro, A. A. F. (2012). Um Protocolo de Distribuição de Dados para Redes de Sensores Sem Fio Heterogêneas com Sink Móvel. In *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC '12)*, 1-14.

Maia, Guilherme; Guidoni, D. L.; Silva, Thiago H.; SOUZA, Fernanda S. H.; Vaz de Melo, P. O.; Soares, César A. O.; Almeida, Jussara M.; Loureiro, A. A. F. (2012). Análise da Rede de Colaboração do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos: As Primeiras 30 Edições. In *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC '12)*, 1-14.

★ Maia, Guilherme; Aquino, A. L. L.; Viana, A. C.; Boukerche, A.; Loureiro, A. A. F. (2012). HyDi: A Hybrid Data Dissemination Protocol for Highway Scenarios in Vehicular Ad hoc Networks. In *ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet '12)*, 1-8.

Bibliography

- Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks*, 38(4):393--422.
- Akyildiz, I. F., Vuran, M. C., and Akan, O. B. (2004). On exploiting spatial and temporal correlation in wireless sensor networks. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt '04)*, pages 71--80.
- Anastasi, G., Conti, M., and Di Francesco, M. (2008). Data collection in sensor networks with data mules: An integrated simulation analysis. In *IEEE Symposium on Computers and Communications (ISCC '2008)*, pages 1096--1102.
- Aquino, A. L. L. and Nakamura, E. F. (2009). Data centric sensor stream reduction for real-time applications in wireless sensor networks. *Sensors*, 9(12):9666--9688.
- Bai, F. and Krishnamachari, B. (2009). Spatio-temporal variations of vehicle traffic in VANETs: facts and implications. In *ACM International Workshop on Vehicular InterNetworking (VANET '09)*, pages 43--52.
- Bai, F. and Krishnamachari, B. (2010). Exploiting the Wisdom of the Crowd: Localized, Distributed Information-Centric VANETs. *IEEE Communications Magazine*, 48(5):138--146.
- Bakhouya, M., Gaber, J., and Lorenz, P. (2011). An adaptive approach for information dissemination in vehicular ad hoc networks. *Journal of Network and Computer Applications*, 34(6):1971--1978.
- Bar-Yossef, Z., Friedman, R., and Kliot, G. (2008). RaWMS - Random Walk Based Lightweight Membership Service for Wireless Ad Hoc Networks. *ACM Transactions on Computer Systems*, 26(2):1--66.
- Barbera, M. V., Stefa, J., Viana, A. C., de Amorim, M. D., and Boc, M. (2011). VIP Delegation: Enabling VIPs to Offload Data in Wireless Social Mobile Networks. In

- International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS '11)*, pages 1--8.
- Basagni, S., Carosi, A., Melachrinoudis, E., Petrioli, C., and Wang, Z. M. (2008). Controlled sink mobility for prolonging wireless sensor networks lifetime. *Wireless Networks*, 14(6):831--858.
- Behrisch, M., Bieker, L., Erdmann, J., and Krajzewicz, D. (2011). SUMO - Simulation of Urban MObility: An Overview. In *International Conference on Advances in System Simulation (SIMUL '11)*, pages 63--68.
- Boukerche, A. (2004). Performance evaluation of routing protocols for ad hoc wireless networks. *Mobile Networks and Applications*, 9(4):333--342.
- Boukerche, A. (2005). *Handbook of Algorithms for Wireless Networking and Mobile Computing*. Chapman & Hall/CRC. ISBN 1584884657.
- Boukerche, A. (2008). *Algorithms and Protocols for Wireless Sensor Networks*. Wiley-IEEE Press. ISBN 0471798134, 9780471798132.
- Cavalcanti, D., Agrawal, D., Kelner, J., and Sadok, D. (2004). Exploiting the Small-World Effect to Increase Connectivity in Wireless Ad hoc Networks. In *IEEE International Conference on Telecommunications (TSP '04)*, pages 388--393.
- Cesana, M., Fratta, L., Gerla, M., Giordano, E., and Pau, G. (2010). C-VeT the UCLA campus vehicular testbed: Integration of VANET and Mesh networks. In *European Wireless Conference (EW '10)*, pages 689--695.
- Chatzigiannakis, I., Kinalis, A., and Nikolettseas, S. (2008). Efficient data propagation strategies in wireless sensor networks using a single mobile sink. *Computer Communications*, 31(5):896--914.
- Chen, F.-W. and Kao, J.-C. (2013). Game-based broadcast over reliable and unreliable wireless links in wireless multihop networks. *IEEE Transactions on Mobile Computing*, 12(8):1613--1624.
- Chen, Z. D., Kung, H., and Vlah, D. (2001). Ad hoc relay wireless networks over moving vehicles on highways. In *ACM International Symposium on Mobile Ad hoc Networking & Computing (MobiHoc '01)*, pages 247--250.
- Eckhoff, D., Sommer, C., and Dressler, F. (2012). On the Necessity of Accurate IEEE 802.11p Models for IVC Protocol Simulation. In *IEEE Vehicular Technology Conference (VTC '12)*, pages 1--5.

- Feeney, L., Ahlgren, B., and Westerlund, A. (2001). Spontaneous networking: an application oriented approach to ad hoc networking. *IEEE Communications Magazine*, 39(6):176--181.
- Friedman, R., Kliot, G., and Avin, C. (2008). Probabilistic quorum systems in wireless ad hoc networks. In *IEEE/IFIP International Conference on Dependable Systems and Networks (DNS '08)*, pages 277--286.
- Group, E. D. C. (2008). Sinalgo - simulator for network algorithms. <http://deg.ethz.ch/projects/sinalgo/>.
- Hamida, E. B. and Chelius, G. (2008). A line-based data dissemination protocol for wireless sensor networks with mobile sink. In *IEEE International Conference on Communications (ICC '2008)*, pages 2201--2205.
- Hartenstein, H. and Laberteaux, K. P. (2008). A tutorial survey on vehicular ad hoc networks. *IEEE Communications Magazine*, 46(6):164--171.
- Helmy, A. (2003). Small worlds in wireless networks. *IEEE Communications Letters*, 7(10):490--492.
- IEEE (2010). Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. IEEE Standards.
- Jain, S., Fall, K., and Patra, R. (2004). Routing in a delay tolerant network. In *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '04)*, pages 145--158.
- Johnson, D. B., Maltz, D. A., and Broch, J. (2001). Ad hoc networking. chapter DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks, pages 139--172.
- Karp, B. and Kung, H. T. (2000). GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *ACM International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 243--254.
- Kiess, W. and Mauve, M. (2007). A survey on real-world implementations of mobile ad-hoc networks. *Ad Hoc Networks*, 5(3):324--339.
- Knorr, F., Baselt, D., Schreckenberg, M., and Mauve, M. (2012). Reducing Traffic Jams via VANETs. *IEEE Transactions on Vehicular Technology*, 61(8):3490--3498.

- Ko, Y. and Vaidya, N. H. (1998). Location-Aided Routing (LAR) in mobile ad hoc networks. In *ACM International Conference on Mobile Computing and Networking (MobiCom '98)*, pages 307--321.
- Korkmaz, G., Ekici, E., and Ozguner, F. (2006). An efficient fully ad-hoc multi-hop broadcast protocol for inter-vehicular communication systems. In *IEEE International Conference on Communications (ICC '06)*, pages 423--428.
- Korkmaz, G., Ekici, E., Ozguner, F., and Ozguner, U. (2004). Urban multi-hop broadcast protocol for inter-vehicle communication systems. In *ACM International Workshop on Vehicular Ad hoc Networks (VANET '04)*, pages 76--85.
- Lee, U. and Gerla, M. (2010). A survey of urban vehicular sensing platforms. *Computer Networks*, 54(4):527--544.
- Li, J. and Mohapatra, P. (2007). Analytical modeling and mitigation techniques for the energy hole problem in sensor networks. *Pervasive Mobile Computing*, 3(3):233--254.
- Lim, H. and Kim, C. (2000). Multicast tree construction and flooding in wireless ad hoc networks. In *ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM '00)*, pages 61--68.
- Lin, K. and Levis, P. (2008). Data Discovery and Dissemination with DIP. In *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '08)*, pages 433--444.
- Liu, A.-F., Wu, X.-Y., Chen, Z.-G., and Gui, W.-H. (2010). Research on the energy hole problem based on unequal cluster-radius for wireless sensor networks. *Computer Communications*, 33(3):302--321.
- Lochert, C., Scheuermann, B., Wewetzer, C., Luebke, A., and Mauve, M. (2008). Data aggregation and roadside unit placement for a vanet traffic information system. In *ACM International Workshop on Vehicular Inter-NETworking (VANET '08)*, pages 58--65.
- Lu, H. and Poellabauer, C. (2011). Analysis of Application-Specific Broadcast Reliability for Vehicle Safety Communications. In *ACM International Workshop on Vehicular Inter-Networking (VANET '11)*, pages 67--72.
- Luo, H., Ye, F., Cheng, J., Lu, S., and Zhang, L. (2005). TTDD: Two-Tier Data Dissemination in Large-Scale Wireless Sensor Networks. *Wireless Networks*, 11(1):161--175.

- Luo, J., Panchard, J., Piórkowski, M., Grossglauser, M., and pierre Hubaux, J. (2006). MobiRoute: Routing towards a mobile sink for improving lifetime in sensor networks. In *IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '06)*, pages 480--497.
- Mainwaring, A., Culler, D., Polastre, J., Szewczyk, R., and Anderson, J. (2002). Wireless sensor networks for habitat monitoring. In *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA '02)*, pages 88--97.
- Marcelloni, F. and Vecchio, M. (2008). A simple algorithm for data compression in wireless sensor networks. *IEEE Communications Letters*, 12(6):411--413.
- Marfia, G., Rocchetti, M., Amoroso, A., and Pau, G. (2013). Safe Driving in LA: Report from the Greatest Intervehicular Accident Detection Test Ever. *IEEE Transactions on Vehicular Technology*, 62(2):1--14.
- Nakamura, E. F., Loureiro, A. A. F., and Frery, A. C. (2007). Information fusion for wireless sensor networks: Methods, models, and classifications. *ACM Computing Surveys*, 39(3):1--55.
- Ni, S.-Y., Tseng, Y.-C., Chen, Y.-S., and Sheu, J.-P. (1999). The broadcast storm problem in a mobile ad hoc network. In *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, pages 151--162.
- Paek, J., Greenstein, B., Gnawali, O., Jang, K.-Y., Joki, A., Vieira, M., Hicks, J., Estrin, D., Govindan, R., and Kohler, E. (2010). The Tenet Architecture for Tiered Sensor Networks. *ACM Transactions on Sensor Networks*, 6(4):1--44.
- Panichpapiboon, S. and Pattara-Atikom, W. (2008). Connectivity requirements for self-organizing traffic information systems. *IEEE Transactions on Vehicular Technology*, 57(6):3333--3340.
- Pattam, S., Krishnamachari, B., and Govindan, R. (2008). The impact of spatial correlation on routing with compression in wireless sensor networks. *ACM Transactions on Sensor Networks*, 4(4):1--24.
- Pazzi, R. W. and Boukerche, A. (2008). Mobile data collector strategy for delay-sensitive applications over wireless sensor networks. *Computer Communications*, 31(5):1028--1039.

- Peng, W. and Lu, X.-C. (2000). On the reduction of broadcast redundancy in mobile ad hoc networks. In *ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc '00)*, pages 129--130.
- Perkins, C. and Royer, E. (1999). Ad-hoc on-demand distance vector routing. In *IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, pages 90--100.
- Ratnasamy, S., Karp, B., Yin, L., Yu, F., Estrin, D., Govindan, R., and Shenker, S. (2002). GHT: A Geographic Hash Table for Data-centric Storage. In *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA '02)*, pages 78--87.
- Ros, F., Ruiz, P., and Stojmenovic, I. (2012). Acknowledgment-Based Broadcast Protocol for Reliable and Efficient Data Dissemination in Vehicular Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, 11(1):33--46.
- Ruj, S., Nayak, A., and Stojmenovic, I. (2013). Pairwise and triple key distribution in wireless sensor networks with applications. *IEEE Transactions on Computers*, 62(11):2224--2237.
- Schwartz, R. S., R. Barbosa, R. R., Meratnia, N., Heijenk, G., and Scholten, H. (2011). A directional data dissemination protocol for vehicular environments. *Computer Communications*, 34(17):2057--2071.
- Shah, R. C., Roy, S., Jain, S., and Brunette., W. (2003). Data MULEs: modeling and analysis of a three-tier architecture for sparse sensor networks. *Elsevier Ad Hoc Networks*, 1(3):215--233.
- Sharma, G. and Mazumdar, R. (2008). A case for hybrid sensor networks. *IEEE/ACM Transactions on Networking*, 16(5):1121--1132.
- Sheng, B., Li, Q., and Mao, W. (2006). Data storage placement in sensor networks. In *ACM International Symposium on Mobile Ad hoc Networking and Computing (MobiHoc '06)*, pages 344--355.
- Sheng, B., Tan, C. C., Li, Q., and Mao, W. (2007). An approximation algorithm for data storage placement in sensor networks. In *International Conference on Wireless Algorithms, Systems and Applications (WASA '07)*, pages 71--78.
- Shenker, S., Ratnasamy, S., Karp, B., Govindan, R., and Estrin, D. (2003). Data-centric storage in sensor networks. *Computer Communication Review*, 33(1):137--142.

- Sommer, C., German, R., and Dressler, F. (2011a). Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. *IEEE Transactions on Mobile Computing*, 10(1):3--15.
- Sommer, C., Joerer, S., and Dressler, F. (2012). On the Applicability of Two-Ray Path Loss Models for Vehicular Network Simulation. In *IEEE Vehicular Networking Conference (VNC '12)*, pages 64--69.
- Sommer, C., Tonguz, O., and Dressler, F. (2011b). Traffic information systems: efficient message dissemination via adaptive beaconing. *IEEE Communications Magazine*, 49(5):173--179.
- Song, L. and Hatzinakos, D. (2007). Architecture of wireless sensor networks with mobile sinks: Sparsely deployed sensors. *IEEE Transactions on Vehicular Technology*, 56(4):1826--1836.
- Spyropoulos, T., Psounis, K., and Raghavendra, C. S. (2008). Efficient routing in intermittently connected mobile networks: the multiple-copy case. *IEEE/ACM Transactions on Networking*, 16(1):77--90.
- Tonguz, O. K., Viriyasitavat, W., and Bai, F. (2009). Modeling urban traffic: a cellular automata approach. *Communications Magazine*, 47(5):142--150.
- Tonguz, O. K., Wisitpongphan, N., and Bai, F. (2010). DV-CAST: a distributed vehicular broadcast protocol for vehicular ad hoc networks. *IEEE Wireless Communications*, 17(2):47--56.
- Toor, Y., Muhlethaler, P., and Laouiti, A. (2008). Vehicle ad hoc networks: applications and related technical issues. *IEEE Communications Surveys Tutorials*, 10(3):74--88.
- Tseng, Y.-C., Ni, S.-Y., and Shih, E.-Y. (2003). Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network. *IEEE Transactions on Computers*, 52(5):545--557.
- Uppoor, S. and Fiore, M. (2011). Large-scale urban vehicular mobility for networking research. In *IEEE Vehicular Networking Conference (VNC '11)*, pages 62--69.
- Uppoor, S. and Fiore, M. (2012). Insights on metropolitan-scale vehicular mobility from a networking perspective. In *ACM International Workshop on Hot Topics in Planet-Scale Measurement (HotPlanet '12)*, pages 39--44.

- Urgaonkar, R. and Krishnamachari, B. (2004). FLOW: An efficient forwarding scheme to mobile sink in wireless sensor networks. In *IEEE International Conference on Sensing, Communication and Networking (SECON '04)*, pages 1--3.
- Vahdat, A. and Becker, D. (2000). Epidemic routing for partially-connected ad hoc networks. Technical report, Duke University.
- Varga, A. and Hornig, R. (2008). An overview of the OMNeT++ simulation environment. In *International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops (Simutools '08)*, pages 1--10.
- Vecchio, M., Viana, A. C., Ziviani, A., and Friedman, R. (2010). DEEP: Density-based proactive data dissemination protocol for wireless sensor networks with uncontrolled sink mobility. *Elsevier Computer Communication*, 33(8):929--939.
- Viana, A. C., Hérault, T., Largillier, T., Peyronnet, S., and Zaïdi, F. (2010). Supple: a flexible probabilistic data dissemination protocol for wireless sensor networks. In *ACM International Conference on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM '10)*, pages 385--392.
- Viana, A. C., Ziviani, A., and Friedman, R. (2009). Decoupling data dissemination from mobile sink's trajectory in wireless sensor networks. *IEEE Communications Letters*, 13(3):178--180.
- Viriyasitavat, W., Bai, F., and Tonguz, O. (2010). UV-CAST: An urban vehicular broadcast protocol. In *IEEE Vehicular Networking Conference (VNC '10)*, pages 25--32.
- Vuran, M. C., Akan, O. B., and Akyildiz, I. F. (2004). Spatio-temporal correlation: theory and applications for wireless sensor networks. *Computer Networks*, 45(3):245--259.
- Whitehouse, K., Tolle, G., Taneja, J., Sharp, C., Kim, S., Jeong, J., Hui, J., Dutta, P., and Culler, D. (2006). Marionette: using RPC for interactive development and debugging of wireless embedded networks. In *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '06)*, pages 416--423.
- Williams, B. and Camp, T. (2002). Comparison of broadcasting techniques for mobile ad hoc networks. In *ACM International Symposium on Mobile Ad hoc Networking & Computing (MobiHoc '02)*, pages 194--205.

- Wisitpongphan, N., Tonguz, O., Parikh, J., Mudalige, P., Bai, F., and Sadekar, V. (2007). Broadcast storm mitigation techniques in vehicular ad hoc networks. *IEEE Wireless Communications*, 14(6):84--94.
- Wu, X., Chen, G., and Das, S. (2008). Avoiding energy holes in wireless sensor networks with nonuniform node distribution. *IEEE Transactions on Parallel and Distributed Systems*, 19(5):710--720.
- Yang, H., Ye, F., and Sikdar, B. (2006). SIMPLE: Using Swarm Intelligence Methodology to Design Data Acquisition Protocol in Sensor Networks with Mobile Sinks. In *IEEE International Conference on Computer Communications (INFOCOM '06)*, pages 1--12.
- Yi, C.-W., Chuang, Y.-T., Yeh, H.-H., Tseng, Y.-C., and Liu, P.-C. (2010). Streetcast: An Urban Broadcast Protocol for Vehicular Ad-Hoc Networks. In *IEEE Vehicular Technology Conference (VTC '10)*, pages 1--5.
- Yick, J., Mukherjee, B., and Ghosal, D. (2008). Wireless sensor network survey. *Computer Networks*, 52(12):2292--2330.
- Zhang, X. and Jacob, L. (2003). Adapting zone routing protocol for heterogeneous scenarios in ad hoc networks. In *International Conference on Parallel Processing (ICPP '03)*, pages 341--348.
- Zhang, Y. and Lee, W. (2000). Intrusion detection in wireless ad-hoc networks. In *ACM International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 275--283.
- Zhao, J., Zhang, Y., and Cao, G. (2007). Data Pouring and Buffering on The Road: A New Data Dissemination Paradigm for Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, 56(6):3266--3277.
- Zussman, G. and Segall, A. (2003). Energy efficient routing in ad hoc disaster recovery networks. *Ad Hoc Networks*, 1(4):405--421.