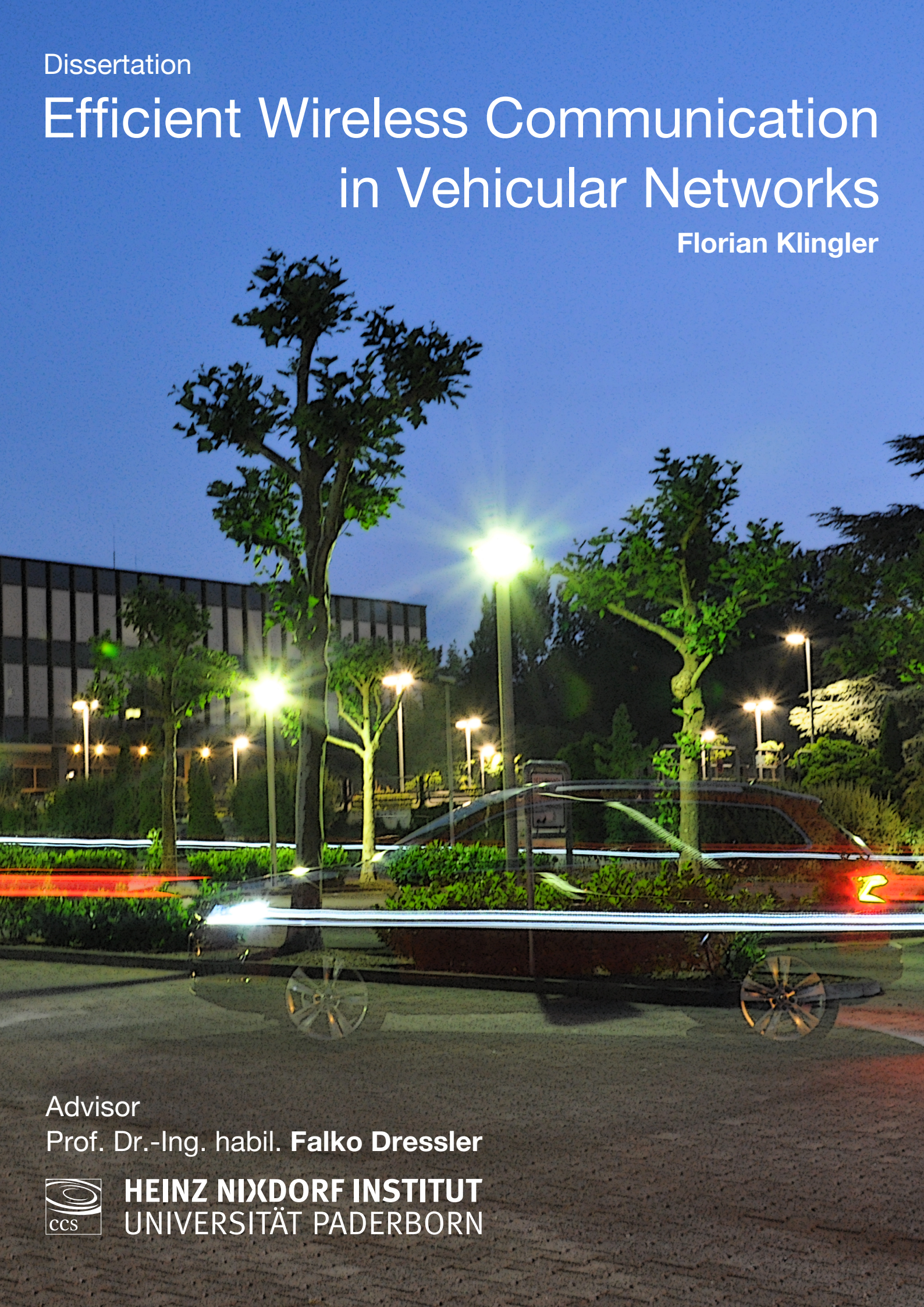
Dissertation

Efficient Wireless Communication in Vehicular Networks

Florian Klingler



Advisor
Prof. Dr.-Ing. habil. **Falko Dressler**

HEINZ NIXDORF INSTITUT
UNIVERSITÄT PADERBORN

HEINZ NIXDORF INSTITUT
UNIVERSITÄT PADERBORN

Florian Klingler

Efficient Wireless Communication in Vehicular Networks

Dissertation

July 2018

Please cite as:
Florian Klingler, “Efficient Wireless Communication in Vehicular Networks,” PhD Thesis (Dissertation), Heinz Nixdorf Institute, Paderborn University, Germany, September 2018.

Distributed Embedded Systems (CCS Labs)
Heinz Nixdorf Institute, Paderborn University, Germany

Fürstenallee 11 · 33102 Paderborn · Germany

http://www.ccs-labs.org/

Efficient Wireless Communication in Vehicular Networks

Dissertation
zur Erlangung des Grades

DOKTOR DER NATURWISSENSCHAFTEN

vorgelegt von

Florian Klingler

geb. am 28. Dezember 1985
in Hall in Tirol, Österreich

angefertigt in der Fachgruppe

Distributed Embedded Systems
(CCS Labs)

Heinz Nixdorf Institut
Universität Paderborn

Betreuer:	**Prof. Dr.-Ing. habil. Falko Dressler**
Gutachter:	**Prof. Dr.-Ing. habil. Falko Dressler**
	Prof. Dr. Jiannong Cao
	Prof. Dr.-Ing. Lars Wolf
Tag der Abgabe:	**26. Juli 2018**
Tag der Promotion:	**17. September 2018**

Abstract

Wireless communication among vehicles has been shown to be beneficial for a variety of use cases in the automotive domain ranging from pure safety to traffic efficiency and to entertainment applications. To accomplish communication, different protocol stacks have been standardized around the world, e.g., ETSI ITS-G5 in Europe and IEEE 1609 WAVE in the U.S., both building upon IEEE 802.11p WLAN, yet for many applications, efficiency is still a problem. We thus begin this PhD thesis with an analytical investigation of the capacity bounds of IEEE 802.11p. As a first contribution towards efficient wireless communication, we study the performance of IEEE 802.11p based unicast communication, which is, e.g., used by the ETSI ITS-G5 GeoNetworking specification. Our investigations reveal that unicast communication employing retransmissions at the MAC layer is not only not beneficial in vehicular communications, but maybe harmful in typical scenarios, as it leads to higher communication delays. Based on our findings and current limitations of ETSI ITS-G5, we present as a second contribution a purely broadcast based networking architecture, which categorizes communication demands of applications into four distinct classes. A central building block of our network layer is the support of 2-hop neighbor information using space efficient Bloom filters to provide nodes a better overview of their vicinity. In our third contribution, we take a detailed look on how to properly maintain this neighbor information and propose Bloom Hopping, a 2-hop message dissemination protocol, which operates independently from the road topology. Simulation results show that it can outperform traditional 2-hop approaches (not using Bloom filters) in terms of requiring less channel resources and providing better application performance. As a fourth contribution, we focus on the scalability of vehicular communication by taking advantage of multi-channel operation similar to what has been proposed in IEEE 1609.4 WAVE. In particular, we design a set of scheduling algorithms that answer the question when to send which information on which channel. Results reveal that our system has lower channel resource requirements and provides better application layer performance in comparison to single-channel protocols. As a summary, we believe the work presented in this PhD thesis brings vehicular communication forward in research and one step closer to the road.

Kurzfassung

Die Verwendung von Funkkommunikation zum Austausch von Informationen zwischen Fahrzeugen, um die Sicherheit und Verkehrseffizienz zu erhöhen, hat sich als vorteilhaft erwiesen. In der Vergangenheit wurden dafür weltweit verschiedene Standards entworfen, z. B. ETSI ITS-G5 in Europa, oder IEEE 1609 WAVE in den USA; beide basieren auf IEEE 802.11p WLAN. Trotzdem stellt effiziente Kommunikation noch ein Problem für viele Anwendungen dar. Wir beginnen daher diese Dissertation mit einer analytischen Betrachtung der Leistungsfähigkeit von IEEE 802.11p. Als ersten Beitrag untersuchen wir die Effizienz von IEEE 802.11p basierter Unicast Kommunikation in hoch mobilen Szenarien. Diese Art der Übertragung wird in der ETSI ITS-G5 GeoNetworking Spezifikation als ein zentrales Kommunikationsparadigma verwendet. Unsere Ergebnisse zeigen, dass Unicast Kommunikation keinen Mehrwert in typischen Szenarien bietet und teils höhere Kommunikationslatenzen verursachen kann. Basierend auf diesen Ergebnissen und Einschränkungen des aktuellen ETSI ITS-G5 Standards entwickeln wir als zweiten Beitrag dieser Arbeit eine ausschließlich Broadcast basierte Netzwerkarchitektur, welche vier verschiedene Kommunikationsparadigmen unterstützt. Ein zentraler Bestandteil dieser Architektur ist die Verwendung von 2-hop-Nachbarschaftsinformationen mittels Bloom filter, um Fahrzeugen eine bessere Übersicht der Netzwerktopologie zu ermöglichen. In unserem dritten Beitrag werfen wir einen detaillierten Blick auf diese Nachbarschaften und entwickeln einen Algorithmus, um unabhängig der Straßentopologie 2-hop-Nachbarn zu informieren. Im Gegensatz zu herkömmlichen Ansätzen ohne Bloom filter können wir damit Kanallast einsparen und gleichzeitig die Anzahl der informierten Fahrzeuge erhöhen. Im vierten Beitrag dieser Arbeit widmen wir uns der Skalierbarkeit von Fahrzeugkommunikation mittels mehrerer Funkkanäle, ähnlich zu IEEE 1609.4 WAVE. Im Detail entwickeln wir Algorithmen um die Fragen, wann welche Information auf welchem Kanal gesendet werden soll, zu beantworten. Wir können mit unserem Ansatz Kanallast verringern und dabei eine höhere Anzahl von Knoten im Netzwerk erreichen. Die entwickelten Ansätze in dieser Dissertation bringen die Forschung im Bereich Fahrzeugkommunikation einen Schritt weiter und können helfen, zukünftige Systeme in diesem Bereich zu verbessern.

Contents

Chapter 1

Introduction and Motivation

ACCORDING to the World Health Organization (WHO) global status report on road safety [1], each year over 1.2 million fatal traffic accidents happen on roads world-wide. Moreover, in 2012 road traffic injuries have been the main death causes of people aged 15-29 years. An important conclusion of the report is that (besides the need to strengthen road safety laws especially in low- and medium-income countries) vehicle safety is a critical component of saving lives on roads, for which cars sold in many countries around the world still fail – leading to a huge room for improvement.

Safety systems available in today's vehicles aim to either lower the impact of a crash, e.g., airbags, or try to avoid crashes altogether, e.g., emergency braking systems. These technologies take their decisions mainly based on data provided by local sensors, like radar, lidar, camera based systems, or ultrasonic sensors. However, often it is not sufficient to only rely on data provided by local sensors, since most of the time this data is only available when the event in question already happened.

Let us consider a freeway scenario as outlined in Figure 1.1, where a vehicle uses adaptive cruise control to regulate its speed automatically to maintain a constant distance to the vehicle in front based on data provided by the radar sensor. Instead of relying only on this local sensor information to reduce the speed when another

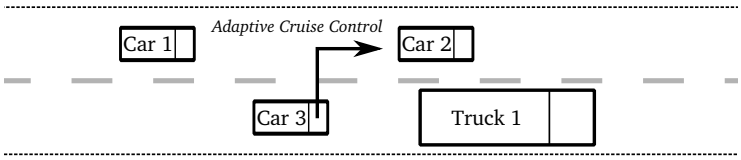


Figure 1.1 – Example scenario of four vehicles on a freeway: Car 1 uses adaptive cruise control to maintain a constant distance to Car 2 based on radar sensor data. Car 3 starts to overtake the truck and squeezes itself between the two cars.

vehicle starts to overtake and cuts into the other vehicles' lane, wouldn't it be much better to get this information beforehand, namely at the time the driver intends to begin an overtake maneuver – and even better being able to inform a driver to cancel this maneuver in case it would cause a dangerous situation?

Surely we can argue that traditional signaling, like the use of the turn signal, can indicate such an overtake process. However, sometimes no direct Line of Sight (LOS) between drivers is available (as shown in Figure 1.2) making indication of such events difficult to nearly impossible. Here, a vehicle drives between two trucks on the same lane, and a second vehicle from behind (having a higher speed) approaches the truck and starts overtaking while at the same time the car starts overtaking. This dangerous situation could be avoided if the two vehicles in question have the possibility to coordinate with each other by making them aware of upcoming overtake maneuvers.

The above example can easily be extended to urban and rural areas where obstacles (e.g., buildings) can negatively affect light based signaling between drivers of vehicles too. Here an example could be two cars approaching an uncontrolled intersection having a building beside – the two cars have no possibility to see each other. In such cases it would be beneficial if a vehicle has the ability to *look* through this building to see whether other cars approach the intersection too, thus have a chance to reduce or even increase the driving speed to avoid a potential crash [2].

In all these cases, wireless communication could be beneficial to allow exchange of information (e.g., lane change, intention to turn, or simply cooperative awareness) between vehicles to solve above problems.

Whenever wireless communication is deployed among vehicles, it could be also used to offer more sophisticated services like road traffic congestion information, or green light speed advisories in urban areas incorporating traffic lights.

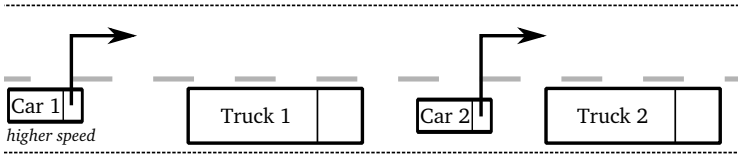


Figure 1.2 – Example scenario of a dangerous situation employing four vehicles on a freeway: Car 1 approaches Truck 1 with a higher speed and starts to overtake; at the same time Car 2 starts to overtake Truck 2.

1.1 Wireless Communication as a Promising Solution

Wireless signals have the possibility to penetrate obstacles (e.g., buildings, vehicles), therefore taking advantage of their properties could be helpful to support information dissemination among vehicles.

In the past, many approaches have been proposed to apply wireless communication to the automotive domain, which led to the evolution of a new research field – vehicular communications, being termed Vehicle-to-Vehicle (V2V) and Vehicle-to-X (V2X) communications. A combination of these schemes not only allows to be used for safety-related applications, but also allows to build Intelligent Transportation Systems (ITS), which take advantage of infrastructure support for, e.g., toll collection, or traffic management.

One possibility to achieve information exchange among vehicles are Vehicular Ad Hoc Networks (VANETs) in which nodes form a distributed network by using wireless communications. Research in this area evolved from theoretical studies of communication protocols and technologies [3], [4] to simulation studies [5] and field tests [6] for protocol development in terms of various application domains. We can even provide Open Source prototypical implementations incorporating common standards for inter-vehicular communication nowadays [7].

For many applications in the automotive domain, vehicular communications has been found beneficial ranging from pure safety purposes up to road traffic efficiency and to comfort applications. Sample applications outlined by Karagiannis et al. [8] and Sommer and Dressler [9] are: intersection collision warning, wrong way driver warning, vehicle platooning, or traffic information services.

Two different communication paradigms for inter-vehicular communications evolved in the past which can be roughly categorized in *(a)* cellular based communication leading to the C-V2X standard included in 3GPP Release 14 [10]–[12], and *(b)* ad-hoc communication leading to the ETSI ITS-G5 [13] standard in Europe, the IEEE 1609 [14]–[17] set of standards in U.S., and the ARIB STD-T109 [18] standard in Japan. However, these standards currently have several difficulties to fulfill safety requirements for road traffic:

Cellular based communication protocols mostly rely on infrastructure, which could be a problem in rural areas since not everywhere base stations are deployed or can be deployed in the future. Although many cellular based protocols have the ability to also communicate in an ad-hoc fashion (like C-V2X), their full performance can only be achieved when infrastructure is present.

Looking at protocols designed for pure ad-hoc communication, many applications are currently built to be completely independent from each other [19]–[21]. This could work if there is only a single application, e.g., intersection collision avoidance, however when wireless communication will be deployed in vehicles, many different

applications most likely will operate at the same time, e.g., cooperative awareness, vehicle platooning, a green light optimal speed advisory. At this point, applications need to cooperate among each other or at least be aware about another application's presence to gain best performance, avoid overloading the wireless channel, and deal with the fact that they can influence each other.

In this PhD thesis we focus on pure ad-hoc communication based on IEEE 802.11 WLAN [22] and its amendment IEEE 802.11p, which was designed for vehicular communications to cope with the high mobility dynamics of vehicles.

1.2 Why Current Approaches Are Not Sufficient

As outlined, currently many specific applications and their corresponding protocol stacks are built in a way such that they are independent from each other.

In Figure 1.3 we illustrate such a scenario incorporating multiple applications and protocol stacks, where each of them has a dedicated radio and wireless channel. For many proposals of application specific communication protocols in the literature this is the main approach [19]–[21]. Each vehicle needs as many radios as channels (or protocol stacks) are available, which leads to high hardware costs.

Moreover, by statically assigning complete protocol stacks to wireless channels, it can happen that the wireless spectrum is not utilized evenly leading to overutilization, and thus by consequence unusable wireless channels.

However, since there are only a few non-overlapping 10 MHz channels in the 5.9 GHz band available for ITS [23], it is evident that this approach does not scale

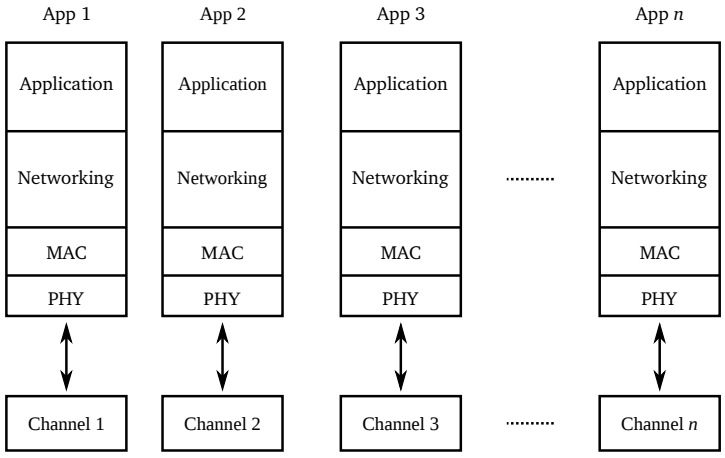


Figure 1.3 – Approach *1*: All applications and their corresponding networking stacks use a separate wireless channel and radio.

well and easily leads to problems when multiple protocol stacks need to be operated concurrently.

In this case, some applications need to share the same wireless channel and radio (employing the Media Access Control (MAC) and Physical layer (PHY)) as depicted in Figure 1.4. This could introduce influences among the protocol stacks operating on the same radio device [24]. Here the upper layers of the protocol stacks are not aware of each other and thus can suffer from performance degradation, and/or overloaded wireless channels if each application assumes to be the only one operating on the network. Even worse, protocol stacks can actively use such a scenario to perform denial of service of other protocol stacks operating on the same MAC and PHY, by either sending more data than the wireless channel can handle, or by using communication paradigms shown to be harmful [25].

Another possibility to allow multiple applications to operate on the same wireless channel is to share a common Network layer as we outline in Figure 1.5 and what is very similar to the current ETSI ITS-G5 protocol stack. All applications again share a common wireless channel, and the use of a second wireless channel (and eventually radio) makes it necessary to provide either channel switching techniques [17], [26], or a complete second protocol stack for this additional wireless channel. To this end, often many decisions (e.g., channel selection, to whom to communicate) are still part of higher layer protocols, in particular the application logic.

Moreover, the current idea of ETSI ITS-G5 is to generically limit wireless traffic of higher layer protocols (i.e., applications) by means of the Decentralized Congestion Control (DCC) [13] module, but without taking into account individual application

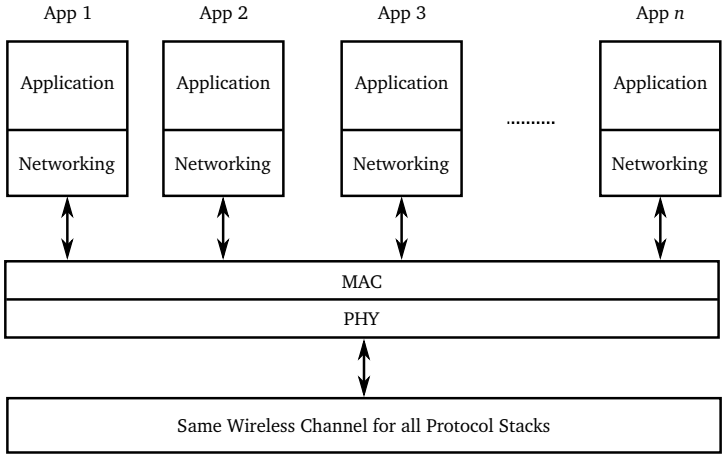


Figure 1.4 – Approach *2*: All applications and their corresponding networking stacks share the same wireless channel and radio.

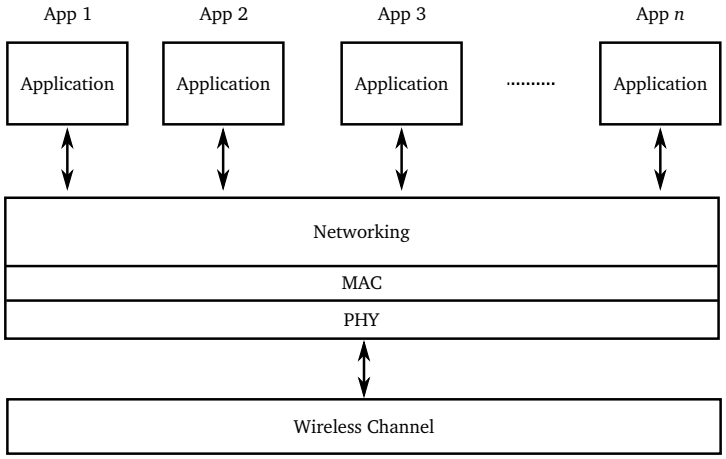


Figure 1.5 – Approach *3*: All applications build upon a common networking stack, similar to what ETSI ITS-G5 is proposing.

requirements. However, such a separation is needed, as otherwise decisions of the rate limiting algorithms could negatively impact application performance.

Beside periodic information sent as 1-hop Cooperative Awareness Messages (CAMs) and event based information sent as Decentralized Environmental Notification Messages (DENMs), the current ETSI ITS-G5 standard also provides GeoNetworking functionality [27], which aims to forward messages towards a specific geographic region. Here, communication is based on IEEE 802.11 unicast employing automatic retransmission at the MAC layer for not acknowledged frames, which could negatively impact wireless communication performance [25].

We believe that the current approaches for vehicular communications need to be improved by providing a generalized networking concept which separates networking decisions from application logic, and provides novel scheduling algorithms for efficient channel use in single-radio multi-channel networks. In particular we believe that 2-hop neighbor management (in contrast to only 1-hop neighbor management as available by ETSI ITS-G5) will bring a whole set of new possibilities for wireless network communication in the automotive domain.

1.3 Problem Description and Research Questions

Based on current standardization efforts for Inter-Vehicle Communication (IVC), we believe that the proposed networking architecture of ETSI ITS-G5 is not sufficient to support a wide range of different applications and their communication principles. In particular the GeoNetworking approach employing GeoBroadcast [27] takes

advantage of only single-hop neighbor information which does not allow to derive high quality measures about a node's vicinity.

One example could be the very common (hidden terminal) scenario of an intersection with buildings at each of the four corners as shown in Figure 1.6. When vehicles now broadcast information about their presence to achieve cooperative awareness, nodes located on streets between two buildings will observe a much lower channel load, and by consequence interference, than nodes staying in the middle of the intersection. This is due to attenuation of the wireless signals by the buildings. It could happen that the vehicle at the intersection (Car A) cannot correctly receive information from the vehicle staying between two buildings (Car B) due to interference, because this node is affected by transmissions of nodes present on all four streets. At the same time, the vehicle between the two buildings (Car B), however, can receive information from Car A and may wrongly consider this node as a usable neighbor which could be a promising candidate for message forwarding. Wrongly in that sense, as this node (Car A) most probably will not be able to decode information received from the originating node (Car B) for message dissemination.

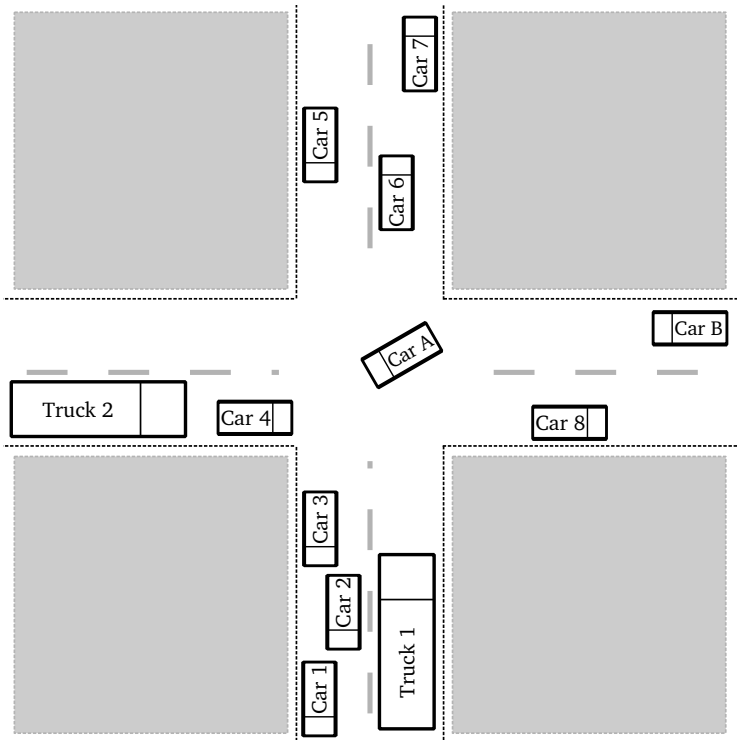


Figure 1.6 – Example scenario consisting of an intersection with four buildings: A vehicle in the middle of the intersection (Car A) will observe a higher channel load (and by consequence a higher probability of lost frames) than a vehicle located between two buildings (Car B).

Information about 2-hop neighbors, i.e., neighbors of neighbors, would allow a much better overview about the network topology of a VANET, e.g., giving a node the ability to check whether a symmetric communication channel between neighbors exist. This way it would be possible to provide efficient message forwarding algorithms which aim to reach all 2-hop neighbors, e.g., as it would be beneficial for an intersection collision avoidance application.

The proposed Contention-based Forwarding (CBF) mechanism available in the GeoNetworking approach [27] needs as input parameter the theoretical maximum communication distance as a fundamental requirement for protocol operation. This is necessary for a receiving node to determine whether to forward a message based on its distance to the origin node. However, this theoretical communication distance is – as the name states – only theoretical and influenced by many different factors, e.g., channel load, interference, transmission power, obstacles. Moreover, the achievable communication distance can quickly change due to the inherent mobility of vehicles. Based on the operation of the CBF algorithm, where the retransmission time scales proportionally to the distance between receiver and sender, it could happen that other receiving nodes do not hear the retransmission of a particular node due to obstructions. In that case, they would not cancel their retransmission, which could lead to synchronized retransmission – and thus by consequence to packet collisions. Further, if nodes are very near to each other and have the same distance to the origin node, they would use the same retransmission delay – again leading to collisions. Using a different (namely sender-based) forwarder selection would be more beneficial in such cases.

Greedy forwarding mostly relies on IEEE 802.11p unicast communication which has been shown to be harmful in vehicular scenarios [25], [28]. Retransmissions caused by IEEE 802.11p unicast at MAC layer degrades also the protocol operation when using Greedy forwarding together with CBF, in which retransmission at the MAC layer compete with retransmission by the network layer [29]. A scheme providing GeoNetworking not dependent on unicast communication could be more beneficial.

To provide scalability of vehicular communication in dense networks, it could be beneficial to take advantage of multiple wireless channels for message dissemination. For single-radio networks the IEEE 1609 [16], [17] set of standards provide the possibility to use multiple wireless channels in a split-phase approach. The open issue here is to provide adequate algorithms for channel selection, coordination and prioritization of messages in order to take full advantage of the additional spectrum provided by multi-channel operation.

1.3.1 Research Questions

In this thesis we focus on the following important research questions which allow us to develop new approaches to solve the identified problems from above:

The transmission of unicast frames of IEEE 802.11 provides reliability by relying on acknowledgments. Whenever those are not received, retransmissions are invoked up to a configurable limit. To what degree is this retransmission scheme providing reliability in highly dynamic networks? If at all, does it increase application layer performance? And if not, is it harmful? In the case of not being beneficial, are there possibilities to improve MAC layer mechanisms to provide better reliability?

Building standardized networking protocols to provide functionality for a wide range of possible applications is a non-trivial task. In order to do so it is necessary to define separate classes representing possible communication paradigms for applications. How can applications be categorized into different classes, and which criteria should be applied to perform this? Which requirements on network communication do each of the classes have, and how can those be fulfilled? Which candidate protocols for each of the classes could be beneficial to perform message dissemination among nodes? How does the protocol performance scale for different vehicle-densities and message generation rates? To what degree do candidate protocols influence each other when being operated concurrently on a wireless channel?

Focusing on candidate protocols for message dissemination, one possibility is to investigate sender-based selection of forwarding nodes based on information from neighbor tables. Interesting research questions in this field are: When should a node be included in the neighbor table? When should a node be pruned from the neighbor table? How does a node make sure that a symmetric link is available between itself and a potential neighbor? Which nodes from a set of 1-hop neighbors should be considered as forwarding nodes in order to maximize the reachability of all 2-hop neighbors? What kind of influence do protocol parameters, vehicle density, or neighbor update rate have on the quality of neighbor tables and the performance of neighbor selection for message dissemination? And most importantly: How to address the question of what is a neighbor (or what is not) in the presence of an (unreliable) wireless connection? Finding here the ground truth to compare protocol operation against to is the main challenge.

Moving back to MAC layer mechanisms, the investigation of multi-channel protocols for vehicular communications in single-radio environments could be beneficial to provide scalability when vehicle density is high. Here, the main research questions are: When to send an announcement for a specific channel to not overload the wireless network? Consequently, if we decided to transmit an announcement, which channel to announce, and thus, later switch to? How can the above question be

answered when an additional requirement is necessary: Prioritization of messages? And most importantly: What is the impact in terms of delay and application layer performance when multiple channels are used by a single-radio system. Is it possible to outperform single-channel systems in all three dimensions, namely channel load, delay, and application layer performance (in that case, reliability)?

1.4 Summarizing Our Contributions

Having studied the main research questions we address in this thesis, we now focus on the main contributions in this work.

As a first contribution of this PhD thesis we show that IEEE 802.11 unicast communication is not feasible in vehicular networks due to high vehicle dynamics and the inherent frame retransmission scheme with exponential backoff of IEEE 802.11. We show that the original idea of having communication by using retransmissions according to IEEE 802.11 unicast is not only not practicable, but also harmful as it degrades the overall networking performance. This serves as fundamental motivation for broadcast based protocol design.

We take advantage of this finding and provide as a second contribution of this PhD thesis an alternative and purely broadcast based networking architecture which allows multiple applications to co-exist on the same network. In this holistic networking concept for IVC we classify applications into four distinct categories and implement candidate protocols which provide message forwarding algorithms according to the different needs of applications.

As a third contribution, we propose a multi-hop message dissemination protocol called Bloom hopping which, in contrast to the GeoUnicast approach of ETSI ITS-G5, operates based on 2-hop neighbor information instead of geographic positions, and employs pure broadcast-based communication. By using space efficient Bloom filter data structures, we can keep the channel utilization at low values and provide 2-hop neighbor information which allows candidate protocols of other classes to build their forwarding decisions on.

As a fourth contribution of this PhD thesis, we propose a design for multi-channel scheduling algorithms which allow efficient utilization of the radio spectrum dedicated for inter-vehicle communication in single-radio environments leading to increased performance in comparison to single-radio single-channel protocols.

1.5 Structure of the Thesis

The remainder of this PhD thesis is organized as follows:

In Chapter 2 we present fundamentals and related work in the field of vehicular communication and outline important milestones in this research field. In detail, we give an overview of different approaches in the area of VANET communication protocols based on IEEE 802.11 WLAN up to latest developments and standardization for ITS. As the focus of this PhD thesis is on efficient wireless communication based on IEEE 802.11 WLAN ranging from the application layer down to the MAC, cellular based communication technologies like 3G/UMTS, LTE, C-V2X, and very recently 5G technologies are outside the scope of this thesis.

In Chapter 3 we study the performance of IEEE 802.11 unicast communication in scenarios with high mobility which will serve as a motivation to use a pure broadcast based protocol design. Since reliable communication is an important aspect in vehicular communication, we focus on how to transfer information between two nodes in a reliable manner. Reliability in our case not only defines the fact that a message needs to be successfully received, but also incorporates the time it takes to get this message transferred. The IEEE 802.11 WLAN standard – thus also IEEE 802.11p – already provides a method to retransmit frames which have not been received successfully. As in the literature many networking protocols rely on this technique, we investigate this topic in detail in terms of analytical calculations, simulations and real world experiments, and reveal that the frame retransmission scheme provided by IEEE 802.11 is not feasible to be used in highly dynamic networks like VANETs. In particular, lost frames employing the aforementioned retransmission scheme increase end-to-end delays of communicating nodes from around 2 ms up to 200 ms – per frame.

In Chapter 4 we present a novel and holistic Network layer architecture for VANETs which allows to separate application logic and network control (how to forward messages) in a way to support communication paradigms needed by past and future applications. We provide an overview of requirements of different applications with respect to their communication patterns and outline example algorithms to allow efficient message forwarding independent of the underlying road topology. In particular, we study the influence of concurrent operation of different classes of communication protocols on each other when they operate on the same wireless channel at the same time. With results from these studies we develop a novel multi-hop message forwarding algorithm which takes the probabilistic nature of Bloom filters into account to drastically lower the size of packets needed to be transmitted over the wireless network. This approach – we call it Bloom Hopping – aims to select a good set of 1-hop neighbors to forward a message in order to reach most or all 2-hop neighbors.

To achieve this goal, in Chapter 5 we propose a sophisticated neighbor table algorithm which allows to efficiently maintain 2-hop neighbor information in a vehicular scenario. This neighbor table not only provides information for our Bloom

hopping algorithm, but can be used by all four classes of broadcast protocols in our Network layer architecture.

Since efficient vehicular communication is not only bound to efficient algorithms for message forwarding, we focus in Chapter 6 on how to take advantage of multiple wireless channels as available with IEEE 802.11p. Here we study the feasibility of how to design multi-channel scheduling systems in order to allow channel assignment and synchronization in highly dynamic networks like VANETs to gain lower individual channel utilization, and better networking performance, both at the same time. With MCB we present a multi-channel beaconing protocol taking advantage of a split phase approach similar to what IEEE 1609.4 proposes, where a single-radio system constantly switches every 50 ms between a Control Channel (CCH) and one of multiple available Service Channels (SCHs). Results show that it is not only feasible to use multiple channels for vehicular communication, but it also helps to significantly reduce channel congestion when traffic density is high.

Finally, in Chapter 7 we conclude the need for efficient wireless communication in vehicular scenarios and how the presented approaches help to bring VANETs a step closer to the road.

1.6 Publications

This section lists the publications which I have worked on during my PhD thesis.

1.6.1 Papers This Thesis Is Based On

1. F. Klingler, F. Dressler, and C. Sommer, "IEEE 802.11p Unicast Considered Harmful," in *7th IEEE Vehicular Networking Conference (VNC 2015)*, Kyoto, Japan: IEEE, Dec. 2015, pp. 76–83, © 2015 IEEE.
   My contribution in this conference publication was the extension of the network simulator to allow unicast communication and evaluation of lost acknowledgments in static and highly dynamic scenarios. Finally I evaluated the performance of IEEE 802.11 unicast communication in terms of a larger simulation study investigating the impact on a Geocasting protocol.

2. F. Klingler, F. Dressler, and C. Sommer, "The Impact of Head of Line Blocking in Highly Dynamic WLANs," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 8, pp. 7664–7676, Aug. 2018, © 2018 IEEE.
   My contribution in this journal article was the design, implementation and evaluation of test cases to trigger lost acknowledgments in static scenarios for both, simulation and real world experiments. Further I designed and conducted experiments to actively jam (interfere) acknowledgments in order to study the

behavior of commodity WLAN hardware in situations when acknowledgments were not received. For this I had to extend an existing system for generating interfering frames in a way to cope with the short timings necessary to interfere acknowledgment frames. Moreover I performed analytical evaluations and numerical simulations to validate my experiments. Finally I studied the impact of unicast communication on realistic scenarios in the VANET context in terms of simulation studies.

3. F. Klingler, "Context-Aware and Class-Based Broadcasting in VANETs," in *International Conference on Networked Systems (NetSys 2015), PhD Forum*, Cottbus, Germany, Mar. 2015.
   My contribution in this Regional Workshop paper was the outline of important metrics to investigate a holistic networking concept for IVC and a general outline of further topics to investigate in my PhD thesis.

4. F. Dressler, F. Klingler, C. Sommer, and R. Cohen, "Not All VANET Broadcasts Are the Same: Context-Aware Class Based Broadcast," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 17–30, Feb. 2018, © 2018 IEEE.
   My contribution in this work was the design of candidate protocols and the implementation of the class based networking architecture as well as the analysis and evaluation of different networking protocols.

5. F. Klingler, R. Cohen, C. Sommer, and F. Dressler, "Bloom Hopping: Bloom filter based 2-Hop Neighbor Management in VANETs," *IEEE Transactions on Mobile Computing*, 2018, available online, © 2018 IEEE.
   My contribution in this work was the design and the implementation of the 2-hop neighbor table algorithm as well as the analysis and evaluation of it in different scenarios.

6. F. Klingler, F. Dressler, J. Cao, and C. Sommer, "Use Both Lanes: Multi-Channel Beaconing for Message Dissemination in Vehicular Networks," in *10th IEEE/IFIP Conference on Wireless On demand Network Systems and Services (WONS 2013)*, Banff, Canada: IEEE, Mar. 2013, pp. 162–169, © 2013 IEEE.
   My contribution in this conference publication was to study the feasibility of multi-channel beaconing in vehicular networks based on IEEE 1609.4 split-phase channel switching.

7. F. Klingler, "Improving Multi-Channel Beaconing in Vehicular Networks," in *3rd GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*, Ulm, Germany, Mar. 2015.
   My contribution in this Regional Workshop paper was the design and evaluation of first results of a multi-channel beaconing approach in comparison to a single-channel protocol.

8. F. Klingler, F. Dressler, J. Cao, and C. Sommer, "MCB - A Multi-Channel Beaconing Protocol," *Elsevier Ad Hoc Networks*, vol. 36, no. 1, pp. 258–269, Jan. 2016, © 2016 Elsevier B.V.
   My contribution in this journal article was the design, implementation and evaluation of the multi-channel message dissemination algorithm.

1.6.2 Additional Papers

1. F. Dressler, F. Klingler, M. Segata, and R. Lo Cigno, "Cooperative Driving and the Tactile Internet," *Proceedings of the IEEE*, 2018, to appear.

2. I. Turcanu, F. Klingler, C. Sommer, A. Baiocchi, and F. Dressler, "Duplicate Suppression for Efficient Floating Car Data Collection in Heterogeneous LTE-DSRC Vehicular Networks," *Elsevier Computer Communications*, vol. 123, pp. 54–64, Jun. 2018.

3. J. Heinovski, F. Klingler, F. Dressler, and C. Sommer, "A Simulative Analysis of the Performance of IEEE 802.11p and ARIB STD-T109," *Elsevier Computer Communications*, vol. 122, pp. 84–92, Jun. 2018.

4. F. Klingler, J. Blobel, and F. Dressler, "Agriculture meets IEEE 802.11p: A Feasibility Study," in *15th IEEE International Symposium on Wireless Communication Systems (ISWCS 2018)*, Lisbon, Portugal: IEEE, Aug. 2018.

5. S. Loewen, F. Klingler, C. Sommer, and F. Dressler, "Backwards Compatible Extension of CAMs/DENMs for Improved Bike Safety on the Road," in *9th IEEE Vehicular Networking Conference (VNC 2017), Poster Session*, Torino, Italy: IEEE, Nov. 2017, pp. 43–44.

6. G. S. Pannu, F. Klingler, C. Sommer, and F. Dressler, "QQDCA: Adapting IEEE 802.11 EDCA for Unicast Transmissions at High Topology Dynamics," in *9th IEEE Vehicular Networking Conference (VNC 2017)*, Torino, Italy: IEEE, Nov. 2017, pp. 295–302.

7. B. Bloessl, F. Klingler, F. Missbrenner, and C. Sommer, "A Systematic Study on the Impact of Noise and OFDM Interference on IEEE 802.11p," in *9th IEEE Vehicular Networking Conference (VNC 2017)*, Torino, Italy: IEEE, Nov. 2017, pp. 287–290.

8. F. Klingler, G. S. Pannu, C. Sommer, and F. Dressler, "Connecting Simulation and Real World: IEEE 802.11p in the Loop," in *23rd ACM International Conference on Mobile Computing and Networking (MobiCom 2017), Poster Session*, Snowbird, UT: ACM, Oct. 2017, pp. 561–563.

9. F. Klingler, G. S. Pannu, C. Sommer, B. Bloessl, and F. Dressler, “Field Testing Vehicular Networks using OpenC2X,” in *15th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys 2017), Poster Session*, Niagara Falls, NY: ACM, Jun. 2017, pp. 178–178.

10. S. Laux, G. S. Pannu, S. Schneider, J. Tiemann, F. Klingler, C. Sommer, and F. Dressler, “OpenC2X - An Open Source Experimental and Prototyping Platform Supporting ETSI ITS-G5,” in *8th IEEE Vehicular Networking Conference (VNC 2016), Demo Session*, Columbus, OH: IEEE, Dec. 2016, pp. 152–153.

11. J. Heinovski, F. Klingler, F. Dressler, and C. Sommer, “Performance Comparison of IEEE 802.11p and ARIB STD-T109,” in *8th IEEE Vehicular Networking Conference (VNC 2016)*, Columbus, OH: IEEE, Dec. 2016, pp. 1–8.

12. M. Mutschlechner, F. Klingler, F. Erlacher, F. Hagenauer, M. Kiessling, and F. Dressler, “Reliable Communication using Erasure Codes for Monitoring Bats in the Wild,” in *33rd IEEE Conference on Computer Communications (INFOCOM 2014), Student Activities*, Toronto, Canada: IEEE, Apr. 2014, pp. 189–190.

13. F. Erlacher, F. Klingler, C. Sommer, and F. Dressler, “On the Impact of Street Width on 5.9 GHz Radio Signal Propagation in Vehicular Networks,” in *11th IEEE/IFIP Conference on Wireless On demand Network Systems and Services (WONS 2014)*, Obergurgl, Austria: IEEE, Apr. 2014, pp. 143–146.

14. F. Klingler, S. Tang, X. Liu, F. Dressler, C. Sommer, and J. Cao, “Faster Distributed Localization of Large Numbers of Nodes Using Clustering,” in *38th IEEE Conference on Local Computer Networks (LCN 2013)*, Sydney, Australia: IEEE, Oct. 2013, pp. 728–731.

15. M. Segata, B. Bloessl, S. Joerer, F. Erlacher, M. Mutschlechner, F. Klingler, C. Sommer, R. Lo Cigno, and F. Dressler, “Shadowing or Multi-Path Fading: Which Dominates in Inter-Vehicle Communication?” University of Innsbruck, Institute of Computer Science, Technical Report CCS-2013-03, Jun. 2013.

Chapter 2

Fundamentals and Related Work

BEFORE Vehicular Ad Hoc Networks (VANETs) became popular lots of research was done in the field of Mobile Ad Hoc Networks (MANETs), where vehicular communication could be described as a special case of a MANET [49]. However, due to the inherent mobility and high topology dynamics in vehicular networks, many proposed communication protocols are not suitable to fulfill all application requirements in VANETs. This way, vehicular communications evolved as an independent research domain, where the main goal is to provide efficient communication protocols which *(a)* do not overload the wireless channel, *(b)* take into consideration the – in some degree predictable – mobility of nodes in the network, to provide *(c)* protocols to transmit information from one node to another, or a group of other nodes with respect to high performance in terms of high reliability and low delay.

At the same time, development for a communication technology based on IEEE 802.11 WLAN started to support the high mobility of nodes present in a vehicular network scenario [50]. Work in this topic lead to the amendment IEEE 802.11p [51], which was later integrated into the IEEE 802.11 standard [22]. In particular, a dedicated frequency band specifically for applications in the Intelligent Transportation Systems (ITS) domain was reserved at 5.9 GHz.

Intuitively, to use wireless communication in vehicular scenarios, the first choice would be to use one of the already available standards IEEE 802.11a/b/g/n/ac, because they are already available, wide-spread in consumer and industrial hardware, and thus rather affordable. IEEE 802.11b/g/n operates in the 2.4 GHz band, and IEEE 802.11a/n/ac in the 5 GHz band, both being declared as Industrial, Scientific and Medical (ISM) bands. This regulations makes the frequencies being used by many different wireless devices, e.g., smart phones, private wireless networks. This therefore leads to quite crowded wireless channels on those bands. Moreover, protocol operation on ISM bands is usually not restricted to one particular protocol stack, thus multiple technologies, e.g., WLAN, Bluetooth, ZigBee, share a common frequency band, although they are not compatible with each other which makes coexistence a challenge [52].

A further disadvantage of IEEE 802.11a/b/g/n/ac is their limited usability in highly mobile networks: vehicles easily observe high relative velocities, and thus suffer from multipath and Doppler effects. Based on this high mobility, nodes forming a VANET are also loosely connected, thus wireless links can change very frequently.

IEEE 802.11a/b/g/n/ac usually support mutually exclusive ad hoc or infrastructure mode, where a dedicated station or Access Point (AP) periodically emits beacons to inform neighboring nodes about the presence of a particular network. Therefore to connect to such a network, a node needs to perform an association procedure requiring frame exchanges before data can be transceived, which leads to

non-negligible communication delays – too large for highly mobile networks like VANETs.

To address these issues, in Europe the Electronic Communications Committee (ECC), as well as in the U.S. the Federal Communications Commission (FCC) reserved several dedicated, and non-overlapping channels in the 5.9 GHz spectrum [23]. One particular channel was elected as Control Channel (CCH), the remaining ones are denoted as Service Channels (SCHs).

To cope with the high mobility of vehicles, the IEEE 802.11a Physical layer (PHY) was adopted to use only a channel bandwidth of 10 MHz instead of 20 MHz for commodity WLAN. Additionally, a new operation principle called "outside the context of a BSS" (OCB) was introduced, in which no association request/response is needed to exchange data between nodes.

In summary, the main changes between IEEE 802.11p and IEEE 802.11a are:

- **frequency band:** dedicated frequency band at 5.9 GHz instead of the 2.4 GHz or 5 GHz ISM bands.
- **bandwidth per channel:** 10 MHz in comparison to 20 MHz for IEEE 802.11a.
- **timing parameters:** doubled the Orthogonal Frequency Division Multiplex (OFDM) symbol durations to mitigate multipath effects.
- **gross data rate:** 3 Mbit/s to 27 Mbit/s in comparison to 6 Mbit/s to 54 Mbit/s for IEEE 802.11a.

These changes at the PHY layer allow more tolerance for a receiver, leading to a theoretical maximum communication range over several kilometers, depending on the transmit power and receiver characteristics. We will investigate these characteristics in the next section.

Very recently (in March 2018), a new study group *IEEE 802.11 Next Generation V2X* was formed, which focuses on new developments of vehicular communications to adopt recent technologies on the PHY and Media Access Control (MAC) layer (of e.g., IEEE 802.11n/ac) for new Vehicle-to-X (V2X) applications with the goal to keep backwards compatibility with IEEE 802.11p.

The remainder of this chapter is organized as follows: In Section 2.1 we analytically explore the limits of IEEE 802.11p based communication and the demand of vehicular networks, which will serve as a motivation to develop intelligent and resource-conserving networking protocols which are designed within this PhD thesis. In Section 2.2 we give an overview about current efforts on communication protocols in VANETs by focusing on beaconing protocols and network congestion algorithms specifically built for IEEE 802.11p. A fundamental basis of our developed network architecture are Bloom filters, for which we outline details in Section 2.3. Finally,

in Section 2.4 we present a short overview about the simulation framework we are using for developing and evaluating our proposed network protocols and algorithms. Related work which correlates to each specific topic presented in this PhD thesis is covered in its related chapter.

2.1 On the Capacity Bounds of IEEE 802.11p

In the following we analytically evaluate the capacity bounds of wireless communication based on IEEE 802.11p. The main idea is to derive the maximum theoretical possible communication distance, channel utilization and achievable goodput at application layer for a typical IEEE 802.11p compatible wireless card as it is used in many prototypes for vehicular communications [43], [53]. Finally, we put these results in the context of a realistic vehicular scenario, which serves as a motivation for creating resource efficient VANET protocols.

2.1.1 Communication Distance

First, we evaluate the maximum communication distance for wireless data transmissions according to IEEE 802.11p [22]. For simplicity we assume a freespace path loss model and a communication range only limited by the receiver sensitivity of the wireless card. Further, no antenna gains are assumed. In Figure 2.1 we show the received signal strength over distance when transmitting with a power of 33 dBm on 5.9 GHz as suggested by [23]. The received signal strength $P_r[dBm]$ is derived as

$$P_r[dBm] = P_t[dBm] - 10\log_{10}\left(16\pi^2\frac{d^\alpha}{\lambda^\alpha}\right) \tag{2.1}$$

where $P_t[dBm]$ is the transmit power, d is the distance, α is a path loss coefficient depending on the environment, and λ is the wavelength [54].

As an example, when transmitting with a data rate of 24 Mbit the signal can be detected up to a distance of 1 km for a sensitivity of −83 dBm. This sensitivity value depends on the used wireless chip; for our case we used a wireless card of type WLM200N5-23ESD[1]. We have to note that we do not consider any bit error probabilities here – thus communication distance in reality could be lower. However, we clearly see the distance at which communication partners are influenced for channel access. This is especially important when considering channel access according to CSMA/CA where a node possibly has to backoff because another node already is transmitting data on the channel.

[1] https://www.pcengines.ch/pdf/wlm200n5-23

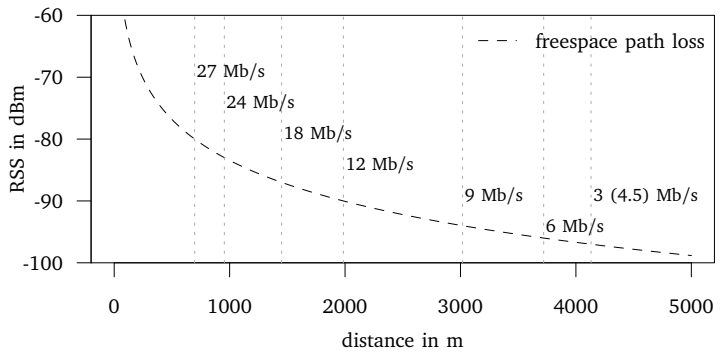


Figure 2.1 – Freespace path loss model with $\alpha = 2.2$ and a transmit power of 33 dBm on channel 180 (5.9 GHz); the vertical lines denote the receiver sensitivity and thus the maximum range at a given data rate for a WLM200N5-23ESD WLAN card outfitted with an AR9220 wireless chip.

2.1.2 Channel Utilization

Next, we consider how well channel access works for different packet sizes and data rates for communication. We assume here a fully saturated channel, thus a node continuously sending frames on the wireless channel. According to the IEEE 802.11 standard [22] the time for transmitting a broadcast frame including time for channel access is derived as

$$t_{\text{tx-mac}} = t_{\text{AIFS}} + \mathcal{U}(0, CW_{\text{min}}) \times t_{\text{SLOT}} + t_{\text{tx}} + t_{\text{SIFS}} \tag{2.2}$$

where $t_{\text{AIFS}} = t_{\text{SIFS}} + \texttt{AIFSN} \times t_{\text{SLOT}}$. The number of `AIFSN` defines the arbitration time for channel access and is set to 6 as defined in IEEE 802.11 [22]. The remaining parameters were configured to $CW_{\text{min}} = 15$, $t_{\text{SLOT}} = 13\,\mu\text{s}$, and $t_{\text{SIFS}} = 32\,\mu\text{s}$.

Moreover, the time to actually transmit the frame is derived as

$$t_{\text{tx}} = T_{\text{preamble}} + T_{\text{signal}} + T_{\text{sym}} \times \left\lceil \frac{16 + l + 6}{N_{\text{DBPS}}} \right\rceil \tag{2.3}$$

where $T_{\text{preamble}} = 32\,\mu\text{s}$, $T_{\text{signal}} = 8\,\mu\text{s}$, $T_{\text{sym}} = 8\,\mu\text{s}$, l is the number of bits including all lower layer headers, and N_{DBPS} defines the number of bits transmitted by each symbol. For example configuring a data rate of 54 Mb/s on a 20 MHz wide channel corresponds to a data rate of 27 Mb/s on a 10 MHz wide channel and leads to $N_{\text{DBPS}} = 216$ bit.

The channel utilization, i.e., the fraction of time the channel is busy is derived as

$$f_{\text{busy}} = \frac{t_{\text{tx}}}{t_{\text{tx}} + t_{\text{AIFS}} + \mathcal{U}(0, CW_{\text{min}}) \times t_{\text{SLOT}} + t_{\text{SIFS}}}. \tag{2.4}$$

In Figure 2.2 we show the channel utilization for different data rates with increasing packet sizes (not including lower layer headers). For each frame a header length of 8 B (LLC) and 28 B (WLAN) is added. For a data rate of 12 Mb/s and a packet size of $l_{\text{payload}} = 1000\,\text{B}$, the channel utilization is about 80 %. We have to note that no interference and collisions are taken into account in this example.

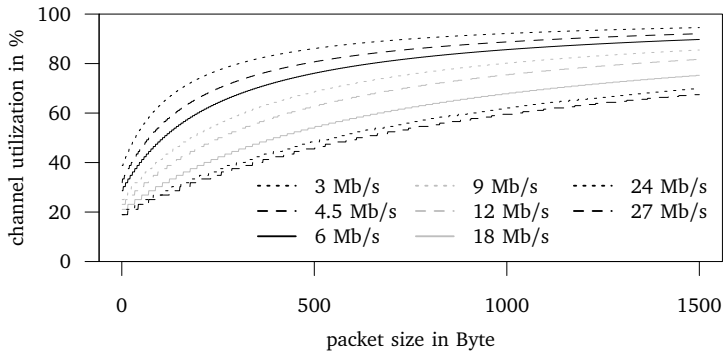


Figure 2.2 – Expected channel utilization for different data rates and packet sizes. With higher data rates the time spent transmitting the packet on the channel gets lower, thus the fraction to the time spent for channel access increases.

2.1.3 Goodput

The goodput for communication, i.e., the achievable throughput at application layer is derived as

$$g = \frac{1\,\text{s}}{t_{\text{tx-mac}}} \times l_{\text{payload}}. \tag{2.5}$$

We show the results for different sizes of l_{payload} in Figure 2.3.

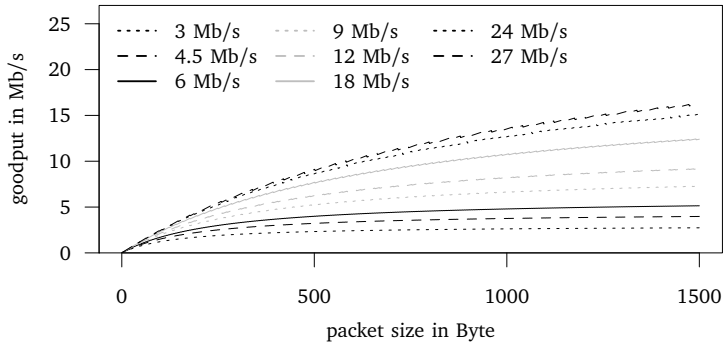


Figure 2.3 – Expected goodput at application layer for different data rates and packet sizes. With larger packet sizes the impact of channel access and time spent for transmitting protocol headers decreases, thus the achievable goodput increases.

We clearly see that the achievable goodput at application layer is much lower than the configured data rate on the wireless link. This happens due to two main reasons: (*a*) time lost for channel access, in particular during the arbitration time t_{AIFS} and channel contention $\mathcal{U}(0, CW_{\text{min}}) \times t_{\text{SLOT}}$, where no data transmission occurs, and (*b*) additional headers for lower layers, in particular 8 B for LLC, 28 B for WLAN, and $T_{\text{preamble}} + T_{\text{signal}}$ for the PHY layer.

2.1.4 Vehicular Scenario

In this section we evaluate the required data rate for a realistic vehicular scenario based on a simple example application – vehicle platooning. As a scenario we consider a typical highway with $l_{\text{lanes}} = 6$ lanes (3 lanes per direction), and no obstacles, thus Line of Sight (LOS) for radio communication. To model multiple densities of vehicles on the highway we vary the inter-vehicle gap on each lane, thus the space needed for each vehicle including the safety distance to the front vehicle. We choose this distance $d_{\text{length+gap}}$ between 4 m and 100 m representing a fully congested scenario and a very low density scenario, respectively. Further we assume a communication distance $d_{\text{comm-dist}}$ of 1000 m, that means that within that distance at most one transmission at the same time that will not cause collisions (or lost frames) can take place. We also assume a deployment fraction of $f_{\text{deployment}} = 0.5$, meaning that only half of the vehicles in the scenario are equipped with radio communication technology to participate in the wireless network. For data transmission we configure a message rate of $r_{\text{msg}} = 10$ packets per second for each vehicle. This configuration has been found beneficial for many applications, e.g., vehicle platooning [55], [56]. We then vary the size for each packet l_{payload} between 100 B and 1500 B.

We calculate the number of communicating cars per kilometer in the scenario as

$$n_{\text{cars}} = \frac{1000\,\text{m}}{d_{\text{length+gap}}} \times l_{\text{lanes}} \times f_{\text{deployment}}. \tag{2.6}$$

Next, we derive the number of transmissions per second within the communication distance for all vehicles as

$$r_{\text{total-packets}} = \frac{n_{\text{cars}}}{1000\,\text{m}} \times d_{\text{comm-dist}} \times r_{\text{msg}}. \tag{2.7}$$

Finally, the overall data rate observed within the communication distance is

$$g_{\text{necessary}} = r_{\text{total-packets}} \times l_{\text{payload}}. \tag{2.8}$$

In Figure 2.4 we plot the necessary goodput for different vehicle densities and packet sizes. Please note the scaling for goodput on the y-axis. In comparison to the achievable data rate in Figure 2.3 we see that wireless communication has difficulties

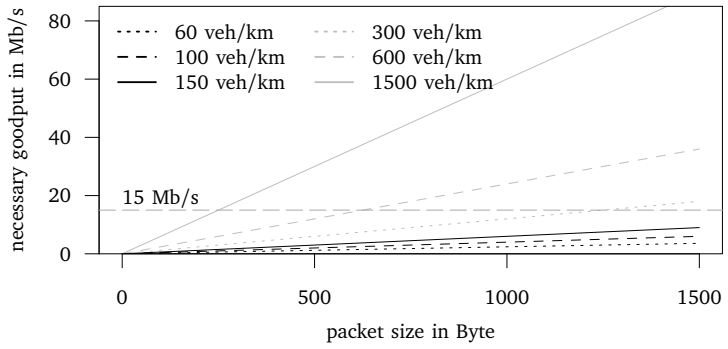


Figure 2.4 – Expected necessary goodput within 1000 m communication range on a 6 lane (3 per direction) highway for a given vehicles density (vehicles per kilometer on all lanes) and packet size; 50 % of the nodes are using wireless communication and sending 10 packets per second.

to fulfill high datarate demands in dense network scenarios. When all vehicles in the scenario communicate the data rate will be doubled.

2.1.5 Lessons Learned

Radio Communication according to IEEE 802.11p [22] is suitable to fulfill communication among a distance of more than 1 km range. However, this advantage at the same time becomes a disadvantage, since all nodes within interference distance of a sender are affected by its data transmission. We define interference distance here as the distance for which transmissions negatively affect ongoing reception of different frames on other nodes. Consequently, high bandwidth demanding applications, e.g., video streaming, may suffer from the limited bandwidth of maximum 27 Mbit/s and can cause severe congestion on the wireless channel. These findings will serve as a fundamental motivation to create resource aware and bandwidth preserving communication protocols for vehicular communications to avoid overloading the wireless channel and to support multiple different applications to access the wireless channel at the same time.

2.2 VANET Communication Protocols

In the following section we give an overview about popular communication protocols in the context of VANETs and summarize protocol stacks and algorithms which are later used in the chapters of this thesis. More detailed insights to related work and fundamentals are outlined in the corresponding chapters.

In general, protocol stacks for vehicular communications often focus on a single application domain, e.g., *cooperative awareness*, but offer the capability to make

the underlying broadcast protocol available to other applications [57]. A popular use case for this is *beaconing*, in which Cooperative Awareness Messages (CAMs) standardized within ETSI EN 302 637-2 [58] in Europe, or Basic Safety Messages (BSMs) standardized in SAE J2735 [59] within the U.S., are broadcast at fixed intervals usually ranging between 0.1 s . . . 1 s [58], [60]. This scheme is used to inform neighboring nodes about a vehicle's current status, e.g., position, speed, acceleration, heading. Although there are specific differences among these two message types, e.g., information about a hard-brake event is announced within a BSM but not within a CAM [61], in Europe there is an additional event-based message type defined called Decentralized Environmental Notification Message (DENM) specified as ETSI EN 302 637-3 [62]. Since all these schemes often rely on broadcast, their communication is not sufficiently reliable, thus approaches which keep the number of retransmission at a low value have been investigated [63]. The most important challenge for all broadcast based protocols is that the frequency of possible transmissions strongly depends upon the available capacity of the wireless channel, which is shared among all participating vehicles. When using simple retransmissions of packets, the underlying broadcast protocol needs to prevent broadcast storms [64]. To avoid channel congestion, adaptive beaconing solutions which focus on congestion control of the wireless channel have been developed [65], [66]. As a next step, fairness was added as a prime objective to these adaptive beaconing solutions [67].

One of the first approaches focusing on the problem of channel congestion is Adaptive Traffic Beacon (ATB) [66], which focuses on two main research questions: How often can the protocol send beacons; and how often should the protocol send beacons? To accomplish this, two different metrics have been developed: *channel quality* C and the *message utility* P, which together allow the calculation of the beacon interval I at which beacons are broadcast. ATB adjusts I such that it becomes minimal only for the highest message utility and the best channel quality, with the aim to send as often as possible, but avoid frame collisions at any time. However, in all other cases, the beacon frequency is reduced drastically, allowing channel usage by other applications.

Another possibility to avoid channel congestion is to adapt the transmit power to allow spatial reuse of the spectrum [68]. However, research investigations showed that transmit power control could be counter-productive when using safety-critical applications [69].

Many of these ideas have been picked up by standardization bodies: ETSI ITS-G5 incorporates a standardized beaconing protocol, which provides with Decentralized Congestion Control (DCC) Transmit Rate Control (TRC) [13] the possibility to adapt the inter-beacon interval according to the measured channel load. This is accomplished by using a state machine which refers to different beaconing intervals.

The wireless channel utilization is periodically sampled and TRC performs the necessary state transitions whenever the channel busy ratio is above or below a defined threshold. This allows the beaconing protocol to react to conditions of an overloaded wireless channel, while providing the lowest beaconing interval when channel utilization is low.

Pulsar [70] takes advantage of piggybacking 2-hop information for congestion control in order to maximize the beacon rate among all nodes. This is especially useful when using safety applications.

Moreover, LIMERIC [71] provides a novel adaptive congestion control algorithm which adapts the beacon rate with the goal to provide fairness among all nodes. Whenever abrupt topology changes take effect, [72] shows that a more aggressive approach for beaconing would be beneficial – which consequently leads to an faster update rate, and thus lower beaconing interval. However this comes at the cost of an increased channel load.

To keep the required data rate (and thus channel load) low, probabilistic data structures like *Bloom filters* [73] have been investigated and found beneficial for network applications [74]–[77]. These approaches help to further address congestion on the wireless channel besides the already mentioned adaptive beaconing solutions. However, these approaches only focus on very specific application scenarios and do not investigate generic neighbor management, which is an important building block for VANETs.

A first step towards neighbor information dissemination among 2-hops using Bloom filters has been presented very recently [78]. However, using the intersection of several Bloom filters as proposed by their approach increases the false positive error rate, which further causes loss of information. Moreover, in comparison to approaches not using Bloom filters, the presented work only allows the reduction of beacon overhead without substantial performance gain on application layer.

Another direction for communication is *Geocasting* which goes a step further than simple 1-hop or 2-hop broadcasting [79]. Here broadcasting is combined with geographical knowledge to fulfill many additional requirements of the application [80], [81].

Even more recent approaches combine Geocasting with capabilities of Delay Tolerant Networking (DTN) [82] where protocols exploit vehicle trajectories for their operation [83].

2.2.1 The ATB Protocol

An important question for beaconing protocols in vehicular networks is *(a) how often* information should be exchanged among nodes to not overload the channel, but still provide high information dissemination speeds, and *(b) which* information

should be selected for an opportunity to broadcast a beacon to all neighbors within the communication distance.

Sommer, Tonguz, and Dressler [66], [84] present the single-channel ATB protocol, which addresses these questions and serves as a basis for our work presented in Chapter 6. What follows now in this section is a summary of this ATB protocol [66], [84]. First we start with a description how information is stored on vehicles and how the beacon interval is calculated. Subsequently, we outline the algorithms for determining the channel conditions and message priorities.

2.2.1.1 Knowledge Base Management

Similar to other approaches [85], ATB stores received traffic information within an individually maintained knowledge base per vehicle. This knowledge base is represented as an ordered list of entries containing traffic information items sorted according to an individually calculated priority. Every change in this knowledge base (e.g., received beacons causing an update of information or new traffic events) leads to a recalculation of the message utility representing the priority of each entry.

The fundamental idea of ATB is to only store information relevant for the vehicle (e.g., the most recent information about a traffic jam), and suppress sending irrelevant information. To achieve this goal, each event (either locally perceived from sensors, or received from neighboring nodes via beacons) either updates existing entries of the knowledge base or is appended to it. Moreover, a garbage collector removes outdated and nonrelevant entries to limit the size of the knowledge base.

For each opportunity to transmit a beacon, ATB takes as many entries from the top of the list (i.e., the most important ones) as there is room for in a single frame. This frame is then transmitted as a broadcast on the CCH to other vehicles in the vicinity. Sending only a single frame has the advantage that there is no need for managing fragmentation of messages, and thus avoids possible retransmissions of lost chunks. A positive side effect of this is that the channel capacity is used in an efficient way, since overhead is minimized: every frame consists of as many knowledge base entries as there is room for.

There exist a variety of approaches in the literature to calculate the utility of knowledge base entries (and, as a consequence, the target priority), one example being FairAD [86]. Yet, for the sake of simplicity, ATB chooses a sum considering the age of an entry and the proximity to the event origin, as presented in detail in the original publication [66]. To inform neighbor nodes about updated traffic conditions, each knowledge base entry contains the type of event (e.g., accident, closed road), time stamp, location, priority, and an identifier of the affected road.

2.2.1.2 Beacon Interval Calculation

The frequency at which knowledge base entries are broadcasted is in part derived from the *message utility* of the highest priority entry in the knowledge base. Similar to the message utility calculation, possible approaches range from calculations considering the age of an entry and the current proximity to the event origin as presented in [66] up to very recent schemes that are also able to capture metrics of fairness [67], [86].

Besides this, the calculation of the beacon interval also considers a second class of metrics: *channel quality*. The core principle of ATB is to transmit beacons as often as possible, but avoid overloading the wireless channel at any time to prevent any possible wireless collisions.

Based on the message utility P, the channel quality C, and a relative weighting w_I, the desired beacon interval ΔI bounded by $[I_{\min}, I_{\max}]$ is derived according to [66] as follows.

$$I = (1 - w_I) \times P^2 + w_I \times C^2 \tag{2.9}$$

$$\Delta I = I_{\min} + (I_{\max} - I_{\min}) \times I \tag{2.10}$$

The interval weighting factor w_I gives the metric of channel quality a higher impact to decide a reasonable beaconing interval.

In summary, the calculation of the channel quality bases its decision on three metrics that correspond to the channel capacity in the past, the present, and the future:

Past: To get an indication about the channel load in the past, we observe packet collisions and count them. This is straightforward to do in simulations, however in practice that is complicated or even impossible to achieve. However, on receiver side, the number of packet collisions can be estimated: For a received signal which is strong enough to be decoded, but contains bit errors, the possibility is very high that an interferer caused a packet collision.

Present: To capture the current channel conditions, the signal quality for the last reception is taken as a metric, which gives a rough indication about the current channel quality.

Future: To predict future channel quality, possible communication of other vehicles in the future is estimated by taking into account the number of neighbors. Under the assumption that every vehicle uses a unique identifier which is included in each transmitted beacon, the number of individual vehicles contending for wireless channel access can be determined for a predefined period of time.

2.2.2 ETSI ITS-G5

As already explained earlier, ETSI ITS-G5 aims to provide a complete networking stack for Inter-Vehicle Communication (IVC) [23]. At its core is the DCC module [13], which adapts various transmission parameters (e.g., message rate, transmit power) to keep channel utilization at reasonable values.

In particular, the ETSI ITS-G5 protocol stack generates messages both, periodically for cooperative awareness by transmitting information as a CAM [87], and event based by transmitting information within a DENM [88]. These messages are controlled by DCC [13] which acts as a rate-limiter by using discrete states in a state machine. It defines the rate of transmitted messages per second (TRC), the data rate of transmitted messages (Transmit Datarate Control (TDC)), the transmit power of frames (Transmit Power Control (TPC)) as well as the sensitivity of Clear Channel Assessment (CCA) (DCC Sensitivity Control (DSC)). The decision when to switch states in the DCC state machine is based on periodically measured channel load.

The main state machine is outlined in Figure 2.5, which consists of three independent states when operated on the CCH. It is possible to extend this state machine with additional states (or even a separate state machine) within the Active-State as it is done when DCC operates on SCHs.

We now focus on the ETSI ITS-G5 TRC functionality of the DCC: We use the ETSI ITS-G5 TRC mechanism in our simulation studies, which adapts the frequency of beacon transmission and represents a standardized transmit rate adaption algorithm. The beacon interval can take one of three independent values as depicted in

Figure 2.5 – Simplified state machine of ETSI ITS-G5 DCC [13]. The active state can represent multiple different states according to the selected channel.

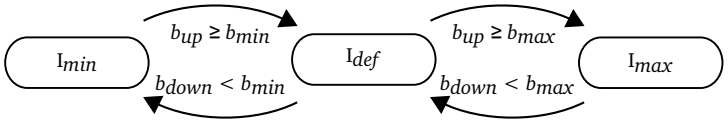


Figure 2.6 – Simplified state diagram of a transmit rate control algorithm. The beacon interval (defined in the three states) changes according to the channel busy ratio. The transition is performed if the busy ratio is above or below a threshold for a given time interval; based on [35] © 2016 Elsevier B.V.

Figure 2.6, and state transitions are performed according to the observed channel busy ratio. The channel busy ratio b_t is measured by the DCC in a periodic fashion according to a straightforward sampling process: It is defined as a fraction of samples within a defined time window of approximately 1 s, for which the channel was measured by the CCA process as busy. Based on values of b_t, state transitions are performed. We use a version of three independent states representing three different beaconing intervals as shown in Figure 2.6. The parameters b_{up} and b_{down} represent filtered channel busy ratios in time windows of 1 s and 5 s, respectively. When the measured channel load is above or below the thresholds b_{min} and b_{max}, the beacon interval is increased or decreased, respectively. This procedure allows the protocol to react to overloaded channels, however with some delay for performing the state transitions.

2.2.3 IEEE 1609 WAVE

For IVC services in most countries, more than a single channel is available for communication between participating vehicles in the 5.9 GHz band. In the U.S., as well as in Europe, several non-overlapping channels are available, depending on the offered service [89] as illustrated in Table 2.1.

When only a single radio is installed per vehicle, it is an open question how to best take advantage of multiple channels and to decide when to switch between channels. The IEEE 1609 WAVE series of standards [90] building upon IEEE 802.11p aim to provide a comprehensive communication stack for IVC. As already outlined previously, IEEE 802.11p is an optimized variant of IEEE 802.11a operating at a dedicated frequency band at 5.9 GHz. The timing values of MAC and PHY were modified to meet the requirements of IVC.

In order to operate on multiple channels, the IEEE 1609 WAVE protocol stack adopts a split phase channel switching approach. All nodes periodically switch their radio to a well-known CCH at specific points in time to exchange information about available services and their respective SCHs, as illustrated in Figure 2.7. These individual services are then provided on the announced SCH.

An example of this IEEE 1609.4 multi-channel system is illustrated in Figure 2.7: Each second, the nodes are synchronized by means of GPS. The radios then continu-

	SCH	SCH	SCH	***CCH***	SCH	SCH	SCH
channel	172	174	176	178	180	182	184
GHz	5.860	5.870	5.880	5.890	5.900	5.910	5.920
dBm	33	33	33	44.8	23	23	40

Table 2.1 – WAVE use cases and parameters adopted from [35], [89]; based on [35] © 2016 Elsevier B.V.

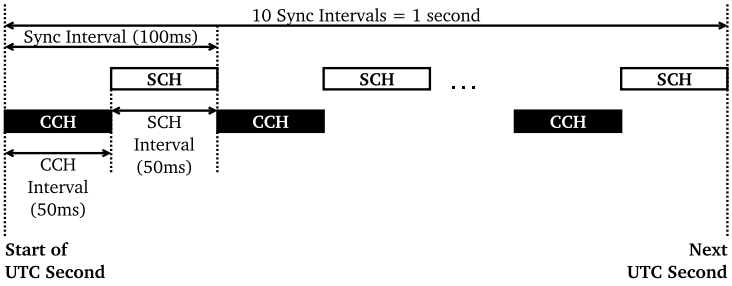


Figure 2.7 – The principle of the WAVE Split Phase Protocol. A node periodically switches between the CCH and one of the available SCHs. Each node synchronizes its Sync-Intervals by GPS. For simplicity we do not show the 4 ms guard intervals; based on [35] © 2016 Elsevier B.V.

ously switch between the CCH and one of the available SCHs, each having an interval of 50 ms. Further, to cope with synchronization inaccuracies, a guard interval of up to 4 ms is included at the beginning of each CCH and SCH interval. We discuss the impact of this switching in more detail in Chapter 6 where we build our work upon this channel switching scheme.

The main research question regarding this multi-channel operation is to select the time *when* to use *which* SCH to allow robust and reliable communication among vehicles with low interference. Moreover, the aim of such a scheduling algorithm is to provide an even utilization among all SCHs in order to not overload a specific channel. This will help avoiding issues introduced by multi-channel operation, like the deafness, and the multi-channel hidden terminal problems [91].

2.3 Bloom Filter

In 1970, Burton Howard Bloom introduced a space-efficient and probabilistic data structure, called a Bloom filter [73]. With this data structure, it is possible to (probabilistically) check whether an element is member of a set with the possibility of false positives. In particular, a Bloom filter $\mathcal{B}$ consists of a bit array of length m representing a hash, where all elements are initialized to 0.

Mathematically that is,

$$\mathcal{B} = \{\mathcal{B}_0, \ldots, \mathcal{B}_{m-1}\}.$$

Further, we need k independent and different hash functions $\{h_0(\cdot), \ldots, h_{k-1}(\cdot)\}$, for which each of them map an element b to one of the m array positions $[0, m-1]$ in the Bloom filter $\mathcal{B}$ by taking advantage of a uniform distribution. When an identifier b of a node is inserted into the Bloom filter $\mathcal{B}$ (written as $\mathcal{B} \leftarrow b$), all bits indicated by any of the k hash functions are set to 1 as shown in Figure 2.8. More formally,

this is expressed as

$$\mathcal{B} \leftarrow y \quad \Rightarrow \quad \forall i \in [0 \ldots k-1] : \mathcal{B}_{h_i(b)} \leftarrow 1.$$

For simplicity, we denote the insertion of all elements b in a set $\mathbb{B}$ as $\mathcal{B} \leftarrow \mathbb{B}$.

Initially, we assume that all bits in $\mathcal{B}$ are set to 0, representing an empty Bloom filter. For each inserted element in the Bloom filter the number of 1 bits intuitively increases. When some elements are present in the Bloom filter, some bits for a new element b' may have been set to 1 already, others need to be explicitly set to 1.

One of the most important operations provided by Bloom filters is to test whether an element b' is part of this particular Bloom filter. In order to do so, the same k hash functions are used to see which bits need to be set in order to conclude (with a chance of a false positive answer) if an element is part of the Bloom filter. This can be expressed more formally as

$$b' \in \mathcal{B} \quad \Leftrightarrow \quad \forall i \in [0 \ldots k-1] : \mathcal{B}_{h_i(b')} = 1.$$

In other words, we feed the element to all k hash functions to derive the corresponding k positions in the bit array $\mathcal{B}$. Whenever one of the bits at these positions is set to 0, the element is definitely not contained in the set. In the other case, if all bits on these k positions are set to 1, either the element is contained in the set, or, the bits were set when adding other elements to the Bloom filter, which thus results in a false positive answer. For standard Bloom filters, false negatives are not possible.

The check for the presence of an element is a probabilistic test which depends on the used hash functions: the only question which can be answered correctly is whether a node was *not* part of the Bloom filter. In other words, false positives are possible for the decision whether an element is part of the Bloom filter. Consequently, the expected fraction of errors gets smaller for larger m, and increases for larger n. This way, a value for m can be chosen to keep the number of false positives small enough for the envisioned applications.

A standard Bloom filter does not allow the deletion of elements, since a bit within the bit array could have been set to 1 by an insertion operation of an element

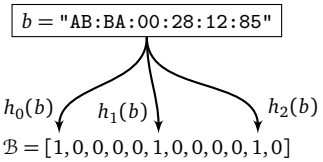


Figure 2.8 – Adding an element b (here, a MAC address) to an initially empty Bloom filter $\mathcal{B}$ (of size $m = 12$ bit and using $k = 3$ hash functions); based on [32] © 2018 IEEE.

which took place after the element which is to be deleted was added to the Bloom filter. One approach to tackle this problem is to instantiate a second Bloom filter, which contains deleted elements. In the case when an element is part of the original Bloom filter, and at the same time part of the one with deleted elements, it can be assumed that the element was deleted – which again is prone to false positives for this particular decision.

By design this overall procedure may cause the problem of false negatives: An element which is falsely queried in the Bloom filter for deleted elements causes a wrong decision whether an element is contained in the original Bloom filter.

Counting Bloom filters [92] represent an alternative approach, which, however, can also suffer from false negatives [93]. Here, instead of a single bit for each position in the vector, a bit array is used which holds the number of how often that particular position was set to 1. Whenever an element needs to be deleted from the Counting Bloom filter, it checks whether the element is part of the Bloom filter, and on success decrements the counters of each position. When the check of presence of an element produces a false positive, the values get decremented although they should not. The consequence for this are false negatives – the wrong answer that an element is definitely not part of the Counting Bloom filter.

In order to construct a standard Bloom filter, following parameters need to be known: the Bloom filter size m and the number of hash functions k to use. A suitable size m (in bits) [75] can be derived by taking into account the expected number of insertions n and acceptable false positive error rate p_{fp} which leads to

$$m = -\frac{n \ln p_{\text{fp}}}{(\ln 2)^2} \,. \tag{2.11}$$

When the expected insertion count is unknown or dynamic (like in vehicular scenarios for the number of neighbors), this number n can be approximated by using traffic density estimation protocols [94], [95]. Moreover, probabilistic data aggregation protocols using Flajolet-Martin sketches [96] help to determine a fitting Bloom filter size. Darwish and Bakar [94] give an overview of traffic density estimation approaches by focusing on infrastructure-free vehicular networks. In this context, when penetration rate is rather low (30 %), [95] was identified as a promising approach, which allows estimating the traffic density in a fully distributed manner.

To continue with the calculation of the optimal number of hash functions k [97] to minimize the false positive error rate for a given Bloom filter size and inserted element count we use

$$k \approx \frac{m}{n} \ln 2 \,. \tag{2.12}$$

For a known Bloom filter size m, number of inserted elements n as well as a known number of hash functions k, the false positive error probability p_{fp} for membership tests in a Bloom filter can be derived [74] as

$$p_{\text{fp}} = \left[1 - \left(1 - \frac{1}{m}\right)^{kn}\right]^k . \tag{2.13}$$

To see the performance of Bloom filters, we show in Figure 2.9 the false positive rate as a function of the Bloom filter size and the number of inserted elements. We calculated the optimal value of k for each Bloom filter size according to Equation (2.12). For small Bloom filter sizes we can clearly see the steep increase of the false positive rate.

Beside simple membership checks in Bloom filters, another operation could be beneficial for Bloom filters: to estimate the number of inserted elements (cardinality), and to perform set-operations on Bloom filters (intersection, union). In order to do so, we introduce common operations performed on Bloom filters, which we use in the remaining part of this thesis. As an example, we suppose to have two Bloom filters $\mathcal{A}$ and $\mathcal{B}$ which use the same number of bits m and the same hash functions. A Bloom filter that represents $\mathcal{A} \cup \mathcal{B}$, can be generated by simply performing a $\vee$ (bitwise OR) operation on $\mathcal{A}$ and $\mathcal{B}$ in order to derive the union of those two Bloom filters.

Similarly, a Bloom filter representing the intersection of two Bloom filters $\mathcal{A} \cap \mathcal{B}$ is generated by performing a $\wedge$ (bitwise AND) operation among $\mathcal{A}$ and $\mathcal{B}$. Here we have to note that this is only an approximation of the intersecting Bloom filter [97], however that can be seen as sufficient for large filters [98].

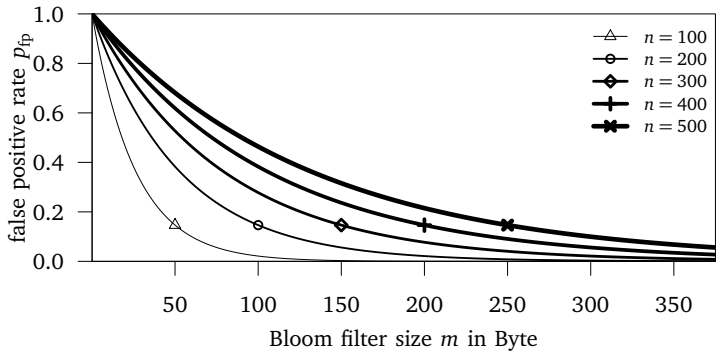


Figure 2.9 – False positive rate p_{fp} vs. Bloom filter size m for increasing numbers of inserted elements n; based on [32] © 2018 IEEE.

Finally, to estimate the number of inserted elements $|\mathbb{C}|$ of a Bloom filter $\mathcal{C}$, better known as cardinality of a Bloom filter, we perform

$$|\mathbb{C}| \approx |\mathcal{C}| = -\frac{m \ln(1 - \frac{c(\mathcal{C})}{m})}{k} , \tag{2.14}$$

where the function $c(\cdot)$ denotes the number of 1 bits within the Bloom filter [99]. For a Bloom filter where all bits are set to 1, the cardinality is not defined. For illustration purposes, we plot this as infinity in the remaining part of the thesis.

2.4 Realistic Network and Road Traffic Simulation

In the following chapters, beside analytical evaluations and measurements on hardware prototypes, we also use realistic network- and road-traffic simulations on which we focus in this section.

The Open Source software framework Veins [2], which we use for our simulation studies, is well established in the vehicular networking community and provides an extensive suite of models for vehicular communications. These models are closely validated against measurements of Field Operational Tests (FOTs) and capture obstacle shadowing [100], two ray fading [54], [101], antenna radiation characteristics [102], and adjacent channel interference [103] as well as Physical layer and MAC layer models of ARIB T109 [44]. Moreover it relies on fully-detailed models of IEEE 802.11p and IEEE 1609.4 WAVE multi-channel operation.

Veins itself evolved from the MiXiM framework [104], [105] which represents a mathematical toolkit for the OMNeT++ discrete event simulator to model continuous signals (e.g., wireless communication) within discrete events. These signals are modeled as two-dimensional (time and frequency) functions of signal power that are influenced by path loss and fading effects (both stochastic and deterministic, e.g., due to buildings). Frame reception is then performed based on division of these functions (for signal, interference, and noise) to derive the Signal to Noise plus Interference Ratio (SNIR) and Signal to Noise Ratio (SNR) to compute from that the bit error rates and evaluate, whether a frame can be received successfully or needs to be dropped due to interference (packet collision). Since Veins 4 the NIST error rate model of ns-3 [106], [107] is used for packet error rate calculations.

Besides realistic simulation of network traffic, Veins provides realistic node mobility by connecting to the SUMO[3] road traffic simulator [108]. The connection of these two simulation frameworks (Veins, SUMO) is bidirectionally coupled, that means not only the road traffic simulator influences network communication, but models in the network simulator can also adapt the mobility of vehicles. One example

[2] `http://veins.car2x.org/`
[3] `http://sumo.dlr.de/`

for this could be the simulation of a Traffic Information System (TIS) disseminating information about road traffic congestions, and thus allowing vehicles to take a detour.

To conclude this chapter we learned important vehicular networking protocols and standards, and the theoretical communication limits for IEEE 802.11p based communication upon which these standards build on. Further we outlined necessary fundamentals on which the remaining part of this PhD thesis is based on.

Chapter 3

The Impact of Head of Line Blocking in Highly Dynamic WLANs

In the previous chapter we learned fundamentals about current communication protocols in the Vehicular Ad Hoc Network (VANET) domain, and how protocol operation is performed in various standardized protocol stacks, e.g., ETSI ITS-G5 outlined in Section 2.2.2. Within this protocol stack, information can be transmitted towards a specific geographic region by means of Geocasting. One algorithm supporting this type of information dissemination takes advantage of IEEE 802.11 unicast communication [27]. For the IEEE 802.11 WLAN standard, unicast communication between a sender and a transmitter is provided by using retransmission of frames in case the reception failed. Here, a receiver which successfully decoded an individually addressed frame – also called unicast frame – acknowledges the reception by transmitting a small frame (acknowledgment) shortly upon reception of the original frame. This way, the sender knows if the transmission was successful, or not, and thus can schedule a retransmission. In the current version of IEEE 802.11, and its amendment IEEE 802.11p for vehicular communications, the time between retransmissions of a single frame increases exponentially. This exponential backoff procedure was selected to allow a relaxation of the wireless channel, as lost acknowledgments were treated as implication of an overloaded wireless channel. However, we believe that this statement does not hold in highly dynamic networks, where nodes can move outside the communication range in the time span between the update of neighbor information and selection of a particular forwarding node to whom unicast frames are transmitted.

In this chapter we study the impact of retransmissions and exponential backoff behavior in vehicular communications, where the remaining chapter is based on the following peer-reviewed publications:

- F. Klingler, F. Dressler, and C. Sommer, "The Impact of Head of Line Blocking in Highly Dynamic WLANs," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 8, pp. 7664–7676, Aug. 2018, © 2018 IEEE.
  My contribution in this journal article was the design, implementation and evaluation of test cases to trigger lost acknowledgments in static scenarios for both, simulation and real world experiments. Further I designed and conducted experiments to actively jam (interfere) acknowledgments in order to study the behavior of commodity WLAN hardware in situations when acknowledgments were not received. For this I had to extend an existing system for generating interfering frames in a way to cope with the short timings necessary to interfere acknowledgment frames. Moreover I performed analytical evaluations and numerical simulations to validate my experiments. Finally I studied the impact of unicast communication on realistic scenarios in the VANET context in terms of simulation studies.

- F. Klingler, F. Dressler, and C. Sommer, "IEEE 802.11p Unicast Considered Harmful," in *7th IEEE Vehicular Networking Conference (VNC 2015)*, Kyoto, Japan: IEEE, Dec. 2015, pp. 76–83, © 2015 IEEE.
  My contribution in this conference publication was the extension of the network simulator to allow unicast communication and evaluation of lost acknowledgments in static and highly dynamic scenarios. Finally I evaluated the performance of IEEE 802.11 unicast communication in terms of a larger simulation study investigating the impact on a Geocasting protocol.

3.1 Motivation

In the following chapter we investigate the problem of head of line blocking when using unicast transmissions in highly dynamic Wireless LAN (WLAN) environments. We focus on VANETs, even though all our investigations also apply to other application domains when high mobility dynamics need to be taken into account.

As we already outlined in the previous two chapters, wireless communication based on IEEE 802.11 WLAN has become the standard for establishing vehicular networks [9]. As explained in more detail in Section 2.2, many protocol suites are building upon IEEE 802.11, like the U.S. IEEE Wireless Access in the Vehicular Environment (WAVE) protocol stack [17] and the European ETSI ITS-G5 protocol suite [23]. Both inheriting the physical and the MAC layer of IEEE 802.11.

Usually, operations of IEEE 802.11 WLAN MAC layer are tied to be used in the context of a Basic Service Set (BSS), in which a set of mobile nodes have synchronized among each other to use a common set of parameters [22]. However, the procedure of joining such a network involves message exchange that has been found to be too time consuming and, thus, infeasible for vehicular networks. This way, the WLAN standard has been amended with IEEE 802.11p which allows to be operated in "outside the context of a BSS" (OCB) mode [51]. This mode of IEEE 802.11p does not need procedures for authentication to other nodes and to scan for, join, or associate to a BSS. With this modifications, IEEE 802.11 becomes a reliable basis for vehicular communication: Commodity WLAN network interface cards [43] can be used with little modifications to the driver. Consequently, communication using OCB also supports low latency data transmission, which is crucial for most types of applications in vehicular networks. Therefore, it has evolved to be the standard extension underlying both the aforementioned U.S. and European VANET protocol stacks.

A fundamental part of the MAC is error control to ensure reliable frame transmission by using retry mechanisms – this is in particular relevant for unicast transmissions. To achieve this, IEEE 802.11 (and, by extension, IEEE WAVE and ETSI ITS-G5)

rely on a simple Automatic Repeat Request (ARQ) error control scheme in which by default any individually addressed (unicast) frame will be kept at the head of the transmit queue until an Acknowledgment (ACK) frame is successfully received. If no ACK frame is received within a pre-defined time window, the frame is automatically retransmitted. This process continues until an ACK frame is received, or until a retransmission limit is reached. Between each retransmission, the time increases exponentially to allow a relaxation of the channel, as there was the fundamental assumption that not received acknowledgments are caused due to an overloaded channel. This retransmission behavior leads to the so-called head of line (HOL) blocking problem in which one frame delays the transmission of other frames. Consequently, any transmit queue that is waiting for an ACK frame destined for a unicast frame is stalled. That means, this queue will neither transmit broadcast frames, nor any other individually addressed frame to another node until an ACK frame is received, or the retransmission limit is exceeded.

In the early days of WLANs [109], the head of line blocking problem has been identified and approaches have been developed to create an alternative MAC layer which monitors the individual wireless stations of a BSS and maintains separate transmit queues for each station. When employing such a MAC layer in a network, it can defer re-transmissions to *bad* stations until the estimated end of a (presumed) burst error. However, as already outlined before, the key assumption of such proposals has always been that lost frames are due to collisions or burst errors in the channel. Meaning, an overloaded channel leads to lost ACK frames, or lost or erroneous data frames in which the receiver does not reply with an ACK frame. For the envisioned target scenario at that time – networks having a static topology – this was a very reasonable assumption. However, even in static networks the effect is rather easy to trigger and can lead to denial of service attacks in which – intentionally – ACK frames of a specific station get jammed (see Section 3.4). Or even easier, to simply provoke a node to send unicast data frames to a node that is not there – this can be exploited by a simple application running on a device. Finally, a completely passive attack provoking denial of service at the sender side can be realized by the destination of a unicast transmission by simply not acknowledging another node's unicast transmission. To this end, the impact of the effect of head of line blocking has been considered to be no worse than reducing the available goodput over the wireless channel. This way, the effect was being widely ignored by standardization bodies.

However, for safety applications depending on low latency communication, and even more so in highly dynamic environments such as VANETs, the effect of head of line blocking can be disastrous. In case of triggering this effect, unnecessary delays of Cooperative Awareness Messages (CAMs) or Basic Safety Messages (BSMs) can easily lead to negative influences of safety applications. In particular the effect is

frequently triggered in networks of high topology dynamics when nodes attempt unicast transmissions to other nodes. The main cause for this is that nodes wrongly consider other nodes to still be neighbors. Such frames will never be acknowledged, remaining at the head of the transmit queue of the sender until they expire. Moreover, retransmissions in such cases do not make any sense, since they just contribute to a higher channel load without being beneficial for other nodes.

As outlined before, head of line blocking not only delays the unicast transmission for which no ACK frame was received, but negatively affects all frames waiting in the same transmit queue within the Enhanced Distributed Channel Access (EDCA) system of WLAN [22]. Which queue is blocked depends on the Access Category assigned to the frame, which itself is a decision of the application accessing the MAC. This way, with only four categories defined in WLAN, a large number of different applications will likely share a single queue. Therefore, a single blocked queue has a negative impacts on a multitude of different applications accessing that particular queue, e.g., safety related applications. Without additional information, a sender can also not determine whether the receiver suffers from interference that keeps it from replying with ACKs frames, or whether ACKs are suppressed, making passive attacks hard, if not impossible, to detect.

Moreover, the impact of head of line blocking is also long-lasting. Often, the destination node is simply no longer a neighbor and remains permanently unreachable, e.g., due to radio signal shadowing [72] when highly dynamic scenarios are taken into account. Consequently, this causes the transmit queue of a sender to block until the maximum number of retries have been exceeded. This behavior wastes channel capacity and keeps other nodes from transmitting, and (even worse) keeps the same node from transmitting potentially safety-critical information. Overall, the attainable performance of the wireless network is degraded when exponential backoff is used in situations where it is not of any help. This leads to our conclusion that unicast communication in IEEE 802.11p for highly dynamic topologies needs to be considered harmful.

In this chapter we build upon our earlier work [25] and point out a way towards a general solution. This is based on an investigation of the effects of head of line blocking in more breadth and depth: We expand our focus from specialized hardware and settings of VANET Field Operational Tests (FOTs) to that of regular commercial off-the-shelf WLAN adapters by investigating both, more general metrics to study the true impact on the application layer, as well as more specific metrics of the reported effects. Another important aspect is that we take great care to cross-validate every step in our investigation between analytics, computer simulations, and hardware experiments. Based on an experimental and analytical study we investigate the behavior of commodity WLAN cards under active attacks. As a possible algorithmic

solution to the problem of head of line blocking we also investigate a rather simple protocol that helps overcoming these issues.

We summarize our main contributions as follows:

- First we investigate the impact of head of line blocking in a small and static network – analytically, in computer simulations, and in hardware experiments (Section 3.3);
- Next, we continue our study by evaluating the backoff behavior of commodity WLAN cards when subjected to selective jamming of acknowledgments. This wave we validate experimental results against analytical evaluations (Section 3.4);
- Finally, we assess the macroscopic view in presence of head of line blocking for a full VANET application scenario incorporating a highly dynamic network (Section 3.5).

3.2 Preliminaries

Information exchange for comfort or efficiency applications in the VANET domain commonly involves vehicles communicating with either a dedicated vehicle, a Roadside Unit (RSU), or a gateway. Applications often use a connection-like oriented communication pattern by taking advantage of unicast routing over multiple hops [49], [110]. Many of them were originally developed for Mobile Ad Hoc Networks (MANETs), and parts of them can be applied to VANETs as well. Besides comfort and efficiency applications, unicast communication is also often used in the literature for specific VANET applications like Geocasting and platooning [111], [112]. Following this up, several detailed surveys on unicast routing protocols for VANETs can be found in the literature: Li and Wang [113] present an overview about different routing strategies and name popular routing protocols according to their communication type. Bernsen and Manivannan [114] classify and characterize available unicast routing protocols for VANETs and provide a qualitative comparison among those. Sichitiu and Kihl [115] focus more on the taxonomy of VANET applications and study the requirements from an underlying network perspective. These works emphasize the frequentness of the unicast communication principle even in protocol designs targeting highly dynamic networks.

Moreover, a number of protocol designs explicitly target, or even rely on, unicast communication patterns and its acknowledgment scheme to support multi-hop routing, as well as single-hop transmissions. In particular, approaches have been presented to mitigate packet duplication of unicast routing protocols introduced by the two (per-hop and end-to-end, respectively) recovery mechanisms of the MAC

layer together with the routing protocol [116]. Similarly, Han et al. [117] take advantage of the MAC retry mechanism based on ACKs and propose an improved retry mechanism based on NACKs (negative acknowledgments). This allows an application layer message to be quickly and repeatedly retransmitted up until it is eventually (successfully) received by its indented destination. To follow that up, Xie, Ho, and Xie [118] present a two-dimensional Markov chain model based on the work of Bianchi [3] to investigate the delay of channel access by using a stochastic road traffic model. A central assumption for reliable VANET protocol design is the need for unicast communication implying retransmissions performed at the MAC layer.

When IEEE 802.11 was designed many years ago, an exponential backoff strategy for unsuccessful unicast communication triggered by not received acknowledgments was chosen to solve channel congestion problems. It was assumed that the node topology is relatively static, leading to the conclusion that the most common causes for not received acknowledgments were hidden terminal situations and, more importantly, a congested and overloaded channel.

However, in our work we show that for VANETs in particular, and highly dynamic scenarios in general, this assumption does not hold anymore. Consequently, we show that reliable unicast communication drastically lowers the performance of VANETs when unicast packets are sent to nodes that are out of range, which speaks against its original idea of unicast – providing *reliable* communication. In realistic simulation studies we show that this is a common occurrence in VANETs.

3.3 Small and Static Networks

As a first step we investigate the impact of the discussed head of line blocking effect in a small and static network which mimics topology dynamics. We do this by focusing on *(a)*, analytically derivations, *(b)* experiments with off-the-shelf and FOT-ready WLAN cards, and finally *(c)* computer simulations.

3.3.1 Analytical Evaluation

Without loss of generality, in the following we focus on an OFDM PHY using 10 MHz bandwidth as specified in the current version of the IEEE 802.11 standard [22]. Following the values and the formalism introduced in the standard, we adopt the PHY timing parameters as follows: $T_{\text{preamble}} = 32\,\mu\text{s}$, $T_{\text{signal}} = 8\,\mu\text{s}$, and $T_{\text{sym}} = 8\,\mu\text{s}$. MAC parameters are also configured according to the default values proposed by the standard, in particular to $t_{\text{SIFS}} = 32\,\mu\text{s}$, $t_{\text{slot}} = 13\,\mu\text{s}$, and $t_{\text{rx\_delay}} = 49\,\mu\text{s}$. Moreover, we assume that the RTS threshold is set above the frame size, such that no RTS/CTS

management frame exchange is invoked. Further, we assume empty EDCA queues and an idle channel.

The time it takes to transmit data is derived according to the PLME-TXTIME.confirm primitive outlined in the standard [22, Section 18.4.3]. For transmitting headers and payload of size l at 6 Mbit/s (thus $N_{\mathrm{DBPS}} = 48\,\mathrm{bit}$), this time can be calculated as outlined in Equation (2.3). For a broadcast frame with a payload of $l = 2400\,\mathrm{bit}$, we obtain $t_{\mathrm{tx}}(2400\,\mathrm{bit}) = 448\,\mu\mathrm{s}$. Similarly, for $l = 112\,\mathrm{bit}$, the size of an ACK frame, we obtain $t_{\mathrm{tx}}(112\,\mathrm{bit}) = 64\,\mu\mathrm{s}$.

For reliable unicast transmission of a frame, the frame exchange sequence is as follows: *send data, wait for a SIFS, and send ACK*. Therefore, we can derive the lower bound for the duration of a unicast transmission (which might be blocking a queue) if the channel has been idle for some time and the transmission is immediately acknowledged as

$$t_{\mathrm{best}} = t_{\mathrm{tx}}(2400\,\mathrm{bit}) + t_{\mathrm{SIFS}} + t_{\mathrm{tx}}(112\,\mathrm{bit}) = 544\,\mu\mathrm{s}. \tag{3.1}$$

When we now focus on the case of a node trying to transmit such a unicast frame to a node that does not exist (or moved outside the communication range), we have to include the time spent for retransmissions. Each of these retries waits for an ACK that does not arrive within $t_{\mathrm{ACK\_wait}}$ and additionally need time spent in backoff. According to the standard [22, Section 9.3.2.8], $t_{\mathrm{ACK\_wait}}$ can be derived as

$$t_{\mathrm{ACK\_wait}} = t_{\mathrm{SIFS}} + t_{\mathrm{slot}} + t_{\mathrm{rx\_delay}} = 94\,\mu\mathrm{s}. \tag{3.2}$$

Backoff times are set to k times t_{slot}, for which the number k is being randomly drawn (incorporating a uniform distribution) from a contention window (CW). This contention window is initially set to CW_{min}. In the worst case, the maximum number is drawn each time. After each unsuccessful transmission (i.e., no ACK was received) the contention window size CW gets updated to $2(CW+1)-1$, upper bounded by CW_{max}. Only when the frame is finally deleted from the transmit queue, in case of successful transmission or exceeded retransmission count, the CW is reset to CW_{min}. This leads to an upper bound of time spent for backoff alone during n attempts to transmit this unicast frame as

$$t_{\mathrm{CW}}(n) = t_{\mathrm{slot}} \times \sum_{i=0}^{n-1} \min\left\{2^i\left(CW_{\mathrm{min}}+1\right)-1, CW_{\mathrm{max}}\right\}. \tag{3.3}$$

Consequently, as backoff values are drawn uniformly from the CW for every transmission attempt, the mean value of the distribution of waiting times can be derived by halving t_{CW}.

However, the channel needs to be idle for at least the duration of an Arbitration Interframe Space (AIFS) before being able to decrement the backoff counter. If we now assume the sender to be operating in OCB mode and using an Access Category AC_BE, depending on the channel busy state it needs to wait for at least AIFSN = 6 slots [22, table 8-106], resulting in

$$t_{\text{AIFS}} = t_{\text{SIFS}} + 6 \times t_{\text{slot}} = 110\,\mu\text{s}. \tag{3.4}$$

The standard [22, pages 1623 and 2425] suggests following default values: $CW_{\text{min}} = 15$, $CW_{\text{max}} = 1023$, and *dot11ShortRetryLimit* = 7 retransmission attempts. For VANETs it has been found beneficial for protocol operation to use this configuration [119]. Surprisingly, in our measurements we discovered that the used wireless cards perform 9 retransmission attempts when no ACK is received (independent of their configuration). Thus, we also follow this and use this value in the following calculations.

In summary, the upper bound for the duration of a unicast transmission is achieved when all following properties hold: the channel just turned idle and the transmission remains unacknowledged for 10 attempts, and each time the maximum backoff time from the CW is chosen. This can be calculated as

$$\begin{aligned} t_{\text{worst}} &= 10\Big(t_{\text{AIFS}} + t_{\text{tx}}(2400\,\text{bit}) + t_{\text{ACK\_wait}}\Big) + t_{\text{CW}}(10) \\ &= 72\,742\,\mu\text{s}. \end{aligned} \tag{3.5}$$

Thus, each unicast transmitted to a node which no longer exists (or the node does not reply with an ACK) blocks *any transmissions* from the same queue for an average of around 40 ms and up to approx. 73 ms. We want to highlight that this effect accumulates if multiple such frames are queued.

3.3.2 Experimental Study

In the following we confirm both the presence and the analytically derived duration of the blocking effect in real world experiments.

As our device on which we performed the experiments, we used an embedded system running Linux 3.18 based on [120] and equipped the system with a Compex WLM200N5-23ESD miniPCI card employing an Atheros AR9220 chipset using the ath9k driver. This setup already has shown to be a reliable basis for building an ITS prototype [7], [43] for FOTs, as it allows tuning the radio to ITS frequencies in the 5.9 GHz band as well as using a bandwidth of 10 MHz as required by IEEE 802.11p. In our previous work [25] we confirm that the effect of head of line blocking is equally present when using specialized equipment especially designed for ITS FOTs worldwide (like the Cohda Wireless MK5).

We modified the Linux kernel to report the time how long each frame was delayed in a transmit queue (from entering into the queue to it being deleted) by using corresponding fields in the radio tap header of each frame. Next, we configured an independent virtual interface configured to monitoring mode to record these statistics. Similarly, the receiver used information made available from a modified kernel to derive the number of backoff slots chosen for a frame. Moreover, an independent monitoring node equipped with a wireless card tracked the channel load $\rho = t_{\text{busy}}/(t_{\text{busy}} + t_{\text{idle}})$ as the fraction of the time the wireless channel was sensed busy.

To investigate the interplay of unicast and broadcast communication modes, we created a process on the device which periodically sends permutations of three independent types of messages to model three different applications sending traffic over the wireless channel. In the following we refer to these applications as App 1, App 2, and App 3. The process enqueues these three messages simultaneously and then waits for the transmission to conclude in order to saturate an otherwise clear channel. The ordering of messages is random, and a sample ordering of messages looks like $(1,2,3),(1,3,2),(2,1,3),(2,3,1),(3,1,2),(3,2,1)$.

The physical layer was configured according to the specifications used in the analytical study: 10 MHz channel bandwidth at 5.890 GHz center frequency, not using RTS/CTS, and transmitting at a rate of 6 Mbit/s. Moreover, the MAC layer was configured to transmit packets using a TXOP value of 0 (one post-transmit backoff after every frame) and Access Category AC_BE which results in an initial contention window size of 15 slots, a maximum contention window size of 1023 slots, and an AIFSN value of 6 slots.

For our *baseline* scenario we performed two different experiments: In the first experiment (denoted as Exp 1), all three applications transmit broadcast packets. In the second experiment (denoted as Exp 2), App 1 was changed to send unicast packets, the remaining applications still send broadcast packets. As an alternative scenario (called *phantom*) we changed App 3 to transmit data to a device that

scenario	experiment	App 1	App 2	App 3
baseline	**Exp 1**	broadcast	broadcast	broadcast
	Exp 2	unicast	broadcast	broadcast
phantom	**Exp 3**	broadcast	broadcast	*lost unicast*
	Exp 4	unicast	broadcast	*lost unicast*
app. retry	**Exp 5**	broadcast	broadcast	*lost unicast*
	Exp 6	unicast	broadcast	*lost unicast*

Table 3.1 – Overview of experiments and their configurations; based on [28]

was no longer there and repeated the previous two experiments (we denote these experiments as Exp 3 and Exp 4). To achieve this, we configured static entries in the ARP tables of the nodes, thus capturing the scenario of a neighbor having existed previously before moving out of reception range. This is a representative case in which a vehicle is trying to send data to a former neighbor, which has been shown in the literature to happen frequently in a VANET [121]. Building upon this scenario, we investigate the results of two more experiments by using a different retransmission strategy (denoted as Exp 5 and Exp 6). In Table 3.1 we outline an overview of all experiments and their configurations.

3.3.3 Computer Simulation

To validate our results in the small and static network we cross-check the analytical and experimental results against a computer simulation consisting of the same scenario. For this we configured simulations in the *Veins* vehicular network simulation framework [122] version 4.4. Details about the simulation framework are outlined in Section 2.4. As Veins in version 4.4 does not provide support for unicast communication employing retransmissions according to IEEE 802.11, we extended the IEEE 802.11p MAC layer to support ARQ according to IEEE 802.11 HCF. In the same way as in the experiments, the MAC layer was configured to send frames using a TXOP value of 0 (meaning one post-transmit backoff after every frame) and Access Category AC_BE (resulting in an initial contention window size of 15 slots, and a maximum contention window size of 1023 slots, as well as an AIFS value of 6 slots to match the settings used in the hardware measurements and analytical evaluations).

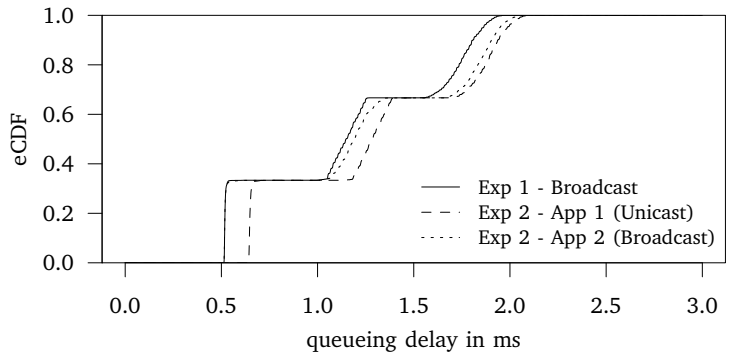
In the simulation model, for all three types of messages, a payload length of 2400 bit was used to saturate an otherwise empty channel. Similar to the experimental study we perform the same set of experiments as outlined in Table 3.1.

3.3.4 Results

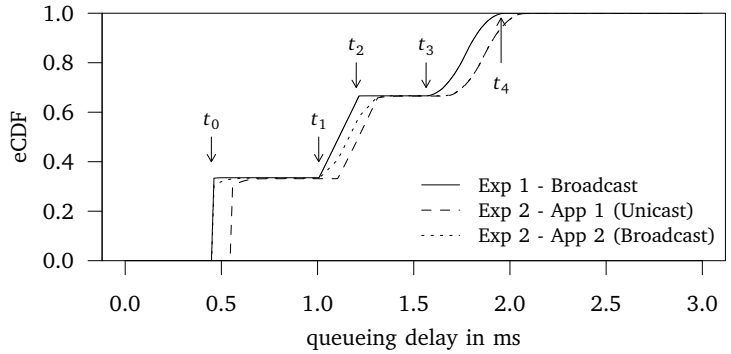
In Figure 3.1 we show the results of analytics, measurements, and computer simulations as empirical Cumulative Distribution Functions (eCDFs). Apart from small delays introduced by the software stack of the measurement setup, queueing delays in the measurements match very well with those obtained in computer simulation, as well as with those derived by analytical calculations: In particular, the delays of frames fall into three clear distinguishable categories according to how many frames (zero, one, or two) were enqueued before.

In the *baseline* scenario (Exp 1), when sending only broadcast frames all data took either

$$t_0 = t_{\text{tx}}(2400\,\text{bit}) = 448\,\mu\text{s} \tag{3.6}$$



(a) measurements (using Atheros AR9220 based WLAN cards)



(b) simulation (using the *Veins* simulator)

Figure 3.1 – TX queuing delay in the *baseline* scenario; based on [28] © 2018 IEEE.

to transmit (if no frame was already queued) or it had to wait for data of one or two of the other applications to be sent.

Further, when waiting for data of one single application, this additional delay is characterized by a uniformly distributed random value of $\mathcal{U}(0, CW_{\text{min}})$ slots spent in post-transmit backoff which results in a uniformly distributed waiting time ranging between

$$t_1 = t_0 + t_{\text{AIFS}} + 0 + t_{\text{tx}}(2400\,\text{bit}) = 1006\,\mu\text{s}, \tag{3.7}$$

$$t_2 = t_0 + t_{\text{AIFS}} + CW_{\text{min}}\,t_{\text{slot}} + t_{\text{tx}}(2400\,\text{bit}) = 1201\,\mu\text{s}. \tag{3.8}$$

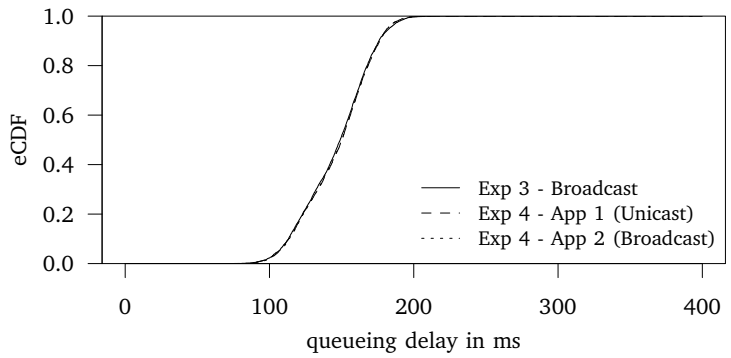
Moreover, when waiting for data of two applications, the delay is characterized by the uniform sum distribution of both backoffs, bounded by

$$t_3 = t_1 + t_{\text{AIFS}} + 0 + t_{\text{tx}}(2400\,\text{bit}) = 1564\,\mu\text{s}, \tag{3.9}$$

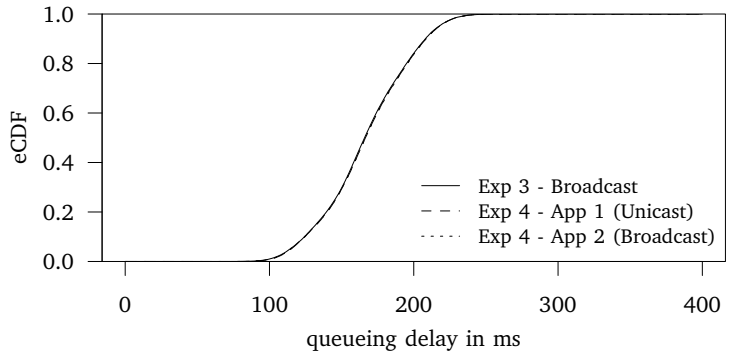
$$t_4 = t_2 + t_{\text{AIFS}} + CW_{\min} t_{\text{slot}} + t_{\text{tx}}(2400\,\text{bit}) = 1954\,\mu\text{s}. \tag{3.10}$$

However, when we change App 1 to perform unicast communication (*baseline*, Exp 2), frames are delayed for the additional ACK frame that needs to be sent (and processed). This does not only induce delays on App 1, which takes longer to send frames, but also for App 2 due to head of line blocking. Still, since unicast frames are acknowledged almost immediately, the head of line blocking issue is of no further consequence for this type of configuration. As the next step we investigate a scenario where the intended receiver is not (or no longer) within communication range of the sender.

In Figure 3.2 we show the effects observed in the discussed scenario of a node not responding to acknowledgments (denoted as *phantom*, Exp 3 & 4). Here App 3 was changed to transmit data to a device that was no longer there representing the scenario of a neighbor having existed previously before moving out of communication range. It can easily be observed that, as already predicted by our analytical



(a) measurements (using Atheros AR9220 based WLAN cards)

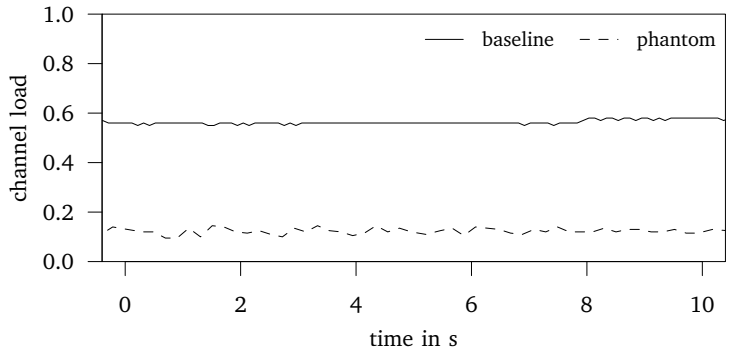


(b) simulation (using the *Veins* simulator)

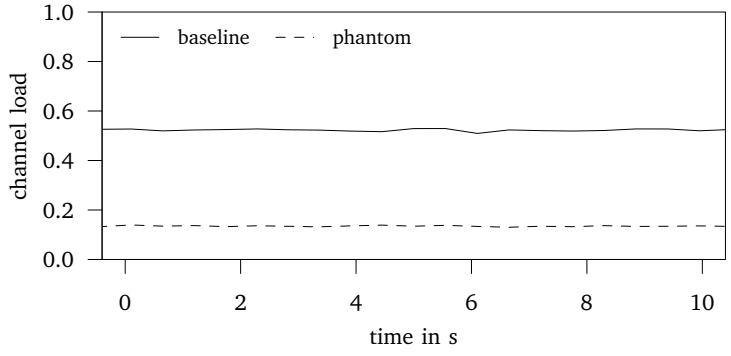
Figure 3.2 – TX queuing delay in the *phantom* scenario; based on [28]

calculations, the missing ACKs of App 3 transmissions had a catastrophic effect on the delay of App 1 and App 2 transmissions. In particular, no relevant difference can be observed between the delay of applications that were generating broadcast frames (Exp 3 and Exp 4, App 2) and those that were generating unicast frames (Exp 4, App 1). The reason for this is that both share a single transmit queue with the frames generated by App 3. For both, missing acknowledgments cause head of line blocking, which increases the time frames spent in the transmit queue until the head of line frame expires. Both broadcast and unicast frames have been queued for a typical duration of 150 ms and delays could exceed 200 ms which is much worse than the demands of many VANET applications [123], [124].

When looking at the channel load, a further consequence of this head of line blocking problem is revealed: Figure 3.3 illustrates that, as expected, in the baseline scenario the applications are able to saturate the channel. However, when looking at the results for the *phantom* scenario, for both computer simulation and actual measurements, missing acknowledgments cause the channel load to drop down to values of around 15 % to 20 %.



(a) measurements (using Atheros AR9220 based WLAN cards)



(b) simulation (using the *Veins* simulator)

Figure 3.3 – Channel load in the *baseline* and *phantom* scenarios; based on [28]

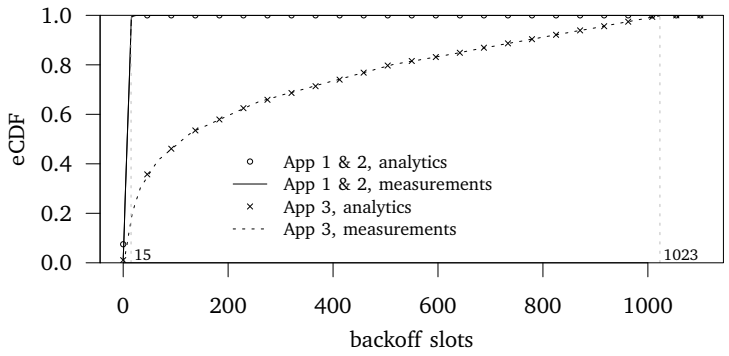


Figure 3.4 – Distribution of chosen backoff slots for each application in the *phantom* scenario; results for hardware measurements and analytical calculations; based on [28] © 2018 IEEE.

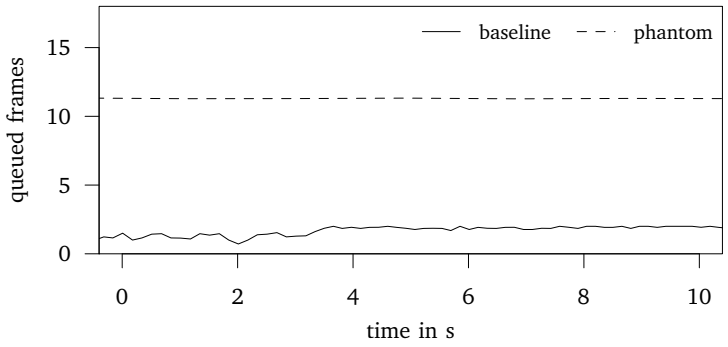
The reason for this can be seen in Figure 3.4, in which we plot the chosen backoff slots for each application and compare these measurements to Monte Carlo simulations following our analytical derivations. As analytics predict, the contention window in measurements increases according to the exponential backoff algorithm when no acknowledgment is received for App 3. For the other two applications, the chosen backoff slots follow a uniform distribution between 0 and 15 slots.

The effect of head of line blocking is also reflected in the queue utilization, shown in Figure 3.5. In both, computer simulations and actual measurements, the queue fill levels in the baseline scenario stay near zero, since traffic is non-bursty and the offered load is below the capacity of the channel. However, the queue levels increase massively in the phantom scenario, which is caused by HOL blocking of frames retransmitted due to missing acknowledgments, and of course the exponential backoff procedure invoked by the lower MAC. These results match our expectations and give a good impression about the negative impact of failed unicast transmissions on the networking performance.

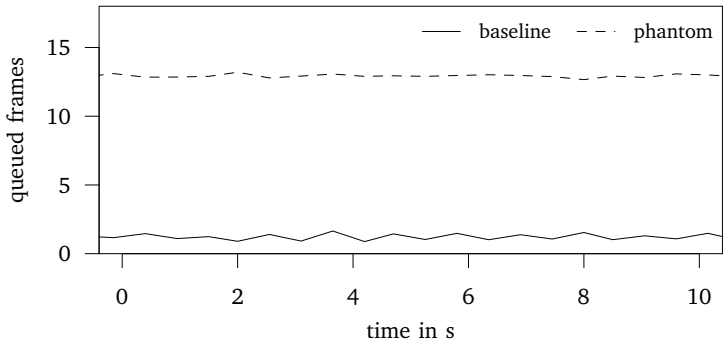
In summary, our experiments, analytical calculations, and simulations show the grave effect that head of line blocking, provoked by unicast frames addressed to a former or not responding neighbor, has on the delay of broadcast frames.

3.3.5 Towards a Solution

As shown in the previous sections, failed unicast transmission have a negative impact on networking performance. A possible approach to avoid this head of line blocking problem is to lower the number of retransmissions at the MAC in case of failed unicast transmissions. In order to maintain the same level of reliability, the application layer needs to take care about necessary retries. In this section we present an approach



(a) measurements (using Atheros AR9220 based WLAN cards)



(b) simulation (using the *Veins* simulator)

Figure 3.5 – Number of queued frames in the *baseline* and *phantom* scenarios; based on [28] © 2018 IEEE.

based on such retransmission at application-layer side which avoids HOL blocking for failed unicast transmissions, but still retransmits those frames.

In the following we outline the algorithm which is performed for each transmitted unicast frame:

- Whenever the transmission was successful (we received an ACK after a SIFS) the frame gets deleted from the queue head. This is the usual behavior when invoking IEEE 802.11 unicast operation.

- In the case of an unsuccessful transmission (that means we did not receive an acknowledgment within the interval $t_{\text{ACK\_wait}}$) we do not increase the contention window, but keep it at CW_{min}. Further we do not perform any immediate retransmission. However, we reinsert the frame at the tail of the queue, that means after all other queued frames for that particular access category. When the retransmission count for that particular frame exceeds the configured maximum number of retries, we drop the frame.

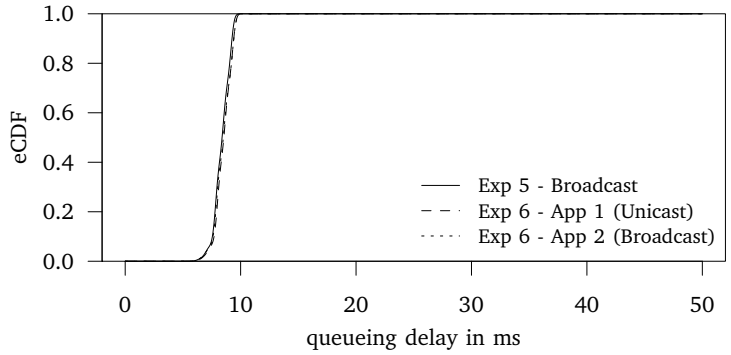
To achieve this goal we configure the maximum number of retries for failed unicast transmissions at the MAC layer to zero, however we still wait for an ACK to be received. Consequently, we do not perform exponential backoff for missing acknowledgments. Instead, we retransmit the packet at the application layer. This behavior will intuitively lead to lower delays for the remaining queued frames since no retransmissions on the MAC layer are performed which would cause HOL blocking, as well no exponential backoff is performed which lowers the attainable throughput on the wireless link.

We evaluate this approach in terms of simulations by using the scenario denoted as *app. retry* (Exp 5 & 6) outlined in Table 3.1. These experiments are otherwise identical to Exp 3 & 4: Again we configure three independent applications which generate messages in random order. App 3 transmits packets as unicast to a former station that no longer exist – therefore no ACKs will be received for those frames.

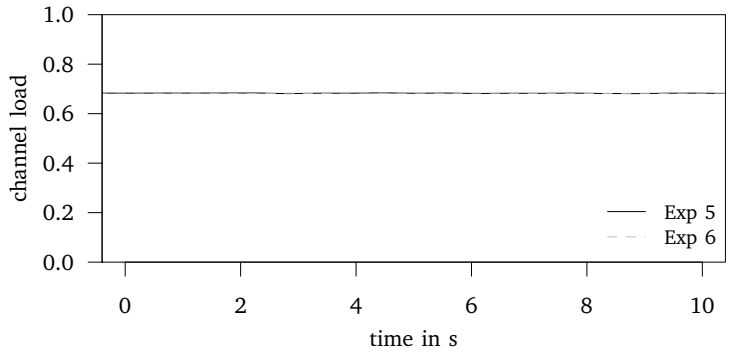
In Figure 3.6a the time those frames spent in the EDCA queue is shown. Due to the fact that the head of line blocking problem is effectively circumvented by our presented approach, no appreciable difference can be seen between the unicast and broadcast experiments. Moreover, in comparison to the default approach in which frames are kept at the queue head until its retransmit count is exceeded, the delay of all frames is now tremendously lowered. The observed delays for both unicast and broadcast frames are around 8.5 ms (down from the approx. 150 ms as shown in Figure 3.2). Consequently, our presented solution also increases the channel load (shown in Figure 3.6b) to values comparable to those measured in the baseline scenario in which no acknowledgments are missing (see Figure 3.3). As every failed transmission in our approach is still retried as often as in the *phantom* scenario, the queue fill level remains comparable as outlined in Figure 3.6c. Moreover, in case of missing acknowledgments are not caused by an absent receiver, and thus a retransmission could be actually beneficial, additional delays are likely induced by the frame taking a round trip through the application layer. A side effect of our proposed scheme is frame reordering, for which this approach most probably negatively influences higher layer transport protocols like TCP – thus requiring special attention [125]. As a conclusion, our approach can serve, due to its simplicity, as a baseline for comparisons and a basis for future work.

3.4 The Special Case of Active Attacks

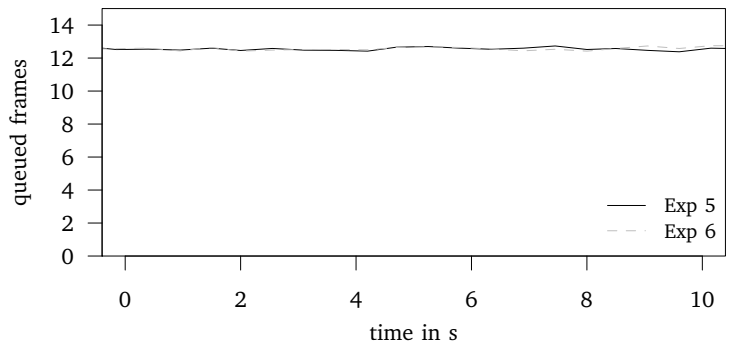
We evaluate the impact of failed unicast transmissions on the goodput of traffic flows by actively jamming MAC acknowledgments. This is a representative case for active denial of service attacks on a wireless network. Similar experiments [126] were performed in the past by taking advantage of custom developed hardware.



(a) HOL blocking delay



(b) channel load



(c) queue fill state

Figure 3.6 – Application layer retransmissions (using the *Veins* simulator). Note that all lines are overlapping; based on [28] © 2018 IEEE.

It was found that different vendors for IEEE 802.11b wireless cards show different performance which is caused by non-standard compliant backoff procedures. In this section, instead, we focus on the impact of not received acknowledgments for unicast transmission on the backoff behavior of commodity WLAN hardware. Further we evaluate to what degree the channel load and goodput are affected in such scenarios.

We build upon the work of Vanhoef and Piessens [127] where the authors present firmware extensions for popular USB WLAN sticks based on the ath9k-htc driver to perform low-layer attacks. In particular, they present a methodology to jam the wireless channel, in which the device stops decoding a frame during reception, and immediately starts sending a custom short frame – thus causing interference at a potential receiver. With high probability this will lead to a wrong Frame Check Sequence (FCS) of the frame at the receiver, thus the receiver cannot further process this frame anymore. In our experiments we measure a minimum delay for the jamming device of around 126 µs from decoding the first Byte up to the point where we are able to start sending our jamming frame. This time is too long to reliably jam MAC acknowledgments of 112 bit size, which is our intended case to trigger the head of line blocking effect. To circumvent this issue, we detect the frame for which we expect an acknowledgment to be sent by the receiver, and then perform busy waiting up to the point where we expect the acknowledgment to be transmitted. At this time we then send our short jamming frame causing the acknowledgment frame to be interfered. We perform all these operations on the firmware of the USB WLAN dongle for which a configuration interface from the host computer exists. With this interface we can configure parameters to define the length and modulation and coding scheme of the jam signal, as well as which frames to detect, and the time offset between frame detection and transmission of the jam signal.

3.4.1 Experiment Setup

For our experiments we configure a scenario consisting of a single sender and a single receiver, where each of them are equipped with a wireless card of type Compex WLM200N5-23ESD employing an Atheros AR9220 chipset together with the ath9k driver again running Linux 3.18. We build upon the work of Lisovy, Sojka, and Hanzálek [120] and configured the devices in OCB mode transmitting on channel 178 (5890 MHz). Our jamming system lacks support of 10 MHz channel bandwidth, thus we use 20 MHz channel bandwidth for all nodes. Therefore, keeping $N_{\mathrm{DBPS}} = 48$ bit results in a bit rate of 12 Mbit. We want to note that our findings are not tight to a specific channel bandwidth, and results will be comparable for different channel bandwidths. Moreover we configure the MAC to use contention window settings of $CW_{\mathrm{min}} = 15$ and $CW_{\mathrm{max}} = 1023$ slots, as well as an AIFSN value of 6. Similar to the experiments done before (see Section 3.3.2) RTS/CTS operation is disabled.

We configure the short and long retry limits to 7 and 4 retransmission per frame, respectively. However, in our experiments we found that the wireless card performs 20 retries regardless of configuration. Further, our findings show that the wireless hardware does not invoke exponential backoff for the case that ACKs are received but cannot be decoded as we detail in Section 3.4.3.

In addition, we use a dedicated node which continuously measures the channel load. Our jamming device is equipped with a Netgear WNDA3200 USB WiFi dongle which we configured similarly to the sender and receiver.

For evaluation, we use the *iperf* tool in UDP mode to saturate the channel and measure goodput on the wireless link between the receiver and the sender. This gives us an impression of application layer performance. We set the packet length of iperf to $l_{\text{payload}} = 800\,\text{B}$ (6400 bit), which results in a total number of 6912 bit to be transmitted over the air. This value includes UDP header (64 bit), IP header (160 bit), LLC header (64 bit), as well as IEEE 802.11 header (224 bit).

In particular, we investigate three different scenarios: First, the *baseline* scenario: here no jamming is performed and all acknowledgments are received. Second, the *jammed* scenario: here acknowledgments are selectively jammed. Finally, the *std* scenario: here acknowledgments are selectively jammed, but we assume the hardware follows the behavior proposed by IEEE 802.11 (7 retries for each frame, exponential backoff).

3.4.2 Analytical Evaluation

At first we calculate the expected goodput (application layer performance) of unicast transmissions on a wireless link for a specific packet size. We do this for both, successful and failed transmissions.

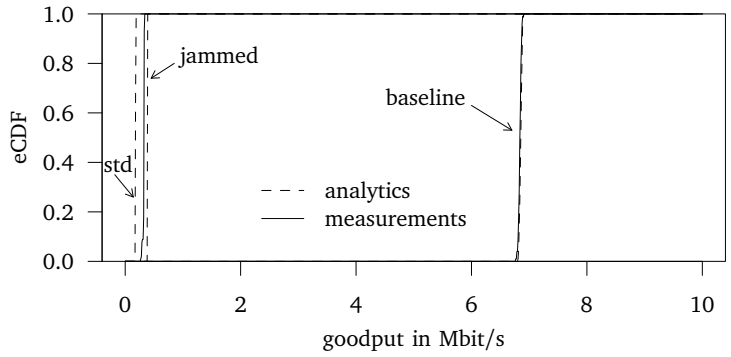


Figure 3.7 – Distribution of measured goodput μ, compared with analytical predictions; based on [28] © 2018 IEEE.

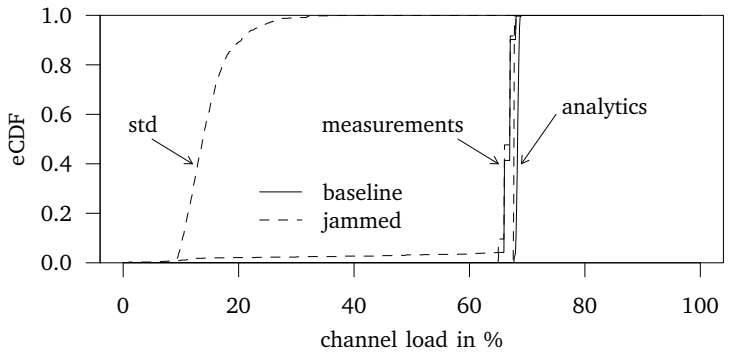


Figure 3.8 – Distribution of measured channel load ρ, compared with analytical predictions; based on [28] © 2018 IEEE.

In the following we use the notation and derivations introduced in Section 3.3.1. However, we substitute t'' for t and T'' for T in order to indicate the use of timing parameters for 20 MHz channel bandwidth. Specifically, we use $T''_{\text{preamble}} = 16\,\mu\text{s}$, $T''_{\text{signal}} = 4\,\mu\text{s}$, $T''_{\text{sym}} = 4\,\mu\text{s}$, $t''_{\text{SIFS}} = 16\,\mu\text{s}$, $t''_{\text{slot}} = 20\,\mu\text{s}$, and $t''_{\text{rx\_delay}} = 25\,\mu\text{s}$.

Based on these values, the transmission time of a frame of 6912 bit can be derived analogous to Equation (2.3) and leads to $t''_{\text{tx}}(6912\,\text{bit}) = 600\,\mu\text{s}$. Consequently, the time to transmit an ACK frame of length 112 bit takes $t''_{\text{tx}}(112\,\text{bit}) = 32\,\mu\text{s}$.

The time for arbitration for which the channel needs to be idle in order to start decrementing the backoff counter is calculated analogous to Equation (3.4) which leads to $t''_{\text{AIFS}} = 136\,\mu\text{s}$.

After the transmission of a unicast frame ends, the sender waits for $t''_{\text{ACK\_wait}} = 61\,\mu\text{s}$, obtained analogous to Equation (3.2), to detect the PHY-RXSTART.indication. This directive defines the point in time at which the signal is started being decoded. For a successful detection of this indication the node waits upon its completion and checks if the received frame is a valid acknowledgment for the unicast data frame sent before. When no indication is detected, or if the frame is not detected as a valid acknowledgment to the unicast data frame, the procedure for exponential backoff shall be invoked as proposed in the standard [22, Section 9.3.2.8].

For the *baseline* scenario (here we use no jamming), the average time for channel access and transmission of a frame is derived as

$$\begin{aligned} t_{\text{busy}} &= t''_{\text{tx}}(6912\,\text{bit}) + t''_{\text{tx}}(112\,\text{bit}) = 632\,\mu\text{s} \\ t_{\text{idle}} &= t''_{\text{AIFS}} + \frac{1}{2}\, CW_{\text{min}}\, t''_{\text{slot}} + t''_{\text{SIFS}} = 302\,\mu\text{s}. \end{aligned} \tag{3.11}$$

Consequently, the average goodput can be calculated by using the packet size without lower layer headers as

$$\mu_{\text{baseline}} = \frac{l_{\text{payload}}}{t_{\text{busy}} + t_{\text{idle}}} \simeq 6.85\,\text{Mbit/s}. \tag{3.12}$$

Next, the utilization of the wireless channel is derived as

$$\rho_{\text{baseline}} = \frac{t_{\text{busy}}}{t_{\text{busy}} + t_{\text{idle}}} \simeq 67.7\,\%. \tag{3.13}$$

For the *jammed* scenario (in particular assuming the nonstandard behavior of our hardware), these values change to

$$\begin{aligned} \overline{t}_{\text{busy}} &= 21\Big(t''_{\text{tx}}(6912\,\text{bit}) + t''_{\text{tx}}(112\,\text{bit})\Big) = 13\,272\,\mu\text{s} \\ \overline{t}_{\text{idle}} &= 21\Big(t''_{\text{AIFS}} + t''_{\text{SIFS}} + \frac{1}{2}t''_{\text{slot}}CW_{\text{min}}\Big) = 6342\,\mu\text{s} \\ \mu_{\text{jammed}} &\simeq 0.326\,\text{Mbit/s} \\ \rho_{\text{jammed}} &\simeq 67.7\,\%. \end{aligned} \tag{3.14}$$

In case of the *std* scenario (in which we use the expected reaction to jamming), we instead derive

$$\begin{aligned} \hat{t}_{\text{busy}} &= 8\Big(t''_{\text{tx}}(6912\,\text{bit}) + t''_{\text{tx}}(112\,\text{bit})\Big) = 5056\,\mu\text{s} \\ \hat{t}_{\text{idle}} &= 8(t''_{\text{AIFS}} + t''_{\text{SIFS}}) + \frac{1}{2}t''_{\text{CW}}(8) = 31\,696\,\mu\text{s} \\ \mu_{\text{std}} &\simeq 0.174\,\text{Mbit/s} \\ \rho_{\text{std}} &\simeq 13.76\,\%. \end{aligned} \tag{3.15}$$

3.4.3 Experimental Evaluation

As a next step we now compare our analytical calculations to real world experiments by using iperf to transmit 800 B UDP packets on the wireless channel in order to saturate an otherwise clear channel. The experiments were performed for both configurations: in a baseline scenario without using a jamming device, and while jamming ACKs. For evaluation purposes for each transmission we track the number of chosen backoff slots by modifying the Linux kernel running on the receiver. In particular that works as follows: We know all inter-frame spaces and signal timings, thus the only unknown value is the time spent for decreasing the backoff counter to zero, which follows an uniform distribution and can be derived by subtracting all known timings from the time interval between receiving the current frame and the previous frame. We note here that this approach only works when packets are

generated continuously on the sender side, thus when fully saturating the wireless channel. This is what we do by using iperf in UDP mode.

For our evaluations we measured the distribution of chosen backoff values by the sender as explained above. We expect that for the baseline scenario the backoff values are uniformly distributed over the range of CW_{min} (though we observed a slight offset of almost one slot which is likely due to measurement inaccuracies). When using our jamming system to interfere the ACK of each unicast frame, our expectation is that the exponential backoff procedure gets invoked for which the contention window is increased. However, results show that the chosen backoff values do not increase and stay at very similar values to the baseline scenario. The cause of this is the – to some degree surprising – behavior of the wireless card: Our measurements show that when it received incomplete (jammed) ACKs, the sender performs 20 retries of each unicast frame. Moreover, the sender never increased the contention window between each retransmission. The outcome of not doubling the contention window in case of no valid acknowledgment is received is confirmed with the backoff behavior of different Atheros chipsets [117]. We note that (as shown in Section 3.3.2) if no valid ACK is received and the wireless channel is idle during the ACK duration, the behavior of the Atheros AR9220 chipset is as expected and the exponential backoff operation is performed.

The impact of not using exponential backoff among retransmission of unicast frames when acknowledgment frames are jammed can also be observed in the goodput metric. As shown in Figure 3.7 the measurement results together with results from Monte Carlo simulations follow the derivations of Section 3.4.2. The measurements match our calculations quite well, which holds for both the baseline scenario, in which we obtain around 6.85 Mbit/s, and the jamming scenario when no exponential backoff is performed resulting at around 326 kbit/s. In addition, we also observe that the expected goodput when performing exponential backoff using 7 retries (denoted *std*) is lowered towards 174 kbit/s.

As a final metric we also recorded the channel load for both scenarios. Figure 3.8 shows that our measurements again fit our analytical findings. Between the baseline scenario and the jamming scenario the channel load does not change significantly. This is a confirmation that the wireless cards under test do not perform the exponential backoff procedure for retransmissions as required by IEEE 802.11 for the case of not decodable acknowledgments.

3.5 Large and Dynamic Networks

As we studied the impact of head of line blocking in static and controlled environments, we now evaluate the implications of this effect in a highly dynamic network.

Therefore we conducted computer simulations of a large and realistic VANET. Our setup consisted of a large number of vehicles running a typical protocol for VANETs which could be switched between using broadcast or (reliable, thus using ACKs) unicast communication. To achieve this, in the Veins simulation we enabled the connection to the microscopic road traffic simulator SUMO (version 0.25.0) which allows modelling realistic road traffic, see Section 2.4 for details.

Specifically, we configured a freeway scenario with a length of 7 km which represents a prototypical VANET scenario [9]. To avoid border effects we performed network simulation only in the center 5 km of the scenario. This 1 km border at each end of the freeway let the vehicles speed up and use realistic mobility patterns. Moreover, we configured three different traffic densities on the freeway: 18 veh/km which represents a very low traffic density scenario (thus having only a few available neighbors which leads to very challenging topology dynamics as we will see in the results); 55 veh/km which represents off-peak traffic on the freeway (low density, thus an in-between case); and finally 169 veh/km which represents a high traffic density on a very busy freeway (this is characterized by a very high number of neighbors and therefore a challenge in terms of channel load as we will see in the results). We model road traffic by sampling from a distribution of five different vehicle types. Two of these types are trucks, and three of these types are cars where both model different kinds of drivers.

We perform result collection within a Region of Interest (ROI) of 3 km in order to not be influenced by border effects of the wireless communication. To this end we configured a warm-up period of the simulation of 289 s in which the freeway gets filled with vehicles. Another 11 s of warm-up time is used for the networking protocols to get into a steady state. Upon finishing these 300 s we start to collect results. We plot for our evaluations the mean value together with the 95 % confidence interval for all our experiments. Please note that these confidence intervals are sometimes very small. To get statistically significant results we repeat each simulation setup at least 50 times with different seeds for the pseudo random number generators used for modeling road traffic and network communication.

In our experiments we show the impact of an application employing reliable unicast communication in a VANET by using a simple neighbor management process which informs a Geocasting protocol. This Geocasting protocol is destined to disseminate information among vehicles. Possible examples for such information might be traffic reports or specific knowledge about certain active road works. We abstract this information in form of *information items* and each of them is simply represented as a block of bits.

To perform neighbor management, each vehicle transmits a beacon at a frequency of 1 Hz by using broadcast communication and maintains with this transceived information a simple 1-hop neighbor table, similar to the process which we explain

in Section 4.4.1 with the difference that we now do not transmit and use 2-hop neighbor information. Upon reception of such a beacon by a vehicle v transmitted from another vehicle u, v adds u to its neighbor table $\mathbb{N}$. Whenever two successive beacons of a vehicle are not received (in our case after 2 s) this particular node is removed from the neighbor table. We perform this update of the neighbor table right before information from the neighbor table is used.

Moreover, the Geocasting protocol builds on information provided by these neighbor tables. The Geocasting protocol is similar to the one presented in Section 4.4.4: First, every node maintains a knowledge base which consists of arbitrary entries, each having geographic constraints. Further every entry has an expiration time. Second, a digest among neighbors is used to exchange information stored in these knowledge bases. Further, information of entries in these knowledge bases can be requested and received by each node. In particular, the protocol works as follows: Upon discovering a new neighbor u by vehicle v, it makes a probabilistic decision if this new neighbor should be informed about stored information in the knowledge base. This probability $p = 1/(\text{new neighbors per s})$ is used by v and decides whether node u will be informed about the active events stored in the knowledge base of v. If the node decided to do so, v sends a small *digest* which includes the fingerprints of all stored events in the knowledge base, limited by the maximum frame size.

If node u receives such a digest, it immediately responds with a *data request* where it includes the fingerprints of information the node is interested into, which we call missing entries. An event is considered as missing if the vehicle is driving towards the destination direction, or if the distance between u and the entry's destination position is lower than the distance between v and the entry's destination position. Thus, a node only marks an entry as missing if it is closer to the entries destination position than the node which offers the entry, or the vehicle in question is driving towards the destination.

A node v, which receives a data request from u, will construct and send a *data packet* to u containing all requested information (which was marked as missing by u). This is again limited by the maximum frame size. This data packet containing the actual information which is stored in the knowledge base can be overheard by all other neighbors by using a monitor interface connected to the transceiver. Each node w which receives this information updates its knowledge base. Finally, w iterates over all neighbors n available in its neighbor table $\mathbb{N}$ and for each neighbor takes a probabilistic decision with $p = |\mathbb{N}|^{-1}$ to decide whether to send a digest to node n or not.

We generate new information items at a rate of 4 Hz in the knowledge base of vehicles at each end of the ROI for our simulation experiments. Moreover, we present results for information item generation rates of 1 Hz and 10 Hz as well. We configured the information items' destination position to be at the opposite end of

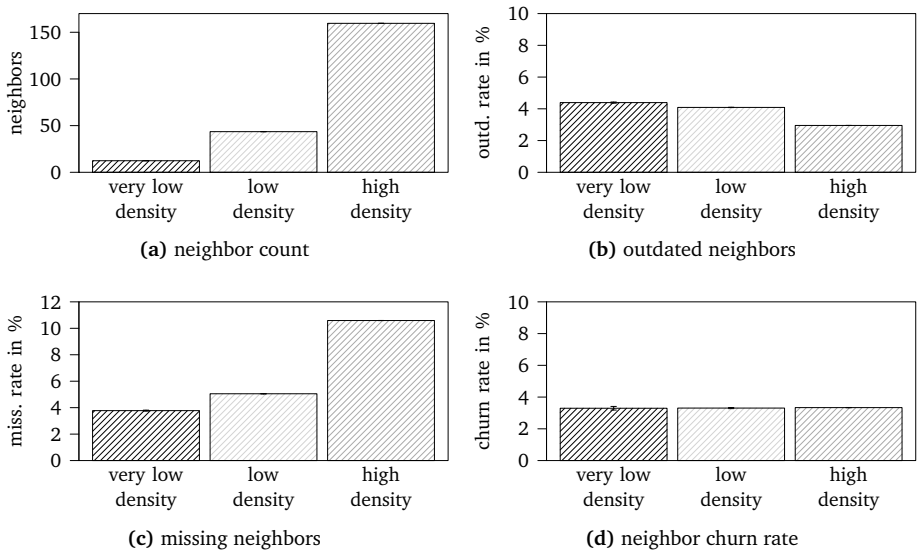
the ROI. That means that each message has to be disseminated through the whole network. After a simulation has reached a steady state (thus, after the warm-up phase has finished), we track each information item as it traverses the network and record the delay each node measures from generation of this particular information item until successful reception. We collect results in our simulations for 5 s.

Additionally, we configure neighbor beacons to use a different EDCA queue for transmission than the digest packets, data request packets, and data packets of the Geocasting protocol. We choose this configuration in order that the applications do not influence each other in terms of head of line blocking as outlined in Section 3.3. In particular, we selected AC_BE with an AIFS value of 6 slots for neighbor beacons, and AC_BK with an AIFS value of 9 slots for the rest. Further, the values for CW_{min} and CW_{max} for both EDCA queues are set to be 15 and 1023 slots respectively. We configured the number of retransmission for reliable unicast communication to 7 retransmissions. We use a maximum frame size of 1024 B, and configured an information item in the knowledge base to take 64 B, and a digest to take 8 B per entry.

3.5.1 Neighbor Management and Topology Dynamics

In the following we report on the performance of the neighbor table. A more detailed overview and investigation of neighbor table performance in VANETs can be found in Sections 4.5 and 4.5.3.1. An important metric to assess the performance of neighbor table maintenance and topology dynamics is the set of 1-hop neighbors which are known to each node, in particular its correctness. Thus we compare the neighbor information against an oracle which derives the ideal neighborhood of a node according to a unit disk model. We use the 99 % quantile of distances of successful frame transmissions to determine the distance of nodes to be treated as 1-hop neighbors. This allows us to derive the fraction of missing and outdated neighbors compared to the oracle which is representative of the quality of the maintained neighbor information. Additionally, we measure the neighbor churn rate which describes how many 1-hop neighbors were deleted from the neighbor table per second. This deletion is mainly caused by either lost beacons or because the node moved outside the communication range. These evaluations allow us to get an indication about the stability of neighbor tables which then will influence the performance of the Geocasting protocol, as we will show later in this chapter.

We obtain a mean value of around 12, 44 and 160 neighbors for each vehicle for the very low, low, and high density scenario, respectively, as can be seen in Figure 3.9a. This is, of course, no indicator of how accurate the neighbor information is. Therefore, we investigate the fraction of outdated (Figure 3.9b) and missing (Figure 3.9c) neighbors in comparison to an oracle. This way we obtain around



(a) neighbor count **(b)** outdated neighbors

(c) missing neighbors **(d)** neighbor churn rate

Figure 3.9 – Neighbor table performance for different traffic densities; based on [28] © 2018 IEEE.

4 % outdated and 5 % missing neighbor information for the low density scenario. For the high density scenario the values change to around 4 % outdated and 11 % missing information respectively. Only in the very low density scenario the value for the outdated neighbor fraction stayed at around 4 % and went slightly below 4 % for the missing neighbor information. We now have a look at the mean churn rate of neighbors shown in Figure 3.9d to investigate the reasons for this observations. As indicated by the plot the value remains constant for all three traffic densities meaning that the neighbor management process does not cause congestion on the wireless channel.

In summary, all results show that even in this simple freeway setting without buildings, the high dynamics of the network topology are non-negligible.

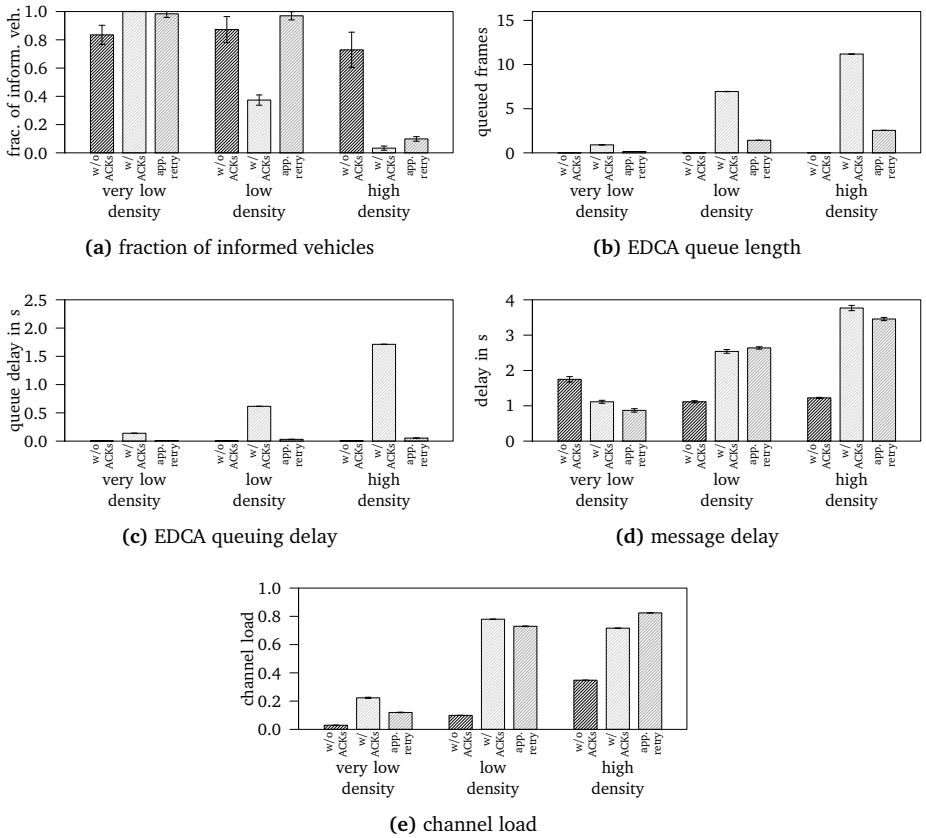
3.5.2 Application Performance

As next step we investigate the performance of the Geocasting application by using three different configurations for application layer messages (except for neighbor beacons which are transmitted as broadcasts in every scenario): (*a*) packets transmitted employing broadcast, which means a node performs no retries and immediately goes into post transmit backoff whenever the transmission of a frame concluded (called *w/o ACKs*); (*b*) packets transmitted employing reliable unicast as defined by the IEEE 802.11 HCF (called *w/ ACKs*); and (*c*) packets transmitted as broadcast frames, but using our application-based retransmission approach outlined in

Section 3.3.5 (called *app. retry*). We note that in all configurations nodes are able to overhear information. This means that a node can overhear unicast packets not designated to it, similar to running an additional interface configured in monitoring mode. Information which is overheard is handed up to the application layer as well.

The premier metric for measuring the performance of the Geocasting application is the fraction of informed nodes for a specific information item. Also the delay up until a specific information is received by the nodes plays an important role. To evaluate lower layer metrics we focus also on the number of queued frames in the EDCA subsystem of the MAC, specifically for Geocasting traffic. We also report on the queuing time of these frames. As a final metric we also investigate the performance of the wireless channel by measuring the channel load.

In Figure 3.10a, we present the fraction of vehicles which have received a specific information item. In particular, we show this metric for three different configurations: very low, low, and high road traffic densities and use a generation frequency of 4 Hz



(a) fraction of informed vehicles

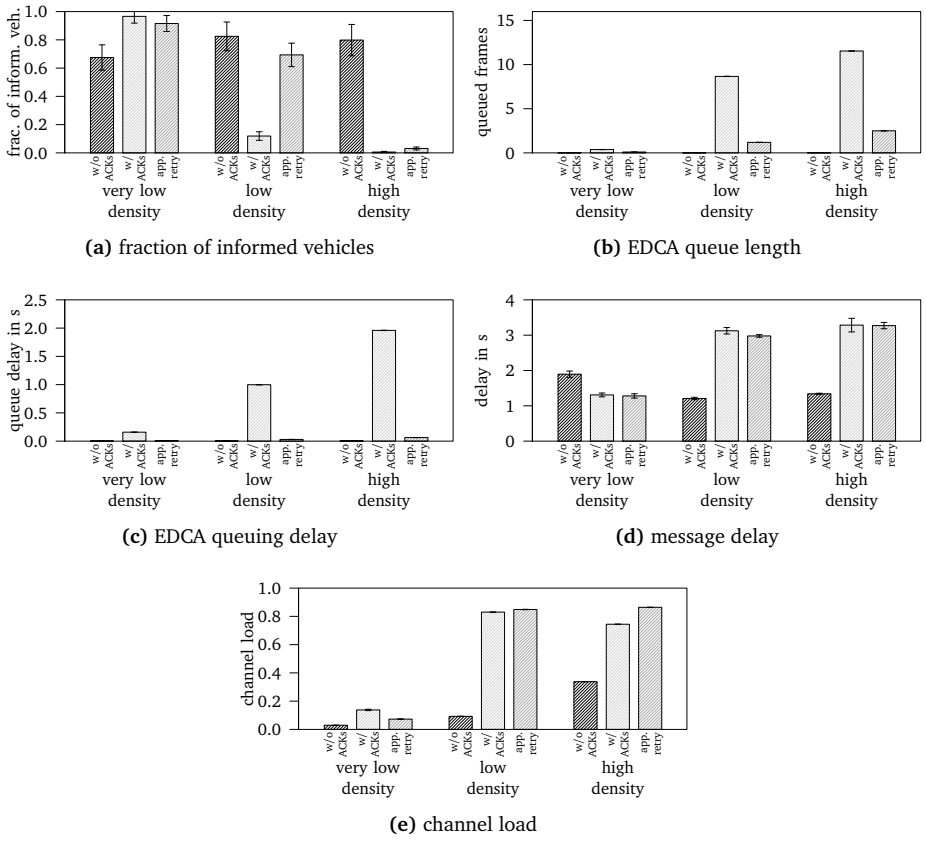
(b) EDCA queue length

(c) EDCA queuing delay

(d) message delay

(e) channel load

Figure 3.10 – Performance of the Geocasting application for three different traffic densities and a message generation interval of 4 Hz; based on [28]



(a) fraction of informed vehicles

(b) EDCA queue length

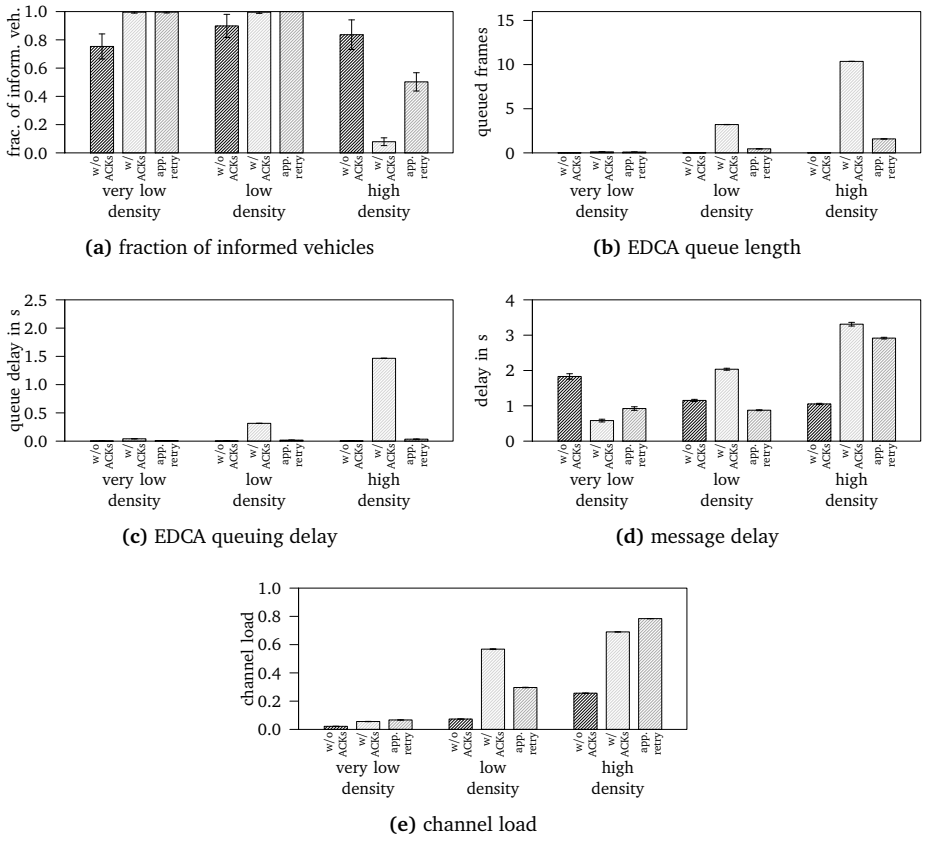
(c) EDCA queuing delay

(d) message delay

(e) channel load

Figure 3.11 – Performance of the Geocasting application for three different traffic densities and a message generation interval of 10 Hz; based on [28] © 2018 IEEE.

for the information items. Intuitively (by not knowing about the issues induced by head of line blocking), one might expect a higher rate of informed nodes if a communication mode employing ACKs is used. Here a better performance would be expected when using the reliable unicast communication mode provided by IEEE 802.11. However, as the results indicate, this intuitive expectation cannot be confirmed. In particular the exact opposite for the low density and high density scenario takes place: The fraction of informed vehicles gets lowered to approximately half when using the low density scenario, and to one twentieth for the high density scenario. In case of using our very simple *app. retry* approach (as described in Section 3.3.5) the success is dependent on the node density: For low node densities the approach performs better than pure broadcast based communication not using retransmissions. When node densities are high, however, not using retransmissions leads to vastly better application performance. However, in case of a very low traffic density scenario, retransmissions of both, our simple *app. retry* approach and the



(a) fraction of informed vehicles

(b) EDCA queue length

(c) EDCA queuing delay

(d) message delay

(e) channel load

Figure 3.12 – Performance of the Geocasting application for three different traffic densities and a message generation interval of 2 Hz; based on [28] © 2018 IEEE.

reliable unicast communication mode of WLAN will increase the fraction of informed vehicles, in comparison to a pure broadcast based communication principle. In general, the results are comparable when using an information item generation rate of 10 Hz as shown in Figure 3.11a and 2 Hz as shown in Figure 3.12a.

The reasons for these results can be observed when we have a closer look at the queues: In Figure 3.10b we show for an information item generation rate of 4 Hz that head of line blocking is the main reason for the low application performance if the reliable unicast communication mode of WLAN is used. Due to the high topology dynamics, the intended recipient of a message is sometimes no longer there to reply with an acknowledgment (see Section 3.5.1). Thus, frames waiting for ACKs both block and fill the queues of the EDCA subsystem. Hence, increasing both the information item generation rate as well as the traffic density, the number of frames queued also increase like shown in Figure 3.11b (for a rate of 10 Hz) and Figure 3.12b (for a rate of 2 Hz).

Next, in Figures 3.10c, 3.11c and 3.12c we investigate how this impacts message age for different traffic densities and information generation rates. Similar to what we discussed in Section 3.3, the impact of blocked queues when using unicast communication employing acknowledgments is cumulative, which yields messages for durations on the order of seconds up until they are finally transmitted. However, our simple *app. retry* is not negatively impacted from head of line blocking, as it works around this problem.

Results shown in Figures 3.10d, 3.11d and 3.12d reveal that retries might be able to spread a message further through the network. However, this comes to the cost of increased average delay at which network participants receive new information. Only for the very low density scenario the delay of broadcast based communication is slightly higher which is mainly caused by the behavior of the application layer protocol behavior: In case new neighbors are detected by the neighbor table algorithm, as outlined in Section 3.5, they will be informed about stored information items in a vehicles' knowledge base.

Moreover, in Figures 3.10e, 3.11e and 3.12e we see one of the reasons why the simple *app. retry* approach is not performing well in high density networks. Here, all presented forms of retransmissions contribute to an increased channel load. In particular for the high density scenarios, this increases to more than the network can handle. In summary, although retransmissions give a certain value to improve the reliability of wireless communication even in the case when network topology dynamics is high, blindly using retransmissions multiple times leads to only few potential gains. However, this comes at the cost of a massively increased channel load, which will most probably have a negative impact on other applications accessing the wireless channel at the same time.

3.6 Lessons Learned

In this chapter we studied the WLAN mechanism for reliable unicast communication (that is, using acknowledgments) in networks with high topology dynamics. We use VANETs as a prime example, and showed that this mechanism frequently causes head of line blocking because of missing ACK frames, either because of a node is gone, or due to interference. In particular we investigated this effect employing measurements on hardware, analytical calculations, as well as (large scale) computer simulations.

Results show that the impact of head of line blocking can be disastrous for many applications which require low latency communication. A not received acknowledgment frame can stall the EDCA queue and thus transmission of other frames for an average of 40 ms. Moreover, one single unicast application is able to block twice as

many broadcast applications on the same node for more than 200 ms. Our results reveal that even for moderate topology dynamics the head of line blocking effect can negatively impact the higher layer protocol performance of a system. This leads to an increase of channel load up to the point when the wireless channel is completely congested, having the consequence of massive message loss and increased delays in the order of seconds. All of these effects are based on the interplay of network topology dynamics and the reliable unicast communication mechanism. Additionally, we demonstrated that head of line blocking is easy to trigger, both by using active or passive attacks in combination with commodity WLAN hardware.

To provide a way of mitigating the problem of head of line blocking we also presented a simple approach that moved the task of retrying failed transmissions from the MAC layer to higher layers and thus avoids using the exponential backoff procedure, as well as letting other frames be transmitted in between the retransmissions of the failed unicast frame. Our investigations show that even using this simple approach reduces delays by an order of magnitude, although the retransmissions still cause non-negligible channel load. We thus conclude that there is a strong need for adapted retry mechanisms for wireless networks of high topology dynamics in case of reliable unicast communication.

Chapter 4

Class Based Broadcast Architecture

In the previous chapter we identified unicast communication to be harmful in vehicular communications since the exponential backoff based retransmission scheme of IEEE 802.11 can easily lead to head of line blocking scenarios where networking performance gets significantly degraded. Based on these findings, and the motivation given in Chapter 1, in this chapter we design a holistic networking architecture purely based on broadcast communication which supports a multitude of different application domains for Vehicular Ad Hoc Networks (VANETs) by categorizing communication paradigms within four different classes. This class-based network layer capable to support past and future application domains for Intelligent Transportation Systems (ITS), decouples applications from their message dissemination logic and allows multiple applications to be operated simultaneously.

The content of the following chapter is based on the following peer-reviewed publications:

- F. Dressler, F. Klingler, C. Sommer, and R. Cohen, "Not All VANET Broadcasts Are the Same: Context-Aware Class Based Broadcast," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 17–30, Feb. 2018, © 2018 IEEE.
  My contribution in this work was the design of candidate protocols and the implementation of the class based networking architecture as well as the analysis and evaluation of different networking protocols.

- F. Klingler, "Context-Aware and Class-Based Broadcasting in VANETs," in *International Conference on Networked Systems (NetSys 2015), PhD Forum*, Cottbus, Germany, Mar. 2015.
  My contribution in this Regional Workshop paper was the outline of important metrics to investigate a holistic networking concept for IVC and a general outline of further topics to investigate in my PhD thesis.

4.1 Motivation

Broadcast communication[4] can be beneficial in vehicular networks for two main reasons:

As a first reason, almost all applications in the VANET domain need to share the same piece of information which is relevant for many vehicles in some area. As a second reason, even multi hop dissemination often relies on multiple forwarders [128], which make broadcast as the natural scheme for communication between vehicles. Solutions based on IP and IPv6 are currently investigated by the IETF [129], however they are mainly focusing on unicast communication in the area of non-safety

[4]Here we use the term broadcast also for geocast and multicast protocols since their main communication principle is to broadcast information to the wireless medium, and only the receiving nodes decide whether the information needs to be further processed.

applications. As we showed in Chapter 3, unicast communication even causes more problems in VANETs due to the head-of-line blocking problem of IEEE 802.11 in highly mobile environments [25], [28]. Currently, approaches for broadcast protocols follow the "one-fits-all" principle: they propose a single, mostly beaconing-based protocol to support all envisioned VANET applications. However, by studying the properties of applications in the VANET domain, we soon see that they can only be optimally supported when using a specialized Network layer that employs several different broadcast protocols. In recent years many proposals for VANET broadcast protocols were developed, where each of them is designed for a specific application in mind: platooning, intersection safety, cooperative awareness, traffic information, etc. [19]–[21], [60], [130]. The main disadvantage of all these protocols is that they were not indented to cooperate or even co-exist on the same Network layer. This further brings the disadvantage that, although a protocol itself can perform reasonably good, running multiple protocols at the same time on a single wireless network will lead to severe performance problems.

In this chapter we carefully investigate the differences and commonalities of VANET broadcast protocols and identify that *not all VANET broadcasts are the same*. Consequently, we propose a set of four distinguished classes of broadcast protocols that we believe would suit all VANET applications, ranging from ultra-low latency safety to generic range-oriented Geocasting solutions. One fundamental requirement of protocols in each class is that they must be context-aware, namely, their basic properties depend on the application requirements. Thus, each protocol differs greatly in the number of nodes (vehicles) which are selected to initiate a broadcast, the coverage and reliability employing retransmission strategies, as well as the priorities of messages created by each protocol, and so on.

We propose a novel, integrated, context-aware, broadcast-based Network layer for supporting past and future VANET applications. To achieve this goal we design four different broadcast classes (called *class A* to *class D* in this chapter) that match the requirements of all known applications. These classes co-exist on the same Network layer and wireless channel, and also make use of cross-protocol functionality. For example, a protocol from one class relies on the information provided by the protocol from another class, or an event is first broadcast by a protocol from one class to reach all neighbors within n-hops and then by a protocol from another class to further broadcast the message towards a specific geographic region. Our investigations clearly show a strong dependency between the various broadcast classes. The proposed broadcast classes represent the underlying basis for designing new applications and more holistic transport protocols by combining two or more classes. We believe that our key findings for each class of protocols build the basis for future protocol design.

The key contributions in this chapter can be summarized as follows:

- We develop a novel context-aware class-based broadcasting framework for the Network layer of VANETs, which consists of four different classes of broadcast protocols (Section 4.3).
- We design candidate protocols for all classes, or select suitable protocols from the relevant literature (Section 4.4).
- We provide a detailed analysis of the performance of the various proposed protocols (Section 4.5) as well as how they influence each other when being executed concurrently.

4.2 Preliminaries

In the history of VANETs, a variety of generalized protocol stacks have been proposed, supporting different vehicular networking applications [9]. As application requirements are not always the same, specific broadcast protocols have been investigated for use with a single, very specialized application. These protocols almost always require a dedicated radio channel for operation, i.e., one that supports only one application, and does not take into consideration to share resources among different applications. As already outlined, most of these approaches follow a one-fits-all concept, which is very limited in its suitability for all possible IVC applications. Therefore, substantial research towards application-specific broadcast protocols has been conducted in the past.

In particular, application-specific protocols have been investigated in a whole spectrum of potential applications, of which we choose several examples to illustrate the requirements of the selected approaches. At first, we start with cooperative

Class	message priority	expected number of initiators	scope of the broadcast
A	normal	only 1	1-hop transmission circle
B	very high	very few (1-4)	N-hop transmission distance
C	high	few or more (1-20)	a certain geographic area
D	low	few to many (1-100)	a certain geographic area

Table 4.1 – The Proposed Broadcast Classes; Characteristics of Initiators and Scope of Broadcast; based on [31] © 2018 IEEE.

Class	expiration time	broadcast type	report merging
A	less than a second	unreliable	no
B	less than a second	semi-reliable	no
C	several seconds	semi-reliable	yes
D	several minutes	reliable	yes

Table 4.2 – The Proposed Broadcast Classes; Characteristics of Latency, Broadcast type and Report merging; based on [31] © 2018 IEEE.

awareness. This application relies on multi-hop broadcasting in order to overcome the substantial radio signal shadowing that occurs in urban environments. A reliable broadcast protocol that focuses on this problem is published in [131]. Further the use of parked vehicles has been proposed in [132], [133] to increase reliability of message dissemination.

Another application is Cooperative Adaptive Cruise Control (CACC), or platooning of vehicles. This application has possibly the tightest real-time requirements for maintaining cruise control in order to support very low inter vehicle distances. Broadcast based communication protocols have been designed for this application which require a dedicated radio channel for operation [19]–[21], [134].

Less time-critical information exchange is used in road traffic information systems, information downloading, and vehicular cloud applications. Broadcast protocols in that area have been defined for all these application classes, where the primary focus is to provide a certain degree of reliability [60], [130].

All the mentioned solutions cannot easily be combined with other protocols – there is a strong need for dedicated channels for each proposed protocol. In particular, it is crucial to avoid transmission of redundant information which can overload the channel, as well as to provide interaction between these applications to achieve a better message dissemination performance. The need for dedicated channels obviously limits the applicability of the developed protocols depending on the geographic region, since a limited number of available service channels have been dedicated for use in the vehicular networking context [35] (see Chapter 6).

Our aim is to combine both approaches, a generalized and a application-specific protocol design approach and identify the need for different classes of broadcast protocols. Thus, we developed an integrated Network layer for IVC, based on four distinct broadcast classes in a context-aware approach.

4.3 Class Based Broadcasts

We begin our discussion of the Class Based Broadcast concept with a list of classification criteria which are important for VANET broadcast protocols. This list serves as the basis for our proposed broadcast classes. As a next step we present details about the four broadcast classes, and finally show the system design of our holistic Network layer.

4.3.1 Classification Criteria

In the following we list three main classification criteria which we identified. Other, less important, criteria are outlined in Tables 4.1 and 4.2.

The first classification criterion is the *priority* of the event that triggers the broadcast. Here we distinguish between routine events, such as broadcasting a beacon message periodically in order to detect all 1-hop neighbors for cooperative awareness, and extraordinary events, such as announcing an important event like a traffic accident or a slippery road. Moreover, there are events which are not periodic, but also not very extraordinary, such as detecting a free parking lot or the information of a point of interest. This criterion is important because it affects the priority of the information which needs to be transmitted.

As a second classification criterion we use the *expected number of vehicles* (nodes) that are likely to detect an event. As an example, whenever a vehicle experiences a mechanical problem, it is the only node to be aware of this event. Congestion on the highway, however, is likely to be detected by all the vehicles driving in the reverse direction, and all vehicles in the main direction are affected by that congestion. When many nodes detect and report the same event, the channel might become heavily loaded, and many frame collisions on the wireless channel are likely to take place. This case must be handled by the broadcast protocol, and countermeasures need to be taken.

As the third classification criterion the *scope* or target of the broadcast message is used. The scope of the broadcast is crucial for deciding how messages will be disseminated from one vehicle to another and how this dissemination will be stopped. One option for the broadcast protocol is to target all the nodes within a given geographic radius (e.g., 750 m) around the broadcast originator. A use case for this is when a vehicle invokes the broadcast to report the detection of a free parking lot in the center of town, or to report a traffic jam in a relatively distant area (e.g., 5 km from the location of the reporting/detecting vehicle).

4.3.2 Broadcast Classes

The above criteria are used to define four VANET broadcast classes as depicted in Figure 4.1. We outline the main distinctions between these classes in Table 4.1. Further, in Section 4.4, we define specific networking protocols for each of the classes we propose.

4.3.2.1 Class A

This class consists of beaconing protocols, which broadcast periodic CAM or HELLO messages to 1-hop neighbors to achieve cooperative awareness among vehicles. As per definition, in this class there is only one initiator for each broadcast. For instance, only node v initiates the broadcast of HELLO(v) messages. Current standardization efforts in the scope of IEEE 1609, as well as ETSI ITS-G5, focus on this class of protocols for their applications. The information received by protocols of this class

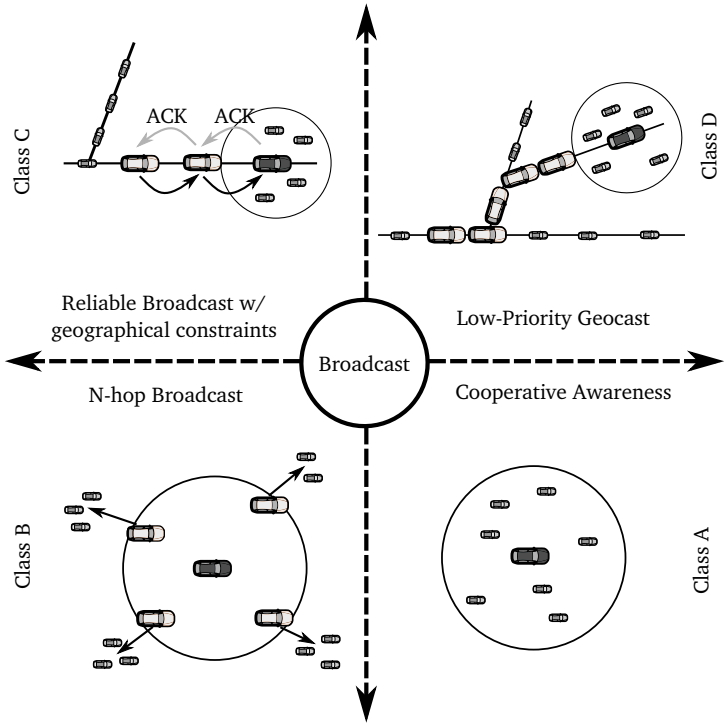


Figure 4.1 – Our vision of VANET broadcast classes: Class A for medium priority Cooperative Awareness Messages (CAMs); Class B to forward highest priority events within *n* hops; Class C for high priority reliable broadcasting towards geographical regions; and Class D for low priority Geocasting; based on [31] © 2018 IEEE.

allows each node to maintain an up-to-date list of its neighbors. Initially, this type of protocol has been considered for cooperative awareness applications only, where no information is aggregated among different beacons. However, in this chapter we show that, in fact, it also allows maintaining information about 2-hop neighbors (neighbors of neighbors), which can be beneficial for the operation of the other broadcast classes. This way Class A protocols serve as basis for all other classes of protocols by providing this 2-hop neighbor information.

4.3.2.2 Class B

Protocols in this class broadcast information about an emergency event likely to be detected only by a very low amount of vehicles. Examples for information transmitted in this class are notifications about an animal on the road, a broken-down vehicle, or a sudden stop due to a traffic accident. A requirement of broadcast protocols in this class is that they should cover all the vehicles that surround the detecting vehicle and are not too far from it, i.e., following vehicles on a freeway or vehicles

approaching the same intersection. The Decentralized Environmental Notification Message (DENM) concept of the ETSI ITS-G5 suite is very close to this class [81], but there is one important difference as follows. DENM incorporates Geocasting capabilities, which delay the propagation of the messages due to the need of up to date positioning information and the need to perform time-consuming forwarding decisions at the Application layer whenever a message of this type is received. Since Class B messages are of very high priority, we believe that a better forwarding strategy than relying on Global Positioning System (GPS) coordinates is needed.

Our alternative proposal is that the vehicles, which need to be informed, be within N-hop radio transmission distance from the originator. The value of N is typically 1 or 2, depending on the event, and is determined by the application that detects, generates and announces the event. There is no need for protocols in this class to merge reports that are originated by different nodes for the same event. This is because first, the number of nodes that detect a Class B event is small, and second because merging different reports requires that the content of different messages be stored and compared in the Application layer, which substantially delays the speed of broadcast. Multiple messages about the same event are identified by their Network layer headers and are pruned from the network in order to reduce the channel utilization and achieve better communication performance. The most important and unique feature of protocols within Class B is that they do not require geographical positions; the dissemination range is within N hops.

4.3.2.3 Class C

In this class we propose protocols that broadcast information about ongoing *important, but-not-very-urgent* events that are likely to be detected by many nodes. Events in Class C are also relevant to nodes that are much farther away from the detecting node, which usually is not the case for an event detected by Class B. The exact geographic area depends on the type of the event, thus the reporting node needs to determine the geographic constraints of the area to which the reported event should be propagated.

A fundamental requirement for protocols in this class is that they must prevent a broadcast storm [64]. Usually this is performed in two ways: In the beginning, a node that identifies an event initiates a new broadcast with probability $p \leq 1$. Next, different instances of the messages which are initiated by different nodes, could be merged (or even fused with additional information) if they report the same event. This operation requires the nodes to process Class C messages in the Application layer. This is also necessary to be able to determine that two events announced by different nodes are actually the same. A typical approach here is Geocasting as the information to be disseminated will likely be of interest in a certain geographical area

only. In contrast to Class B protocols we take advantage of geographical positions when using message forwarding strategies.

4.3.2.4 Class D

Protocols of this class broadcast information about low priority and non-urgent events whose expiration time is much longer than those of Class C events. Usually the detection rate of a Class D event, i.e., the number of detecting vehicles per second, is smaller than in Class C. However, due to the much longer expiration time of a Class D event, it can happen that such an event still be detected by a large number of nodes. Therefore, it is important to identify and merge announcements which are broadcast by different nodes for the same event. Because Class D events are of lower priority than events in all other classes, our proposed Class D protocols are based on distributed caching and their bandwidth consumption is adapted to the available resources on the wireless channel. Further, the dissemination scope is similar to Class C protocols, i.e., the target will be a geographical area too. The most important unique feature of Class D protocols is the ability to merge different reports according to their content, relevance and scope.

4.3.3 Mapping Broadcast Classes for Building Applications

To get a better overview about use cases employing IVC technology, we list in Table 4.3 possible VANET applications for *Safety, Traffic Efficiency, and Infotainment Applications* as well as Network Coordination [8], [9] and outline how they can be mapped to fit our class-based forwarding model. We want to highlight that the proposed assignment of classes is not engraved in stone; it is more a demonstration of the classification idea. The main idea is that one or multiple broadcast classes together build the basis for the application needs, e.g., intersection collision warning. Certainly, this can be further supported by assuming a multi-technology approach when different communication technologies are integrated, e.g., LTE, WiFi, Bluetooth, and others. This concept is also known as *heterogeneous vehicular networking* [135].

Another fundamental aspect is to highlight the dependencies between the classes. This can be demonstrated by the following example: Suppose that a vehicle makes an emergency stop due to some critical event like an accident. The node informs all its 1-hop and 2-hop wireless neighbors of this sudden stop by using a Class B protocol. As we show later in this chapter, the information about these wireless neighbors is available and maintained from the routine execution of Class A protocols. All the nodes within this 2-hop range that are informed of the emergency stop invoke a Class C protocol and inform further nodes towards a specific geographic region, in a distance of 1 km about a traffic jam. The nodes that get informed about the

Application	Class A	B	C	D
Safety Applications				
Cooperative awareness	✓			
Intersection collision warning	✓	✓		
Overtaking vehicle warning		✓	✓	
Lane change assistance (blind spot)		✓		
Rear end collision warning		✓	✓	
Head on collision warning (frontal)		✓		
Emergency vehicle warning		✓	✓	
Cooperative forward collision warning	✓	✓		
Co-operative merging assistance		✓		
Pre-crash sensing / warning	✓	✓		
Emergency electronic brake lights		✓	✓	
Traffic condition warning		✓	✓	✓
Wrong way driver warning	✓	✓	✓	✓
Stationary vehicle warning	✓	✓	✓	✓
Signal violation warning		✓	✓	
Hazardous location notification		✓	✓	✓
Collision risk warning		✓	✓	
Control loss warning	✓	✓		
Traffic Efficiency and Management Applications				
Platooning	✓	✓		
Green light speed advisory			✓	✓
Cooperative navigation / TIS				✓
Public transport lane		✓	✓	
Parking space information			✓	✓
Infotainment Applications				
POI information				✓
Media downloading				✓
Local advertisements				✓
Multi-player games				✓
Network Coordination				
Neighborhood management	✓			
Multi-channel multi-radio coordination	✓	✓		

Table 4.3 – Mapping of VANET Applications to our Class-based Broadcast Architecture; based on [31] © 2018 IEEE.

traffic jam create a Class D event, and disseminate this information to further distant nodes.

In Figure 4.2 we show a simplified design overview of our class-based broadcasting approach. As we see, four distinct sets of protocols are exchanging information over the wireless channel, where each set of protocols uses a different priority. The priorities defined for each class (see Table 4.1) are mapped to MAC layer Access Categories (ACs), as defined in the IEEE 802.11p protocol and employing the Enhanced Distributed Channel Access (EDCA) prioritization scheme defined in IEEE 802.11e [22], [136]. This guarantees prioritization and provides a sufficient level of fairness to prevent starvation of flows. Our architecture helps to provide a completely self-organizing and distributed interaction among all protocol classes. Messages are

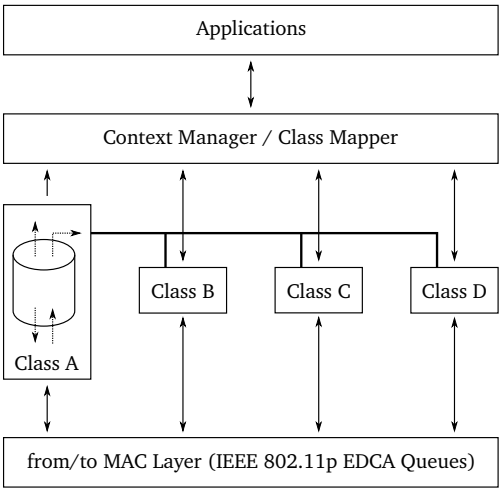


Figure 4.2 – Overview of the System design. A context-aware class mapper manages data exchange of applications with four distinct sets of protocols, each connected to the MAC layer using different priorities and each fulfilling a specific role; based on [31] © 2018 IEEE.

prioritized, in particular Class B over Class C and Class D, as well as Class C over Class D.

New data is generated only locally (for Class A) or received from the Application layer (Class B, Class C, and Class D). Passing the right information to the right protocol is handled by a context-aware class mapper, which could also be accomplished by the envisioned applications running on top of our proposed Network layer. The Class A protocol maintains list of neighbors and provides access to its contents to other protocols to base their forwarding decisions on. This way, each protocol can operate autonomously from the Application layer, e.g., for relaying messages. Finally, the context manager is responsible to classify data received by the Application layer and can transform messages among classes depending on criteria outlined in Section 4.3.1, e.g., Class B to Class C or Class D.

We prioritize messages (cf. Table 4.1) by assigning them to the corresponding EDCA access categories.

The major selling point of our integrated protocol architecture is that we separate Network layer functionality and application logic. This has the advantage that *(a)* message forwarding decisions are transparent to the applications and *(b)* redundant information by applications concurrently accessing the wireless channel can be avoided. Consequently, this allows a better utilization of the shared wireless medium among the nodes to achieve a better overall performance.

4.4 Detailed Design of the Various Protocols

Is a next step, we propose a detailed design of specific protocols for each of the four broadcast classes, which is based on the principles introduced in Section 4.3. Our purpose is to demonstrate how the various requirements of each class can be addressed by specific protocols. Moreover, we want to highlight how those protocols differ with respect to the metrics their forwarding decisions are based on. Our proposals of protocols build upon existing approaches in standardization and in the scientific literature, but also introduce completely new approaches or substantial changes to be suitable for our integrated class-based broadcasting architecture.

4.4.1 Class A Protocols

As a beginning we focus on protocols that broadcast routine periodic (beacon) messages to nearby vehicles; primarily to inform every node about the presence of a car as it is needed by cooperative awareness. While existing protocols have been designed to allow each node to inform its 1-hop neighbors of its existence, we believe that a more sophisticated protocol is necessary, mainly because the information disseminated by this protocol is needed for the other classes. In particular, the Class B, Class C, and Class D protocols we propose later on need to know the identities of the 2-hop neighbors of each node. This way, we propose a scheme to maintain 2-hop neighbor information in our Class A protocols. A first and naïve approach is that each node v includes the identities of its 1-hop neighbors in its beacons as a list. This will greatly increase the length of these messages in urban environments, and thus increase their MAC collision probability since channel utilization quickly gets very high.

To avoid this problem, we use a Bloom filter [73] (see Section 2.3 for more details) in the following way: For each neighbor w (from whom we received a beacon) of node v, node v adds the node ID (or MAC address) of w to a local data structure representing a neighbor table. This neighbor information is stored in a set $\mathbb{T}_v$, which thus represents all 1-hop neighbors of node v. Moreover, each node v maintains bit vector $\mathcal{T}_v$ representing a Bloom filter storing all entries of $\mathbb{T}_v$, i.e., $\mathcal{T}_v \leftarrow \mathbb{T}_v$. This Bloom filter is added to each transmitted beacon messages of node v. This way, whenever a beacon containing a Bloom filter $\mathcal{T}_v$ is received, any node can check with high probability whether a local neighbor is also a neighbor of v. As already outlined, Class A maintains a neighbor table where the information of all the 1-hop neighbors is stored, including their position, and their Bloom filter. A periodic timer expires old entries in the neighbor table $\mathbb{T}_v$ whenever two consecutive beacons have not been received, which can occur either due to an overloaded channel or due to a neighbor moved outside the communication range. When node u receives

the beacon from v, it also updates its information about its 2-hop neighbors that can be accessed via v. All 2-hop neighbors are collected in form of a Bloom filter as $\mathcal{T}'' = \bigcup_{v \in \mathbb{T}} \mathcal{T}_v$. For this, the union of two Bloom filters can easily be calculated by issuing a logical OR operation on each bit field of the two Bloom filters [97] (see Section 2.3 for more details).

In Chapter 5 we will further investigate protocols in this broadcast class and elaborate a sophisticated method to determine the time a neighbor needs to be purged from the neighbor table without being dependent on a predefined timer value. Further we study in Chapter 5 the impact of various Bloom filter sizes to the neighbor table performance.

In essence, our Class A protocol maintains an exact list of 1-hop neighbors with their respective geographical position, as well as a probabilistic list of 2-hop neighbors in form of a Bloom filter. We will discuss the use of this information within the Class B, Class C, and Class D protocols later in this chapter.

4.4.2 Class B Protocols

Protocols in this class broadcast information about an emergency event that is likely to be detected only by a few vehicles. The information needs to be broadcast to all N-hop neighbors of the originating node, where N is determined by the application and is typically 1 or 2. The value of N can be added to a TTL-like field in the Network layer header of the protocol to allow quick decisions: whether to forward the message, or (if the value of this field is 0) to drop it.

In order to mitigate the broadcast storm problem [64] where receiving nodes decide whether to forward a message or not, we follow a sender-based rebroadcast decision, where the origin of the broadcast selects the rebroadcast nodes according to the information provided by the neighbor table maintained by Class A. The origin node includes the IDs of the selected rebroadcast nodes in the message together with an individual offset for each selected node to avoid collisions due to simultaneous rebroadcasts. For $N = 1$, the operation of the protocol is trivial and very similar to the Class A protocol because relay nodes are not used. We now investigate the case where $N = 2$, and highlight that the proposed protocol can be extended for any $N > 2$. Going beyond $N = 2$ requires a change of the protocol behavior not only to add additional nodes in the Bloom filter (or using multiple Bloom filters), which is trivial, but particularly a mechanism to de-synchronize the beacon messages of nodes in a multi-hop vicinity to avoid synchronized collisions. In essence, this leads to a new beacon protocol (Class A), which is beyond the scope of this thesis.

In order to reduce the load on the wireless channel, often a greedy approach is used to gain most progress in distance, where the rebroadcast node is that node being most distant from the originator. This has the disadvantage to be limited to the

underlying road topology, which means the protocol behavior needs to be adapted for every new scenario. However to cover non-regular 2D environments (e.g., urban or inner cities), we follow a new approach by choosing as many nodes as needed to cover all our 2-hop neighbors. We take advantage of the idea proposed in [137], where every node v that receives or generates an event to be broadcast nominates one or multiple of its 1-hop neighbors to rebroadcast the packet. To achieve this goal, node v uses the neighbor information and the Bloom filter sets maintained by the Class A beacon protocol and desires to choose a set of re-broadcasters $\mathbb{R}$ as the minimum subset of 1-hop neighbors in $\mathbb{T}_v$ to cover all of its 2-hop neighbors.

This combinatorial optimization problem is called minimum set cover problem, which is known to be *NP*-hard [138]. Therefore, to speed up computation, we use a greedy iterative process where node v chooses the set $\mathbb{R}$ by selecting a node u that has most new (still uncovered) neighbors and has not yet been selected as a rebroadcast node. Technically, node v starts with an empty set of re-broadcasters $\mathbb{R}$ and a Bloom filter of already-covered 2-hop neighbors $\widehat{\mathbb{T}}''$, which is initialized to the 1-hop neighbors, i.e., $\widehat{\mathbb{T}}'' \leftarrow \mathbb{T}$. It repeatedly chooses the best 1-hop neighbor u defined as

$$u = \arg\max_{u \in \mathbb{T}} \left(\text{diff}(\widehat{\mathbb{T}}'', \mathbb{T}_u) \right) \tag{4.1}$$

and adds this node u to $\mathbb{R}$ and its Bloom filter $\mathbb{T}_u$ to the already covered nodes represented by $\widehat{\mathbb{T}}''$.

The quality of a specific Bloom filter to estimate $\text{diff}(\mathcal{A}, \mathcal{B})$, the number of entries in a Bloom filter $\mathcal{B} \leftarrow \mathbb{B}$ that are not part of a local filter $\mathcal{A}$, can formally be expressed as $\text{diff}(\mathcal{A}, \mathcal{B}) = |\mathcal{A} \cup \mathcal{B}| - |\mathcal{A}|$ by taking advantage of the cardinality of a Bloom filter. For the cardinality estimation [99] of Bloom filters please refer to Section 2.3.

The process of rebroadcast node selection ends when all 2-hop neighbors are covered, as can be derived by comparing $\widehat{\mathbb{T}}''$ and $\mathbb{T}''$ to be equal. The set $\mathbb{R}$ now contains all 1-hop neighbors selected to rebroadcast the message. Since $\mathbb{R}$ is usually small (e.g., it is close to 2 in freeway scenarios), it is added to the broadcast message. To avoid a broadcast storm [64], a node receiving this message chooses its rebroadcast delay based on its index i in $\mathbb{R}$ as $i \times t_{\text{rebroadcast}}$. In dense networks, e.g., in urban scenarios, node v can make the packet dissemination process more robust by adding more 1-hop neighbors to $\mathbb{R}$. Optionally, if the cardinality of $\mathbb{R}$ is too large to include all chosen 1-hop neighbors, the addresses of these nodes could also be replaced by a Bloom filter $\mathcal{R} \leftarrow \mathbb{R}$. This way, a receiving node needs to make a probabilistic test whether its own ID is included in the Bloom filter $\mathcal{R}$. To increase reception reliability duplicates of each Class B message could be sent.

4.4.3 Class C Protocols

Protocols in this class use geo-routing and reliable broadcasts to disseminate information about a certain, detected event within a specific geographic region. A message of this type consists of a destination (currently a simple geo-position) where the information should be propagated to, and a lifetime. While in our Class B protocol a decision whether to forward a message is made in the Network layer, using information that appears in the Network layer header, in our Class C protocol the decision needs to be made in the Application layer. Here it reads the information about the event and makes a forwarding decision based on the location of the detecting node, the event time, and whether a report about a similar event has already been received. We propose the framework for our Class C protocols as follows.

1. First, the node decides whether to report the detected event which is done in a probabilistic manner, based on the number of detecting nodes, the event type, and the number of 1-hop or 2-hop neighbors of v.

2. Second, the node determines the destination of the event which could be a specific geo-position or even a geographical area.

3. Third, the node forwards the event towards its destination where every node that receives an event determines whether it is already acknowledged (that is, a duplicate to be dropped) and whether to forward the event based on its 1-hop neighbors as provided by Class A. To reduce congestion on the wireless channel and keep overhead low we decided to piggyback the Acknowledgment (ACK) to the rebroadcast.

Next, we outline an example algorithm for selecting an appropriate set of forwarders. This algorithm fits the case where the originator v needs to broadcast the information to all the nodes between v and the destination geo-position on a one dimensional highway. In this case, v nominates another node u that will nominate another node w and so on. Generally, we prefer to nominate a node in the direction of the broadcast, but if this is not possible (because of not received acknowledgments or unavailability of a fitting neighbor), we perform store-carry-forward until the Class A protocol provides a new fitting neighbor. The algorithm can be extended to 2D setups by invoking a separate instance of the protocol for each direction of the broadcast. We outline the operation of our class C protocol in Algorithm 4.1.

4.4.4 Class D Protocols

Protocols in Class D disseminate information about non-urgent events, such as reporting an available parking lot or a traffic jam in a further distant area. Recall

Require: e, the event to be forwarded
1: $\mathbb{B} \leftarrow \varnothing$
2: **while** no Application-layer ACK received $\wedge$
retransmit limit not reached **do**
3: $\mathbb{F} \leftarrow \{n \in \mathbb{T} : n \text{ is towards the destination of } e\} \setminus \mathbb{B}$
4: **if** $\mathbb{F} \neq \varnothing$ **then**
5: $u \leftarrow \arg\max_{n \in \mathbb{F}} (\text{distance}(n, v))$
6: $m \leftarrow \text{createMessage}(e, u)$
7: broadcast(m) with delay $\mathcal{U}(0, t_{\text{rebroadcast}})$
8: $\mathbb{B} \leftarrow \mathbb{B} \cup \{u\}$
9: **else**
10: store-carry-forward
11: **end if**
12: **end while**

Algorithm 4.1 – Class C protocol operation for node v; based on [31]

that a traffic jam in a nearby area, which is a much more time-sensitive event, is reported by a Class C protocol. The lifetime of a Class D event is up to a few minutes, which is much longer than the lifetime of Class B and Class C events. During this time period, the event is likely to be detected by many vehicles, thus aggregation and fusion of information needs to take place in order to avoid channel congestion.

The approach we propose for Class D is based on the following concepts:
Information-centric forwarding: the Application layer processes the information, and it can be aggregated, modified, or invalidated before being disseminated to other vehicles.
Store-carry-forward: until it meets a new vehicle with which it shares this information, a moving vehicle carries the information. Spatio-temporal forwarding: the decision whether to forward a piece of information on a specific event depends on the time and place it was triggered.

Class D protocols maintain a knowledge base consisting of entries with geographic constraints and their expiration time. Based on these parameters, a broadcast decision can be taken. We decided to build this protocol upon Adaptive Traffic Beacon (ATB) [66] (see Section 2.2.1 for details), which already supports the management of knowledge bases, message prioritization, and channel quality estimation to allow channel access avoiding congestions. In the following we outline the principle of our Class D protocol:

1. A node u that detects a Class D event, or receives a message about such an event, adds or merges it into its knowledge base which represents a list of active events.

2. The rate of new neighbors ρ detected by the Class A protocol is determined by each node u. Whenever u detects a new neighbor v, u makes a probabilistic decision with $p = \frac{1}{\rho}$ whether to inform v about events in its knowledge base.

3. Whenever u decided to inform v about its relevant stored events in its knowledge base, u transmits a digest including fingerprints of all available events. Next, node v responds with an event request according to Algorithm 4.2. Here, an event is marked as missing if v is closer to its destination position than u, or v is driving towards the destination.

4. After reception of the event request by node u, it constructs and broadcasts a message containing all missed information – limited by the maximum packet size.

5. We define ϕ as the actual size of 1-hop neighbors reported by Class A. Whenever node y receives new information, it informs every node from its 1-hop neighbors with a probability $p = \frac{1}{\phi}$ about any relevant Class D events it has in its knowledge base.

6. Finally, every node z periodically checks its knowledge base and prunes obsolete events: any event whose expiration time has arrived, or any event that is not relevant anymore to the current location of the node.

Moreover, the overall concept can easily be extended to include additional metrics for p: This can range from the due date of the event (when the due date is closer, the probability is smaller); or the area to which this event is relevant (when the node moves closer to the border of this area, the probability decreases); or how busy the wireless channel is (when the channel utilization is higher, the probability is smaller).

Note that typical protocols described in the literature fulfilling similar tasks are based on unicast [81], [139]. We, however, determined that unicast may lead to substantial performance issues in this application domain [25], which we already discussed in Chapter 3.

4.5 Performance Evaluation

To demonstrate the feasibility of our class-based broadcasting approach as well as to gain more insights into the resulting performance we evaluated the system based on simulations. Our main focus is to underline the need for different broadcast protocols in accordance with the selected application requirements. To keep readability of this thesis, we only report on results for selected protocol configurations.

Require: $\mathbb{D}$, the received digest from node u
1: $\mathbb{E} \leftarrow \varnothing$
2: **for** $d \in \mathbb{D}$ **do**
3: **if** distance$(v, d_{\text{dst}}) <$ distance$(u, d_{\text{dst}}) \vee$ v is driving towards d_{dst} **then**
4: $\mathbb{E} \leftarrow \mathbb{E} \cup \{d\}$
5: **end if**
6: **end for**
7: $m \leftarrow$ createMessage$(\mathbb{E}, u)$
8: broadcast(m)

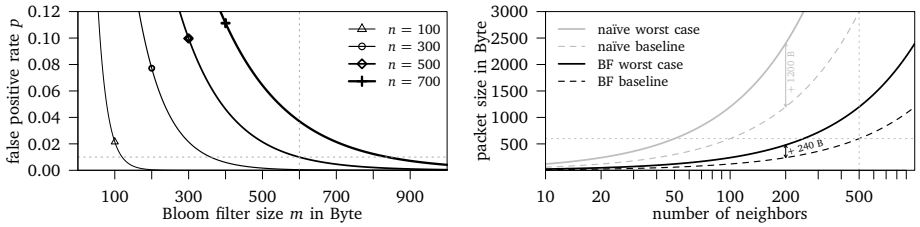
Algorithm 4.2 – Class D Event Request for node v; based on [31] © 2018 IEEE.

4.5.1 Optimal Bloom Filter Size

To derive the optimal Bloom filter size we refer for details to Section 2.3. To estimate the best parameters for our application scenario, we consider an element count derived from empirical evaluations as follows. In Figure 4.3a we plot the false positive rate as a function of the Bloom filter size m and inserted element count n. Our simulation results for the neighbor table experiments show a maximum number of 500 1-hop neighbors for each vehicle in the high density scenario. Thus, assuming a maximum false positive rate of 1 %, a Bloom filter of 600 B perfectly matches. This also nicely fits into a CAM message of up to 800 B, as used by the Class A protocols. We explicitly note that the Bloom filter size can be chosen according to the needed application demands and does not limit the amount of neighbors that can be inserted to it. The performance of the message forwarding algorithm depends on the quality of the Bloom filter information, which intuitively would increase when the Bloom filter size increases. However, since large Bloom filters will lead to severe network congestion when transmitted periodically in the network, a larger Bloom filter not necessarily leads to a better message forwarding. Further investigations on that topic are outlined in Chapter 5.

So far, the evaluation concerned the storage of IDs of cars in a Bloom filter structure assuming that this ID is fixed and does not changed over time. However, privacy preserving schemes suggest the use of so-called temporary pseudonyms for VANETs. In general the idea is to continuously change the ID of the car to a new pseudonym (or even swap it with another nearby car) in order to introduce entropy and to disallow tracking of the car's routes. A wide range of pseudonym handling schemes has been proposed in the past [140]. These ideas have also been picked up by the standardization bodies and the current ETSI ITS-G5 standard recommends to change IDs frequently [141].

In the following, we focus on the impact of such ID changes on the Bloom filter size and compare it to the naïve approach using complete IDs for the exchanged



(a) False positive rate p of a Bloom filter as a function of its length m and the inserted element count n

(b) Packet size for neighbor information as a function of the number of neighbors; for the Bloom filter (BF) approach we assume $p = 0.01$, and for the naïve approach we use 6 B per neighbor entry representing the size of a MAC address

Figure 4.3 – False positive rate of a Bloom filter and packet sizes for neighbor information for a worst case and a baseline scenario; based on [31] © 2018 IEEE.

neighbor tables as outlined in Section 4.4.1. In particular, we investigate the size of neighbor information when using a Bloom filter with a false positive rate of $p = 0.01$, and a naïve approach (neighbor entries sent within a plain list) where each neighbor entry takes 6 B of payload, e.g., the size of a MAC address. Further, we assume that in the worst case each vehicle changes its identifier (or pseudonym) for each sent beacon, which leads to an increase of the number of elements in the neighbor table by the factor of two, if entries are outdated after missing two consecutive beacons. For this, we repeated the simulations based on the Bloom filter size. As shown in Figure 4.3b, the increase of the frame length using the naïve approach is 1200 B for a neighbor count of 200 nodes. However, our Bloom filter approach takes only 240 B of additional payload for the same scenario. This leads to the conclusion that the Bloom filter is a very appropriate data structure even in case of implemented privacy preserving techniques where IDs of nodes can change very frequently.

4.5.2 Simulation Setup and Metrics

For all simulations, we used the *de facto* standard for vehicular networking simulation, Veins [122], which couples the SUMO road traffic mobility simulator with the network simulator OMNeT++ as already outlined in Section 2.4.

We configured *(a)* a 7 km freeway scenario and *(b)* a 9 km road segment of a Manhattan scenario for our simulation, respectively. For the latter, we focused on one of the major avenues (such as 5$^{\text{th}}$ Avenue in Manhattan downtown) including the simulation of all cross traffic. To mitigate potential border effects, we configured Veins to perform the network simulation only within a region centered at the middle of the respective scenario. We further configured a Region of Interest (ROI) in which we collect protocol performance metrics, cf. Table 4.4.

Road traffic for the freeway scenario was modeled in SUMO by sampling from a distribution of five different vehicle types (two types of trucks and three types of cars modeling a variety of driving styles). For the Manhattan scenario no trucks were used and four types of vehicles modeling a variety of driving styles were defined.

We configured a warm-up period of 289 s (freeway) and 59 s (Manhattan) for SUMO to fill the scenario with vehicles and an additional warm-up period of 11 s for OMNeT++ to reach a steady state of Class A protocol operations and to populate 1-hop and 2-hop neighbor tables. Moreover, in this 11 s warm-up period we pre-populate the knowledge base of vehicles with information items. Only after this time

SUMO simulation setup	
Freeway length: SUMO, OMNeT++, ROI	7, 5, 3 km
Manhattan length: SUMO, OMNeT++, ROI	9, 7, 5 km
Vehicle density (freeway): low, high	∼ 43, ∼148 veh/km
Vehicle density (Manhattan): low, high	∼ 56, ∼207 veh/km
Number of lanes	6 (3 per direction)
Percentage of cars and trucks (freeway)	90 %, 10 %
ETSI ITS-G5 TRC	
Min/default/max interval $I_{\min}$, I_{def}, $I_{\max}$	40 ms, 500 ms, 1 s
Channel busy fraction thresholds $b_{\min}$, $b_{\max}$	0.15, 0.40
ATB	
Beacon interval range $I_{\min}$, $I_{\max}$	100 ms, 1 s
Channel/interval weighting w_{C}, w_{I}	2, 0.75
IEEE 802.11p MAC	
Packet size Class A	800 B
Class A Bloom filter size	600 B
Packet size Class B and C	300 B
Packet size Class D digest	each 8 B, max. 1024 B
Packet size Class D KB entry	each 64 B, max. 1024 B
MAC priority Class A	AC_BE
MAC priority Class B	AC_VO
MAC priority Class C	AC_VI
MAC priority Class D	AC_BK
IEEE 802.11p PHY	
NIC TX power	20 mW
NIC sensitivity	−89 dBm
Frequency	5.89 GHz
Path loss model	freespace ($\alpha = 2.0$)
NIC bitrate	6 Mbit/s

Table 4.4 – Simulation Parameters; based on [31] © 2018 IEEE.

we invoke Class B, C, and D protocols and the recording of results. We summarize all relevant simulation parameters in Tables 4.4 and 4.5.

In order to investigate our class-based broadcasting architecture, we study the performance in multiple dimensions. Classical metrics for wireless networking in the vehicular context, such as the packet success rate and the channel utilization, provide only little insight into the behavior of the respective vehicular networking applications [9]. Therefore, we primarily looked at Application layer metrics to gain insights about the performance of each protocol within our class-based broadcast architecture.

We further want to comment on explicit comparison to the state of the art. For Class A, we use protocols as presented in the literature and extend these to take advantage of our Bloom filter based approach. Consequently, we use existing concepts for the Geo-networking solutions. The main emphasis of our approach presented in this chapter is to make all these protocols being able to co-exist. In the experiments, we therefore concentrate on this objective following a stepwise approach starting with Class A, then integrate Class B, and so on.

First, we investigate the beacon interval of Class A to gather insights on the latency of new status messages. Besides providing vehicle status updates via CAM messages, our Class A protocols also maintain the 1-hop and 2-hop neighbor tables. Another important aspect is the up-to-dateness of 1-hop and 2-hop entries, which is an indicator about the quality of the neighbor tables. We therefore compare each Class A beacon protocol against an oracle where this oracle derives the neighbor set according to a unit-disk model. For the distance of nodes to be treated as 1-hop neighbors we use the 99 % quantile of 1-hop distances of sample experiments to derive the communication range of our Class A protocols. This allows us to derive two sets of neighbors: those neighbors estimated by the Class A protocol $\mathbb{T}$ and those neighbors calculated by the oracle $\mathbb{O}$. Based on $\mathbb{T}$ and $\mathbb{O}$, we use the following two metrics to evaluate the quality of neighbor tables: The ratio of missing neighbors of a node compared to the oracle is derived as $\frac{|\mathbb{O}\setminus\mathbb{T}|}{|\mathbb{O}|}$. The ratio of outdated neighbors gives the relative amount of superfluous neighbors compared to the oracle as $\frac{|\mathbb{T}\setminus\mathbb{O}|}{|\mathbb{T}|}$. We measure these metrics every 100 ms after the warm-up period; for the 2-hop

Parameter	Class B	Class C	Class D
triggered nodes	10, 25, 50	1	2
trigger f	10 Hz	{10 Hz, 4 Hz, 2 Hz}	
$t_{\text{rebroadcast}}$	25 ms	25 ms	-
retransmits	-	3	-
duplicates	2	-	-

Table 4.5 – Protocol Parameters; based on [31] © 2018 IEEE.

neighbors analogous metrics are measured by using Bloom filters. Second, for Class B, C, and D protocols, we mainly study two metrics: *(a)* the delay between the observation of an event (the creation of a message) that informs of a new event and the time this message has been received by all target nodes; *(b)* the fraction of successfully informed nodes, which is an indicator for the reliability of the protocol.

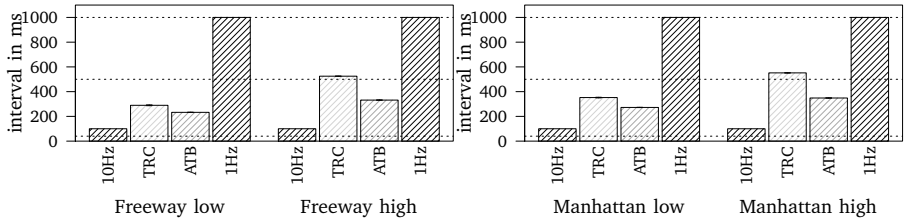
For all simulation experiments we performed at least 10 repetitions with different random seeds for simulating road and network traffic to obtain statistically significant results. Further, we report in every plot the mean value of the selected metric together with its 95 % confidence interval, obtained for all vehicles in all repetitions.

4.5.3 Baseline Experiments

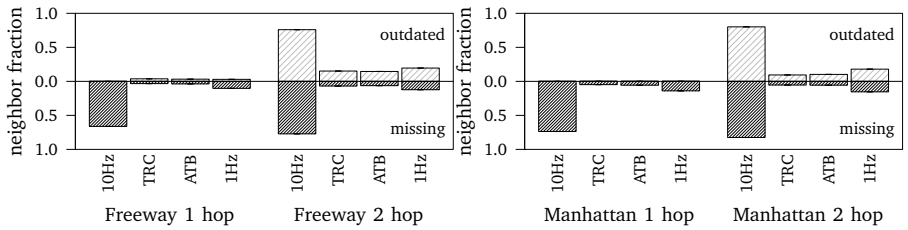
First we evaluate the performance of each individual protocol by running Class A alone. As a next step we use a combination of Class A with Class B, C, and D at the same time for which the performance depends on the neighbor tables established by Class A.

4.5.3.1 Class A Performance

We start with the Class A protocols, which build the foundation for all other broadcast classes because they provide neighbor information. A typical example application is cooperative awareness (cf. Table 4.3). We selected two different configurations for



(a) Resulting beacon interval indicating the latency performance



(b) Neighbor ratio for 1-hop and 2-hop neighbors (high density scenarios)

Figure 4.4 – Performance of Class A protocols for the freeway and Manhattan scenario and different vehicle densities; based on [31] © 2018 IEEE.

static beaconing (1 Hz and 10 Hz) as well as the protocols DCC TRC from ETSI ITS-G5 (see Section 2.2.2) and ATB (see Section 2.2.1).

In Figure 4.4a we plot the mean beacon intervals for all selected protocols. The results for the 1 Hz and 10 Hz protocol options are trivial; i.e., they are 1000 ms and 100 ms respectively. Transmit Rate Control (TRC)'s beacon intervals oscillate between the different protocol states. As the wireless channel becomes more congested, TRC converges to an average delay of 500 ms to relax the wireless channel again. ATB continuously uses smaller beacon intervals, as it is not constrained to discrete steps for beaconing intervals like TRC. When investigating the channel load, it becomes immediately obvious that 10 Hz protocol massively overloads the channel, whereas all other protocols carefully control the channel utilization. This becomes in particular relevant when investigating the quality of Class A protocols in terms of neighbor table maintenance.

Most importantly, we investigate the number of outdated and missing entries in the neighborship data and compared our results to those of an oracle for which we show in Figure 4.4b the results for the high density scenarios. We see for 1-hop-neighbors that the fraction of missing entries is extremely high (around 60 %) when using 10 Hz beaconing. Missing entries are those that have been identified by the oracle but not the selected Class A protocol. This is mainly due to frame collisions and therefore not received neighbor information. All other protocol options perform better, particularly TRC and ATB since they include congestion aware algorithms adapting the beacon interval according to channel conditions. Moreover, we can also observe the impact of the mobility on the Class A protocols: Due to slower driving speeds In the Manhattan scenario the amount of outdated 1-hop neighbors is lower than on the freeway scenario. The results for 2-hop neighbor information are similar, however the amount of outdated neighbors is higher since dissemination time accumulates over 2 hops. The outdated entries are those that should have been pruned, again, according to the oracle. In this case, also a high number of outdated entries can be observed, especially when using 10 Hz beaconing. This is due to many frame collisions on the wireless channel: beacons are lost and direct 1-hop neighbors may be reported as 2-hop neighbors by another direct neighbor. TRC and ATB perform best with respect to this metric.

Further investigations on neighbor table performance are outlined in Chapter 5, where we study the impact of different Bloom filter sizes on the performance of neighbor table management.

Due to the inability of the 10 Hz protocol to provide accurate neighborship information, for the remaining sections in this chapter we only report results for TRC, ATB, and 1 Hz.

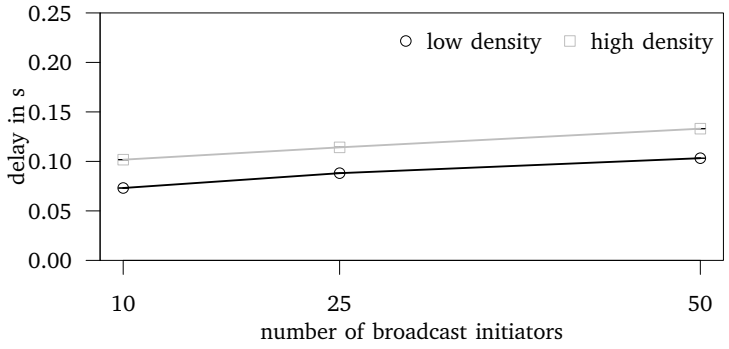
4.5.3.2 Class B Performance

As a next step, we study the performance of our Bloom filter based Class B protocol which is intended to provide urgent information in an N-hop range as needed by many safety related applications (e.g., intersection collision warning). In the following experiments we use $N = 2$. The prime metric we are interested in is the resulting delay and fraction of nodes that successfully received the broadcasts. Consequently, we study an increasing number of selected broadcast initiators, i.e., increasing load on the wireless channel. In Figure 4.5a we show the measured delays in the Manhattan scenarios. Here we only plot results for the experiments using TRC as the Class A protocol. 1 Hz beaconing leads to similar results which also holds for the freeway scenario. For ATB we observe a slightly higher delay which is caused by the lower beaconing interval. The key insight we gain here is that the delay primarily depends on the load on the wireless channel. If we either switch from low density to high density or from a few selected broadcast initiators to a larger number, the delay increases from about 70 ms to more than 150 ms.

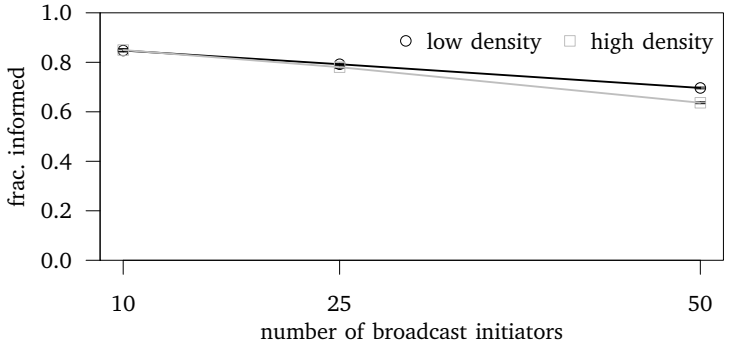
Next in Figure 4.5b we investigate the fraction of informed nodes as a function of the number of broadcast initiators. We observe a significant difference between the low and high density scenarios. When the channel load becomes high due to more broadcast initiators, the success rate shows a decreasing trend. Still, more than 60 % of the vehicles can be informed in the worst case.

In Figure 4.5c, we show the number of selected rebroadcast nodes of our Bloom filter based approach. This number defines the forwarders such that all 2-hop neighbors can be informed. We see that with increasing vehicle density and number of broadcast initiators the number of selected rebroadcast nodes increases as well, which is caused by the increased network load. In particular, when taking into consideration the results of our Class A neighbor table experiments shown in Figure 4.4, missing 2-hop neighbors of a node will lead to an (additional, but not necessarily needed) selection of further rebroadcast nodes to reach all 2-hop neighbors available in the originator's neighbor table. However, this additional selection will help to increase the probability of successful dissemination of the message.

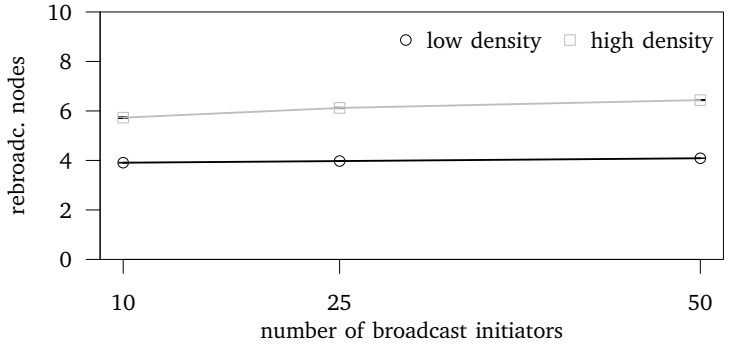
To show the advantage of our Bloom filter based rebroadcast protocol, we compare it against a classical greedy approach (as used, e.g., in DV-CAST [142]). Here we select two rebroadcast nodes from a node's neighbor table, namely the leftmost and rightmost neighbor. In Figure 4.6 we show the fraction of informed nodes for different numbers of broadcast initiators and this protocol configuration. Compared to our Bloom filter approach in Figure 4.5b, we observe a lower number of informed nodes, and this fraction is also much more negatively affected by higher channel load caused by higher vehicle densities. We conclude that the Bloom filter based



(a) Delay for low and high density Manhattan scenarios



(b) Fraction of successfully informed nodes for low and high density Manhattan scenarios



(c) Selected rebroadcast nodes for low and high density Manhattan scenarios

Figure 4.5 – Class B protocol performance for different numbers of broadcast initiators. Plotted are the results using TRC as the Class A protocol and the Manhattan scenario; results are similar for the freeway scenario; based on [31]

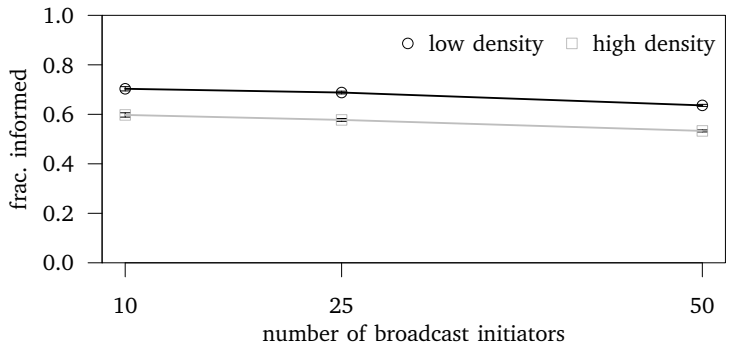


Figure 4.6 – Fraction of successfully informed nodes for low and high density Manhattan scenarios using a greedy approach for message dissemination. Plotted are the results using TRC as the Class A protocol and the Manhattan scenario; based on [31] © 2018 IEEE.

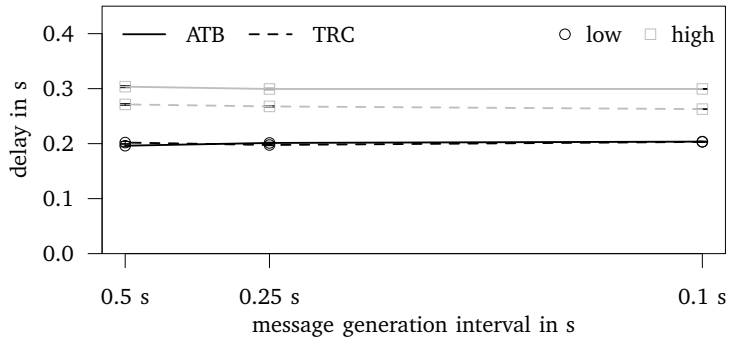
solution is an effective approach for 2-hop data dissemination in larger networks for different network densities.

4.5.3.3 Class C Performance

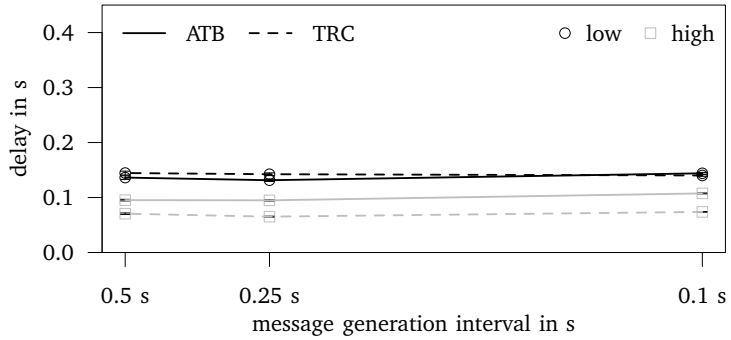
As a next step we performed the same experiments for Class C protocols to measure the resulting delay and the fraction of nodes that successfully received the message. For this type of protocols, an example application is an emergency vehicle warning system (cf. Table 4.3), which disseminates messages along the road towards some geographic position. To observe the behavior of the protocol, we use decreasing message generation intervals, i.e., we are slowly increasing the network load.

As outlined for the Manhattan scenario in Figure 4.7a, the delay for Class C protocols depends on two key factors. First, the channel load plays an important role: for higher traffic densities, the channel becomes more loaded, which translates to higher message delays. Secondly, the more up-to-date the neighbor tables are and the more likely it is to have sufficient rebroadcasters available in the direction of the geocast, the lower delays can be observed. This can be seen in the freeway scenario in Figure 4.7b where the delay in the high density scenario is lower than in the low density.

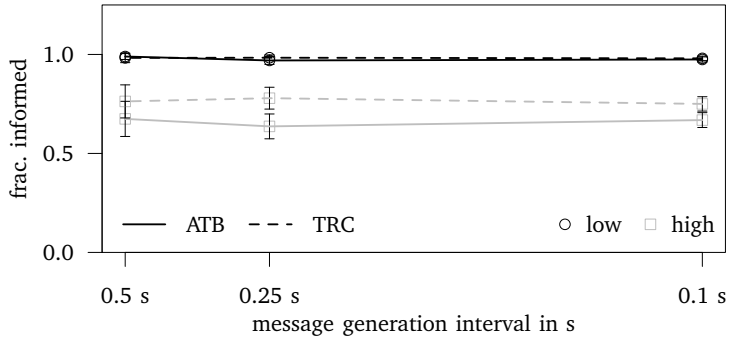
However, the delay is only partly telling the story, it is also necessary to have a look on the reliability of the protocol. In Figure 4.7c we show the fraction of informed nodes, in which we see that a reduced channel load leads to a larger fraction of successfully informed nodes. For the freeway scenario the received fraction is in general lower due to the higher mobility (see Figure 4.4), but shows similar qualitative effects.



(a) Delay for Manhattan scenarios



(b) Delay for freeway scenarios



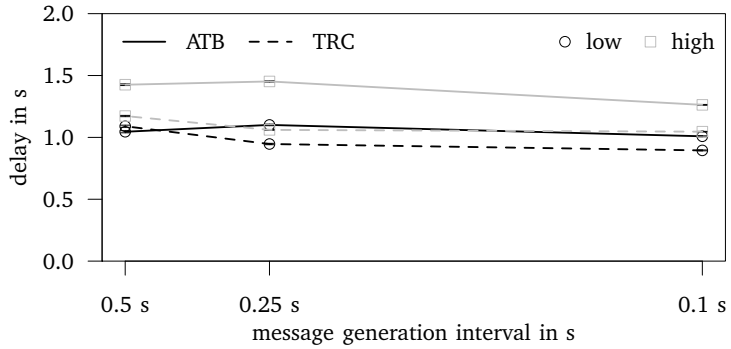
(c) Fraction of successfully informed nodes for Manhattan scenarios

Figure 4.7 – Protocol performance for Class C using low and high density scenarios and different message generation intervals. Plotted are the results using the ATB and TRC Class A protocols; based on [31] © 2018 IEEE.

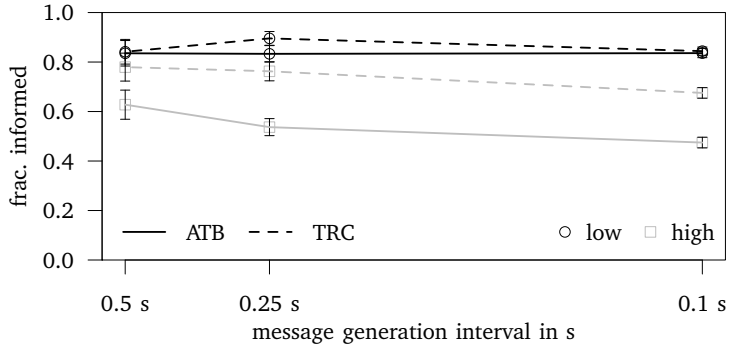
4.5.3.4 Class D Performance

Finally we want to investigate the performance for Class D based protocols. Recall that a possible application using this broadcast class are Traffic Information Systems (TISs). For our experiments we periodically selected two vehicles to insert information items into their local knowledge base. Each entry is configured to have a destination position of the other selected vehicle's geo-position.

At first we have a look at the resulting delay as plotted in Figure 4.8a for the Manhattan scenarios. As can be seen, the delay is not very sensitive in the low density scenarios but becomes larger in the high density scenario, especially for the ATB protocol. We also note the dependency on the underlying Class A protocol. The slightly lower beaconing interval of ATB compared to TRC leads to a higher channel utilization and thus slightly larger delays compared to the TRC protocol in the high density scenarios. Similar effects are observed in the freeway scenario, as well as for 1 Hz beaconing. We can confirm this trend when looking at the fraction



(a) Delay for Manhattan scenarios



(b) Fraction of successfully informed nodes for Manhattan scenarios

Figure 4.8 – Class D Performance for different message generation intervals in low and high density scenarios. Plotted are the results using the ATB and TRC Class A protocols; results for 1 Hz are comparable to TRC; results are similar for the freeway scenario; based on [31] © 2018 IEEE.

of informed vehicles in Figure 4.8b. We notice that our Class D protocol is able to inform more than 80 % of vehicles on the road in the low density scenarios. With increasing road traffic, the Class A beaconing protocol leaves less channel capacity for class D protocols, thus the fraction of informed nodes decrease. This is perfectly in line with our integrated broadcast approach using the EDCA subsystem of the IEEE 802.11p MAC. This way higher prioritized protocols (Class A) have a higher probability for channel access than lower prioritized protocols (Class D). For 1 Hz as well as for the freeway scenario similar trends can be observed.

4.5.4 Studying the Integrated Class Based Broadcast Architecture

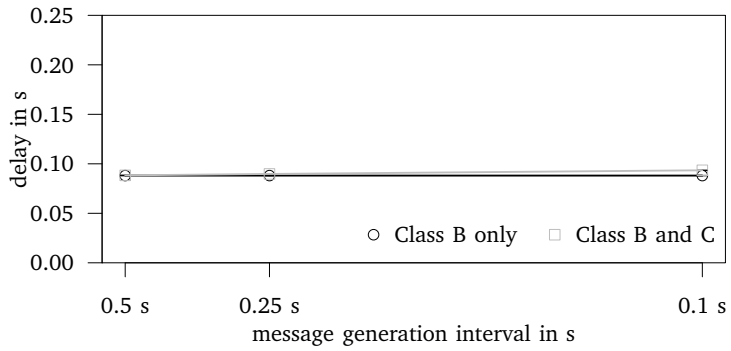
A very important aspect in our class-based broadcasting architecture is the interplay among the different classes when they are operated at the same time. This way the advantages and capabilities of our novel class-based broadcasting architecture for vehicular networks becomes even more visible when we look at the integrated performance in more holistic experiments. To achieve this, we stepwise enable all protocols and investigate the dependencies of the protocol classes.

4.5.4.1 Dependencies Between Class B and Class C

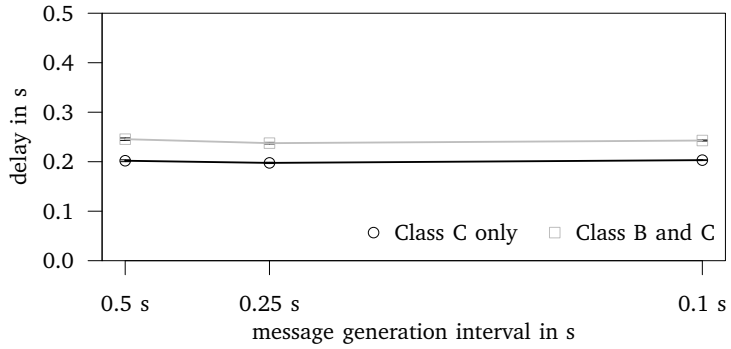
First we investigate the dependencies between Class B and Class C protocols when they simultaneously access the wireless channel. For this we configured a setup using Class A for neighbor table management as well as cooperative awareness, Class B for emergency or warning messages, e.g., about an imminent traffic accident, as well as Class C for informing other cars about lower priority events, e.g., green light speed advisory (cf. Table 4.3).

In Figure 4.9 we show the results for the low density Manhattan scenario by using TRC as a Class A protocol. We keep the number of broadcast initiators of Class B constant (25) and vary the data rate of Class C messages. Figure 4.9a indicates the impact of Class C on Class B in terms of communication delay. As can be seen, the delay of Class B messages is not effected at all, which is exactly the expected result and mainly accomplished by the EDCA subsystem of the IEEE 802.11p MAC. It gives higher priority messages better channel access probabilities and thus lowers the time to wait for a free channel. Moreover, Class B messages are also not affected in terms of the fraction of informed nodes.

However, if we look at the resulting performance of our Class C protocol, we see a high impact of the protocol interaction. Due to the channel load caused by Class B, which is working on a higher EDCA priority level compared to Class C, the measured delay of Class C messages increases by a factor of about 25 %, as we show in Figure 4.9b. The fraction of informed nodes does only marginally depend on the



(a) Delay of Class B messages



(b) Delay of Class C messages

Figure 4.9 – Combined Class A, B, and C performance for different message generation intervals; results plotted for using TRC as Class A protocol and a low density Manhattan scenario; results for ATB and 1 Hz are similar to TRC; results are similar for the freeway scenario; based on [31] © 2018 IEEE.

concurrently running Class B protocol, which is caused by the fact that there is still sufficient time for the Class C protocol to deliver the messages using Geocasting; the delivery is simply delayed. Only when vehicle density is high, packet loss due to interference has a negative impact to the delivery rate. In particular, using ATB as Class A protocol in the high density Manhattan scenario we observe slightly lower delays for Class C operation when Class B protocols are enabled, however with significantly lower delivery rates which are caused due to an overloaded channel.

4.5.4.2 Dependencies Between Class B/C and Class D

In a similar experiment we investigate the influences between Class B and Class D protocols. For these experiments we use again Class A for neighbor table management and cooperative awareness, Class B for emergency messages e.g., about a lane change, and Class D for non-urgent events such as informing about longer lasting traffic jams on a road network provided by a TIS.

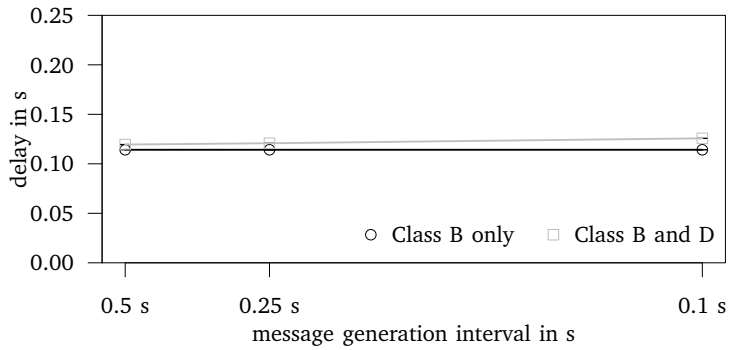


Figure 4.10 – Combined Class A, B, and D performance (delay of Class B messages) for different message generation intervals; results plotted for using TRC as a Class A protocol and a high density Manhattan scenario; results for ATB and 1 Hz are comparable to TRC; results are similar for the freeway scenario; based on [31] © 2018 IEEE.

In Figure 4.10 we show the results for the high density Manhattan scenario by using TRC as a Class A protocol. As in the previous experiments we keep the number of broadcast initiators for Class B messages constant (25) and vary the data rate of Class D messages. We observe that Class D messages have nearly no impact on the Class B protocol. Our Bloom filter based Class B protocol generates and broadcasts even more messages with an increasing number of cars. When combining this with a second protocol, the channel gets even more congested. The fraction of informed cars keeps almost the same too. The causes for this effect were already discussed when studying the dependencies between Class B and Class C protocols.

When switching to Class C together with Class D protocols, we observe a similar behavior. Possible applications for this are listed in Table 4.3 (e.g., rear end collision warning for Class C and TIS using Class D).

4.6 Lessons Learned

In this chapter we proposed a novel, integrated, context-aware, broadcast-based Network layer for supporting past and future VANET applications. We also designed four broadcast classes that match the requirements of all known applications and possible future use cases in IVC. We showed in extensive simulation studies that these classes not only can co-exist on the same Network layer, but also make use of cross-protocol functionality. We analyzed the performance of the proposed protocols in detail and discussed their properties and their ability for co-existence on the same wireless channel. We see our integrated broadcast protocol approach to provide extensibility and applicability for future protocol designs.

As open research question, we highlight that the neighbor table maintenance is still dependent on pre-defined timeout values for the Class A protocols to purge neighbors from the neighborship table, as well as the Bloom filter size which has a non-negligible impact on the performance of the whole system. Both aspects are investigated in detail in Chapter 5.

Chapter 5

Bloom Filter Based Neighbor Management

In the previous chapter we developed an integrated networking architecture for different application domains and presented with *class B* a category of protocols which aims to transmit important information among multiple hops; in our case we used 2 hops. The performance heavily depends on the selection of a good set of forwarding nodes, which itself depends on a good estimate of the local neighborhood of a node. This way, in the following chapter we focus on the topic of neighbor management in highly dynamic networks and present *Bloom Hopping*: a neighbor table and message forwarding protocol specifically designed for highly dynamic networks in the vehicular domain.

The content of the following chapter is based on the following peer-reviewed publication:

- F. Klingler, R. Cohen, C. Sommer, and F. Dressler, "Bloom Hopping: Bloom filter based 2-Hop Neighbor Management in VANETs," *IEEE Transactions on Mobile Computing*, 2018, available online, © 2018 IEEE.
  My contribution in this work was the design and the implementation of the 2-hop neighbor table algorithm as well as the analysis and evaluation of it in different scenarios.

5.1 Motivation

Until now we learned that beaconing is periodic dissemination of control messages by each vehicle to its immediate neighbors, called 1-hop neighbors. These transmitted messages are beneficial for cooperative awareness and road traffic safety applications and have been standardized as Basic Safety Messages (BSMs) and Cooperative Awareness Messages (CAMs) [9]. The Information a single beacon carries is usually a node's ID as well as the current status of the vehicle, e.g., position, speed, and heading.

In Chapter 4 we have shown that it would be beneficial for a node to maintain not only the list of its 1-hop neighbors, but also of its 2-hop neighbors; in other words: the neighbors of its neighbors. This has also been found beneficial in the literature [137]. A sample use case for this are applications where a message needs to be forwarded to all the nodes in a given geographic vicinity. Here, instead of using flooding, which is inefficient and can congest the wireless channel, the originating node can broadcast the message and annotate in it a subset of its 1-hop neighbors which should forward this message. As we learned in the previous chapter we choose this subset such that it contains a minimum number of nodes to cover the 2-hop neighbors of the originator.

The easiest way to maintain a list of a node's 2-hop neighbors is to include the full list of all the node's 1-hop neighbors in the beacon messages. However, this naïve

approach would probably only be feasible for very small or sparse scenarios with very few vehicles being involved. The main problem is the scalability because the beacon size is directly proportional to the number of neighbors. Suppose that regular 6 B MAC addresses are used to uniquely distinguish nodes. On a 6-lane freeway having nodes equipped with wireless communication by using a communication distance of 500 m, a typical number of 1-hop neighbors could be 150. Thus, every beacon needs to carry an expected amount of $150 \times 6\,\mathrm{B} = 900\,\mathrm{B}$ of neighbor information. We clearly see that this is not only a substantial beacon overhead, but – more severe – reduced reliability of beacon messages due to hidden terminal effects and an increased beacon collision probability on the wireless channel. The reason is that when a beacon is broadcast by node v, the message most probability cannot be decoded by a neighbor u of v, if another neighbor of u (a neighbor of u who is a 2-hop neighbor of v) transmits at the same time with v.

In this chapter, we propose a scheme that uses Bloom filters [73] for disseminating and maintaining 2-hop neighborship information by extending the approach presented in Chapter 4 and sections 4.4.1 and 4.4.2. We show that, if used right, employing Bloom filters significantly reduces the length of the beacon messages, and thus keeping channel load and packet collision probability considerably lower than in a naïve scheme that incorporates full neighbor information into the beacons. However, the efficiency of Bloom filters comes with the cost of false positives, in which a check whether an element is contained in a Bloom filter could falsely lead to a positive answer, however it was never inserted.

In this chapter we address the following important decisions:

- What information should be included in each beacon?
- When to add or remove neighbors to/from the local 2-hop neighbor table?
- When to send neighbor information, and at which frequency?

Our scheme provides the basis for a variety of other applications that could build on top of neighborship information: routing and clustering, for example. Requirements of those applications are often to not only have 2-hop neighbor information, but N-hop information for some $N > 2$. Besides this, the usage of Bloom filters for neighbor tables provides also privacy preserving functionality due to their inherent nature to not store plain data but hashes [143].

We want to mention that our approach can also be adapted for $N > 2$ by using multiple Bloom filters indicating different hop ranges, and can also be used for other types of networks, particularly if the network topology changes rapidly.

In summary, our contributions are as follows:

- We introduce a Bloom filter based 2-hop neighbor management scheme specifically designed for dynamic wireless networks (Section 5.3).

- We demonstrate the applicability of this 2-hop neighbor information by taking advantage of our multi-hop broadcast protocol, which we call *Bloom Hopping* (Section 5.4).
- We study the performance of our Bloom filter based neighbor management system both, investigating the quality of neighbor information, and by using our proposed Bloom filter based broadcast protocol (Section 5.5).

5.2 Preliminaries

From the previous chapters we already know that neighborship information is the underlying basis for routing, clustering, and message dissemination in Mobile Ad Hoc Networks (MANETs). Yet, when mobility of nodes is high and thus the underlying network topology is very dynamic, this still constitutes a fundamental research problem, particularly if 2-hop neighbor information is needed. In particular this holds for highly dynamic wireless networks like Vehicular Ad Hoc Networks (VANETs), where protocols need to maintain 1-hop [66], [72], [142], [144] or 2-hop [145]–[148] neighbor information as accurate as possible.

A recent approach using Bloom filters for calculating connected dominating sets in vehicular networks has been presented by Na Nakorn, Ji, and Rojviboonchai [78]. The main idea of this approach is to extend the work in [149] by using Bloom filter set-operations to lower the algorithm's complexity for finding a connected dominating set in an ad hoc network. Particularly, the authors reduce the algorithmic complexity of [149] from $\mathcal{O}(n^5)$ to $\mathcal{O}(n)$ by using two different operations on Bloom filters: *union* (w/o loss of information), and *intersection* (w/ loss of information). Consequently, the authors propose a fixed size beacon structure to incorporate a list of neighbors as well as a list of packet identifiers within a single Bloom filter. The evaluation of the algorithm shows that it is possible to reduce the beacon overhead compared to the original solution, even though the amount of received nodes and the retransmission overhead remains the same.

Bloom filters have also been found beneficial for multi-hop message dissemination in general networks [74], [75] and VANETs [137], [150]. Sometimes, protocols do not explicitly require 2-hop neighbor information like the approach presented in [151]. Here, for each message to be forwarded, a Bloom filter has to be included in the message and a node rebroadcasts this message if it can contribute to reach additional neighbors. For each forwarded message the node appends his local neighbor set in the Bloom filter included in that particular rebroadcast. As this scheme is based on contention based forwarding it is not suitable for the usage within a general network protocol stack, since other applications running on the same node may also rely on exact neighbor information. We fill this gap by focusing

on (a) efficient *neighbor management* algorithms to decide when to add which neighbors into our neighbor table, such that we can use that information to (b) efficiently choose fitting nodes from that neighbor table to rebroadcast our message. This network architecture builds a basis for future applications to retrieve neighbor information.

Other protocols for VANETs, like TO-GO [137], use Bloom filter encoded 2-hop information to allow probabilistic membership tests for selecting a set of forwarding nodes for multi-hop geocast. Even in wireless sensor networks Bloom filters are considered for routing and data discovery. In this area different protocols have been proposed [76], [152], [153], yet, their performance is limited to rather stationary topologies. Bloom filter structures are also used in wireless networks for routing in hierarchical topologies [154], or by aggregating forwarding information to provide geographical routing [155].

In this chapter we show that the efficient use of Bloom filters not only decreases the overhead of transmissions, i.e., the beacon size and thus channel utilization, but actively improves the application layer performance. We investigate the use of the specific nature of Bloom filter encoded 2-hop neighbor information to achieve very efficient multi-hop data dissemination. This way, our system builds a fundamental basis for supporting a wider range of communication protocols with relevant 2-hop neighbor information.

5.3 Neighbor Table Management

In this section we describe the scheme that allows each vehicle to build and maintain a Bloom filter based neighbor table of its 2-hop neighbors. First, each node x maintains a table with the identities of its 1-hop neighbors (how to determine which nodes are 1-hop neighbors is detailed in Section 5.3.1). Next, based on the information contained in this table, node x generates a Bloom filter of all available identities in the table and includes this Bloom filter in the beacons it periodically broadcasts with interval I. In each beacon we transmit, we include the measured channel utilization $b_t = \frac{t_{\text{busy}}}{t_{\text{busy}}+t_{\text{idle}}}$ as the fraction of the time the wireless channel was sensed busy since the last sent beacon, i.e., $t-I$ and t. Additionally, each beacon includes the current interval I of the sending node to indicate at which time the receivers (most probably) can assume to get the next beacon. We also include GPS positioning information of the node. Finally, the neighbor table is annotated with information about each 1-hop neighbor y, including the node's last broadcast Bloom filter, GPS position, distance, and time when the last beacon was received, as well as the last value of b_t.

This scheme, in particular the use of Bloom filters within the 1-hop neighbor table allows node x to make probabilistic decisions about its 2-hop neighbor set

by performing bitwise OR (union) operations on all the Bloom filter vectors. In the following we focus on what we believe is the most important application for acquiring 2-hop neighbor information. We achieve this by choosing a good set of 1-hop neighbors for forwarding, to cover all of the 2-hop neighbors. This "good" set should be as small as possible, but contain neighbors with which node x has a good wireless connectivity. In particular, we investigate the research question *what is a good neighbor* and propose algorithms to determine nodes to rebroadcast messages. We discuss this selection process in detail in Section 5.4.

5.3.1 Maintaining 1-Hop Neighbor Tables

To build a neighbor table, in general a node x may consider any beacon it receives as coming from a direct 1-hop neighbor for maintaining its list of neighbors $\mathbb{X}$ and the corresponding Bloom filter $\mathcal{X} \leftarrow \mathbb{X}$. It is this Bloom filter that will be broadcast periodically by node x to all nodes in its vicinity; we denote the received Bloom filter of node y at node x as $\mathcal{X}_y$.

Unfortunately, communication channels are sometimes asymmetric due to interference, that means, although node x receives a beacon from node y, the same node y might not be able to receive messages from node x. Consequently, node x must verify that y receives its beacons before it can consider node y as an acknowledged 1-hop neighbor. To address this problem, whenever node x receives the beacon from y, it checks whether its own identity is contained in the Bloom filter sent by y. If this probabilistic check is successful, then there is a high probability (depending on the false positive probability of the Bloom filter) that bidirectional communication is possible. If the check failed, then either node y has not yet received a beacon of x, or the communication channel is asymmetric which means that the two nodes should not consider each other as neighbors.

More formally, we can create a subset of acknowledged neighbors $\mathbb{X}'$ based on the neighbors set $\mathbb{X}$ of node x as

$$\mathbb{X}' = \left\{ u \in \mathbb{X} : x \in \mathcal{X}_u \right\} . \tag{5.1}$$

Having this in mind, we have an accurate Bloom filter based neighbor table at hand that only covers nodes that are able to communicate bidirectionally based on a (very small) false positive probability depending on the Bloom filter size. In the following, we call only these neighbors 1-hop neighbors.

5.3.2 Maintaining 2-Hop Neighbor Tables

Maintaining only 1-hop information in a network does not give viable information about the topology of the underlying network. To get an overview about the network

topology, a node can, for example, include the number of neighbors it sees in the periodically exchanged beacons to let other nodes infer about the network connectivity of this neighboring node. However, using such an approach, a potential receiver cannot calculate in detail how the neighbor set of one of its neighbors differs from its own neighbor set. Therefore, having information about 2-hop neighborship at hand, it is possible to derive the difference of the neighbor set among multiple nodes and, thus, gain further information about whether a node could be beneficial when selected as forwarder. However, in contrast to the approach presented in Chapter 4 and section 4.4.1, here we provide an algorithm to automatically prune entries in the neighbor table whenever nodes are not available in the network anymore, without using a predefined and fixed timeout value. This is in particular important, since such timeout values for removing neighbors from the neighbor table need to be adapted for different traffic densities and scenarios.

5.3.2.1 Algorithm

To do so, we extend the concept of our neighbor management approach to 2-hop neighbor tables. First, each beacon contains a Bloom filter of a node's 1-hop neighbors as outlined in Section 5.3.1. This allows us to build a per node 2-hop neighbor table consisting of the respective Bloom filters consisting of all 2-hop nodes that we know from our 1-hop neighbors. However, due to interference, a node may not be aware of some 1-hop neighbors. Further, neighbor tables might be rather unstable, i.e., fluctuating entries because of lost beacon messages caused by the high mobility of nodes and the underlying road topology and obstacles attenuating the signal between sender and receiver. These effects may lead to wrong neighbor tables with two implications: Too many (wrong) entries in the 2-hop neighbor table, and missing entries in the 1-hop neighbor table. In other words, a neighbor can be falsely reported as 2-hop neighbor by a 1-hop neighbor, although in fact this 2-hop neighbor should be in the 1-hop neighbor set as it is inside the communication range of the receiving node. This becomes even severe, when non-symmetric communication channels between neighbors exist.

To address the issue of wrongly assigning nodes as 1-hop or 2-hop neighbors, one possibility is to take advantage of a node's position and other nodes' Bloom filters as well as their announced beacon time. Whenever our neighbor table algorithm wants to remove a neighbor from the 1-hop neighbor table, it checks whether the node's ID is included in the Bloom filter of other nodes, which are more distant to the node that is about to be removed and which have sent a beacon after the announced beacon time of the node to be pruned from the neighbor table. In other words, we check if there is a more distant neighbor which recently transmitted a beacon and still announces the node we want to remove in its 2-hop neighbor set.

Let us assume that node x is checking its neighbor table for stale entries and wants to remove node y from the neighbor table. Then we can decide whether to keep the neighbor y (but mark it as hidden) in three steps:

1. A node x categorizes nodes in its neighbor table according whether they are marked as hidden or not, storing them as

$$\overline{\overline{\mathbb{X}}} = \left\{ u \in \mathbb{X}' : u \text{ is hidden} \right\}, \tag{5.2}$$

$$\overline{\mathbb{X}} = \left\{ u \in \mathbb{X}' : u \text{ is not hidden} \right\} = \mathbb{X}' \setminus \overline{\overline{\mathbb{X}}}. \tag{5.3}$$

2. Next, node x calculates a set of closer neighbors $\mathbb{C}_{x,y} \subseteq \overline{\mathbb{X}}$. It derives the set of all visible 1-hop neighbors of x which are equally or more distant from node y than node x as

$$\mathbb{C}_{x,y} = \left\{ u \in \overline{\mathbb{X}} \ : \ \text{distance}(u, y) \geq \text{distance}(x, y) \right\}, \tag{5.4}$$

   and further the set of neighbors $\mathbb{D}_{x,y} \subseteq \mathbb{C}_{x,y}$ from which a beacon has been received after the announced beacon time of node y

$$\mathbb{D}_{x,y} = \left\{ u \in \mathbb{C}_{x,y} \ : \ \mathrm{t}_{\text{b.-received}}(u) \geq \mathrm{t}_{\text{b.-announced}}(y) \right\} \tag{5.5}$$

   as well as a set of neighbors $\mathbb{E}_{x,y} \subseteq \mathbb{D}_{x,y}$ that have a lower or equal channel busy ratio b_t in comparison to node x

$$\mathbb{E}_{x,y} = \left\{ u \in \mathbb{D}_{x,y} \ : \ b_t(u) \leq b_t(x) \right\}. \tag{5.6}$$

   This calculation is essential to select only those neighbors $\mathbb{E}_{x,y}$ that have a reasonably high probability in order to include node y in their neighbor table and do not suffer from a high amount of non-relevant 2-hop neighbors, which would increase the Bloom filter false positive rate.

3. Finally, node x only deletes node y from its neighbor table if y is not in any of the resulting neighbors' Bloom filters. More formally: if $\mathbb{F}_{x,y} = \left\{ u \in \mathbb{E}_{x,y} : y \in \mathcal{X}_u \right\}$ and $\mathbb{F}_{x,y} = \emptyset$. In the other case it marks y as hidden. We have to note that when a node broadcast beacons, the hidden neighbors are omitted from the Bloom filter.

In essence, we now can summarize all 2-hop neighbors in the Bloom filter $\mathcal{X}''$ as

$$\mathcal{X}'' = \bigcup_{u \in \mathbb{X}'} \mathcal{X}_u \,. \tag{5.7}$$

The key contribution of our solution is that it keeps the neighbor tables itself intact, but limits or avoids the problem of wrongly annotating neighbors as 1-hop or 2-hop neighbors. Still, this depends on the chosen size of the Bloom filter and the resulting false positive rate. This way we can compensate short lasting outages of beacons and do not need to remove this node from our neighbor table which leads to more stable entries in the neighbor table.

5.3.2.2 Time Complexity

As a next step we analyze the complexity of our 2-hop neighbor table approach in time domain. This is in particular relevant, since future implementations of communication protocols for VANETs in cars will most probably be deployed on Electronic Control Units (ECUs) providing limited computational power and running various processes for different applications at the same time. For maintaining 2-hop neighbor tables the time-complexity of the proposed scheme basically depends on Bloom filter membership tests (Equation (5.1)) and calculations of subsets of 1-hop neighbors (Equations (5.3) to (5.6)). The processing time of a Bloom filter membership test does not depend on the number of already inserted elements but only on the number of used hash functions k, which can be derived by Equation (2.12). In most cases k is relatively small, e.g., $k = 37$ for a Bloom filter size $m = 300\,\mathrm{B}$ and an element count $n = 45$. Thus, to test whether a given element is part of the Bloom filter, the time-complexity is $\mathcal{O}(k)$ since k hash values need to be calculated which also holds for inserting an element to the Bloom filter.

To maintain the neighbor tables as outlined in Section 5.3.2.1, thus to decide whether a node should be deleted or just marked as hidden, the following time-complexities hold: Deriving the set of hidden nodes in Equation (5.3), of closer neighbors in Equation (5.4), and of nodes from whom a beacon was received after the announced beacon time in Equation (5.5) fall each into $\mathcal{O}(n)$, where n denotes the number of 1-hop neighbors. To optimize this process at runtime, all necessary calculations can be performed within one traversal of the 1-hop neighbor table. For each node remaining in $\mathbb{E}_{x,y}$, we perform a membership test in the corresponding Bloom filter, for which the traversal through the set can be stopped whenever a successful membership test was performed. Only in the worst case (when a node needs to be marked as hidden), the whole traversal through the list needs to be performed.

The overall process is repeated for all 1-hop neighbors which leads to a time-complexity of $\mathcal{O}(k \times n^2)$ for n 1-hop neighbors and k hash functions. When using $n = 45$ 1-hop neighbors representing a typical scenario, a Bloom filter size of 300 B, and a number of $k = 37$ hash functions, we need to perform 74 925 calculations of hash functions whenever an update of the neighbor table is needed, which is the

case whenever we send a beacon. For a CPU like Intel Atom® N450, which offers 1.6 MHash/s [156] when employing the SHA256 hash function, this would lead up to 749 250 calculations per second and, thus, to a very high CPU utilization when using 10 Hz beaconing. However, as we show in Section 5.5.3 and already investigated in the previous chapter in Section 4.5.3.1, a beaconing rate of 10 Hz would very likely overload the wireless channel. Therefore, using lower beaconing frequencies in the range of up to 1 s is recommended and will also decrease the computation overhead. Moreover, Bloom filters are not restricted to SHA256 hash functions and can also take advantage of different and more inexpensive algorithms which again contributed on lowering the computational effort. A typical optimization for Bloom filters [157] uses the fact that k different hash functions can be replaced by just two hash functions. More formally, this leads to: $g_i(x) = h_1(x) + i \times h_2(x)$, where the index i denotes the hash function within k and $g_i(x)$ denotes the resulting hash function used by the Bloom filter. With this optimization, the time-complexity of our algorithm degrades to $O(n^2)$. Finally, computations for our neighbor table algorithm can even be parallelized since there are no dependencies among traversals through the neighbor table.

5.3.3 Cardinality Estimation of Bloom Filters

As outlined in Section 4.4.1, our Bloom filter dissemination algorithm depends on the estimation of the cardinality of a Bloom filter, that is, the number of entries which have been added to a Bloom filter.

To do so, we conducted several Monte-Carlo simulations to investigate the performance of estimating the cardinality, i.e., to approximate the number of elements in a Bloom filter and the false positive rate (cf. Section 2.3). In particular, our simulation setup consists of two Bloom filters $\mathcal{A}$ and $\mathcal{B}$ having the same bit length and using the same set of hash functions. We randomly add elements to both Bloom filters and made sure that ⅓ of the entries are added to both filters, i.e., both Bloom filters overlap to 50 %. Next, we run simulations for different Bloom filter sizes and repeated each experiment at least 100 times with different seeds for the random number generator to obtain statistically significant results.

In Figure 5.1 we present how good the cardinality of a Bloom filter $|\mathcal{B}|$ can be approximated compared to the true amount of elements $|\mathbb{B}|$ which have been inserted in this particular Bloom filter. The approximation of the cardinality closely matches the ideal behavior up to the point where the Bloom filter is filled, i.e., (almost) all bits are set to 1. This denotes the upper bound of elements to be inserted in a Bloom filter of a given size.

For a second experiment we investigate the applicability of calculating $\mathcal{A} \cap \mathcal{B}$, the intersection of two Bloom filters. Our simulations indicate that the fraction of false

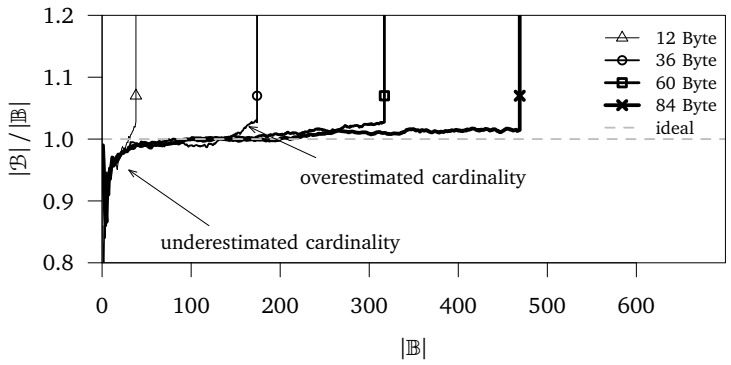


Figure 5.1 – Performance of estimating the number of inserted elements in a Bloom filter. The line width depicts the Bloom filter size; the dashed line indicates the ideal behavior; based on [32] © 2018 IEEE.

positives is much lower when building a Bloom filter from scratch containing only the intersecting elements than the resulting Bloom filter derived by the intersection operation.

However, overall objective is to assess the number of 2-hop neighbors that are not direct 1-hop neighbors. Thus, the quality of a specific Bloom filter in order to estimate diff($\mathcal{A}, \mathcal{B}$), indicating the number of entries in a foreign filter $\mathcal{B}$ that are not part of a local filter $\mathcal{A}$, can mathematically be expressed as

$$|\mathcal{A} \cup \mathcal{B}| - |\mathcal{A}| \stackrel{?}{=} \text{oracle} \stackrel{?}{=} |\mathcal{B}| - |\mathcal{A} \cap \mathcal{B}| \,. \tag{5.8}$$

In Figure 5.2 we show the results for these experiments normalized to the ground truth, that is, using the real number of inserted elements and not relying on the cardinality estimation of Bloom filters. The estimation using the union of the Bloom filters follows the ideal line until one of the two Bloom filters is completely filled. Only a slight overestimation when the Bloom filter is getting close to saturation can be seen. In contrast to the union operation, the approach using the intersection significantly underestimates the number of new elements. We thus conclude that the use of

$$\text{diff}(\mathcal{A}, \mathcal{B}) = |\mathcal{A} \cup \mathcal{B}| - |\mathcal{A}| \tag{5.9}$$

is the most appropriate option to estimate the amount of additional neighbors among a Bloom filter to another Bloom filter which we will use in the following.

5.4 Bloom Filter Based Multi-Hop Broadcast

As already outlined in the previous sections and in Chapter 4, 2-hop (N-hop) neighbor tables are beneficial for a broad range of applications to improve the management of

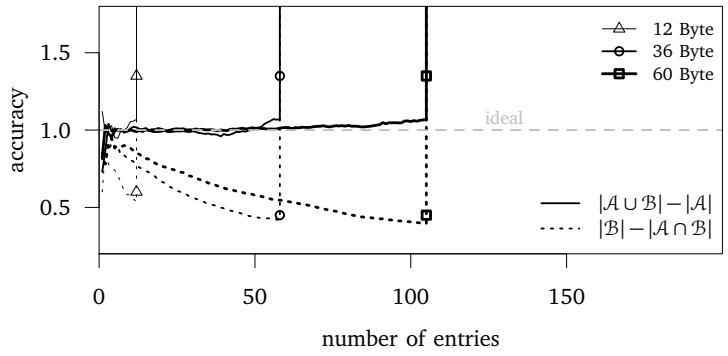


Figure 5.2 – Performance of estimating the accuracy for the number of additional entries provided by a certain Bloom filter normalized to the ground truth. The line width depicts the Bloom filter size; dotted and solid lines represent the two calculation options (union, intersection); based on [32] © 2018 IEEE.

dynamic networks. In this section, we show how the specific Bloom filter based neighbor management protocol can be extended to also support efficient broadcast-based data dissemination, e.g., for warning messages or as the basis for general VANETs. In general this approach is similar to the scheme presented in Section 4.4.2, however now we specifically take advantage of our novel 2-hop neighbor management approach outlined in Section 5.3.2.1 and call our multi-hop message dissemination algorithm *Bloom Hopping*.

In essence, node x chooses the set of re-broadcasters $\mathbb{R}_x$ as the minimum subset of visible 1-hop neighbors $\mathbb{X}'$ to cover all of its 2-hop neighbors.

Technically, a node x starts with an empty set of re-broadcasters $\mathbb{R}_x$ and a Bloom filter of already-covered 2-hop neighbors $\widehat{\mathcal{X}}''$, initially contains the symmetric 1-hop neighbors, i.e., $\widehat{\mathcal{X}}'' \leftarrow \mathbb{X}'$. As a next step it repeatedly chooses the best node u as

$$u = \underset{u \in \mathbb{X}'}{\arg\max} \left(\text{diff}(\widehat{\mathcal{X}}'', \mathcal{X}_u) \right) \tag{5.10}$$

and integrates the chosen 1-hop neighbor u to $\mathbb{R}_x$ as well as its Bloom filter $\mathcal{X}_u$ to $\widehat{\mathcal{X}}''$, that is $\widehat{\mathcal{X}}'' \leftarrow (\widehat{\mathcal{X}}'' \cup \mathcal{X}_u)$. The rebroadcast node selection process ends when all 2-hop neighbors (or a sufficiently large subset) are covered. This can be derived by comparing $\widehat{\mathcal{X}}''$ and $\mathcal{X}''$.

The set $\mathbb{R}_x$ contains a set of 1-hop neighbors which are intended to rebroadcast the message. Usually $\mathbb{R}_x$ is quite small (e.g., it converges to 2 in freeway scenarios), thus it is added to the broadcast message. Moreover, to prevent collisions between rebroadcasting 1-hop neighbors due to nodes starting their rebroadcast at the same time, node x adds to the message the value of an artificial delay for each such neighbor. One possibility to derive this delay would be to make it dependent on the actual link quality reported by that particular neighbor in the periodic beacons.

Another possibility is to derive it by a more sophisticated approach like the following: For each pair of neighbors u, v which have in their Bloom filters no other neighbors in common but the node x, a zero rebroadcast delay is selected. This is beneficial since the message gets forwarded very quickly without risking packet collisions at receiving nodes because the rebroadcast regions do not overlap at the neighbors. For the remaining forwarding nodes, a slightly different delay is chosen for rebroadcasting in order to avoid possibilities of packet collisions.

Furthermore, in very dense networks, e.g., in urban scenarios, node x can make the packet dissemination process even more robust by adding more 1-hop neighbors to $\mathbb{R}_x$. Finally, if the cardinality of $\mathbb{R}_x$ is too large and not all 1-hop neighbors can be included, the IDs of these nodes can be replaced by a Bloom filter $\mathcal{R}_x \leftarrow \mathbb{R}_x$ which further contributes to lower the channel utilization and thus increases link quality.

5.5 Performance in VANETs

To evaluate the performance of our proposed 2-hop neighbor maintenance algorithm, as well as the message dissemination using this neighbor information, we use combined network- and road-traffic simulation. Our investigations focus on three main aspects: First we study the impact on beaconing when using Bloom filters by evaluating channel quality and metrics of the neighbor table based on a novel oracle which we developed and will describe in the following sections. Next, we focus on the quality of our neighbor table algorithm, in particular the correctness of the entries compared to the presented oracle. Finally, we investigate the performance of message dissemination in combination with our 2-hop neighbor table algorithm for different Bloom filter sizes and provide a scalability study for different scenarios.

5.5.1 Realistic Road Traffic and Network Simulation Setup

In the following we used the vehicular networking simulation toolkit Veins [122], which couples the SUMO road traffic simulator with the network simulator OMNeT++, as described in detail in Section 2.4. We selected synthetic, but very realistic road traffic, modeled by SUMO in favor of road traffic traces since it allows us to easily control the scenario in terms of traffic density.

As a first setup we configured a six-lane freeway of which the network simulator used 5 km for simulating wireless communication. To avoid border effects we collected protocol performance metrics in a Region of Interest (ROI) of 3 km. We modeled road traffic as a mixture of 90 % cars and 10 % trucks by sampling from a distribution of five different vehicle types. These consisted of two types of trucks and three types of cars modeling a variety of driving styles. Further we used two different road traffic densities of ∼43 veh/km and ∼148 veh/km to model low and

high density traffic, respectively. As a second scenario, we conducted our simulations by using a Manhattan grid with a road traffic density of $\sim$400 veh/km^2 and four types of cars modeling a variety of driving styles and a ROI of 2.1 km^2 as well as buildings within the scenario. We configured a warm-up period in order to fill the roads with vehicles and reach a steady state of neighbor table protocol operations. This was also used to pre-populate 1-hop and 2-hop neighbor tables.

For transmitting beacons, we selected three approaches as a baseline to compare the performance of our Bloom filter based approach. This setup is comparable with the one used in Chapter 4. In detail, we used simple fixed period beaconing (using 1 Hz and 10 Hz), originally defined for sending CAM messages, as well as Transmit Rate Control (TRC) [13] of the ETSI ITS-G5 Decentralized Congestion Control (DCC) standard.

For measuring the performance of our Bloom Hopping protocol, we randomly select every 100 ms 10 vehicles being uniformly distributed within the ROI to disseminate messages. To investigate the performance of the Bloom filter algorithms we performed simulations for different Bloom filter sizes and show a subset of the most interesting results. To acquire statistically significant results, we performed at least 5 runs with different random seeds for simulating road and network traffic for all experiments. Confidence intervals – even they are barely visible as they are quite small – clearly show that the collected measurements are statistically sound. In essence, we summarize the most important simulation parameters in Table 5.1.

5.5.2 Performance Evaluation Using an Oracle

To observe the protocol performance we investigate the following metrics.

In the beginning, we focus on the selected beacon interval as well as the resulting channel utilization in order to assess the impact of the additional neighbor information piggybacked in the beacons. The performance of these metrics is in particular influenced by the size of the Bloom filters. Next, we evaluate the fraction of nodes that we did not remove from the neighbor table (and therefore marked as hidden) as $p_{\text{hidden}} = \frac{|\overline{\mathbb{X}}|}{|\mathbb{X}'|}$. This can be used as an additional metric to observe channel quality. Furthermore we measure the fraction of not acknowledged neighbors (due to non-symmetric channels) of a node as $p_{\text{not\_ackd}} = 1 - \frac{|\mathbb{X}'|}{|\mathbb{X}|}$ to get an indication about how symmetric the communication channel between nodes is. Following up to results published in [25], [28] and also evaluated in Chapter 3 and section 3.5.1, we evaluate the neighbor churn rate. This metric shows the fraction of 1-hop neighbors pruned from the neighbor table per second due to lost beacons or because a neighbor moved outside the communication range. It helps to understand the dynamics of the network and gives an indication about fluctuations in neighbor information.

ETSI ITS-G5 TRC	
Minimum beacon interval	$I_{\text{min}} = 40\,\text{ms}$
Default beacon interval	$I_{\text{def}} = 500\,\text{ms}$
Maximum beacon interval	$I_{\text{max}} = 1\,\text{s}$
b_{min}	0.15
b_{max}	0.40
ATB	
Minimum beacon interval	$I_{\text{min}} = 100\,\text{ms}$
Maximum beacon interval	$I_{\text{max}} = 1\,\text{s}$
Interval weight w_{I}	0.75
Beacon message	
Packet size	200 B + Bloom filter
Bloom filter size	from 12 B to 350 B
MAC priority	AC_BE
	AIFSN = 6
	$\text{CW}_{\text{min}} = 15$, $\text{CW}_{\text{max}} = 1023$
Multi-hop message	
Packet size	300 B
MAC priority	AC_VO
	AIFSN = 2
	$\text{CW}_{\text{min}} = 3$, $\text{CW}_{\text{max}} = 7$
IEEE 802.11p PHY	
NIC TX power	20 mW
NIC bitrate	6 Mbit/s
Path loss model	freespace ($\alpha = 2.0$)
building obstacle shadowing	$\beta = 9\,\text{dB}$, $\gamma = 0.4\,\text{dB/m}$

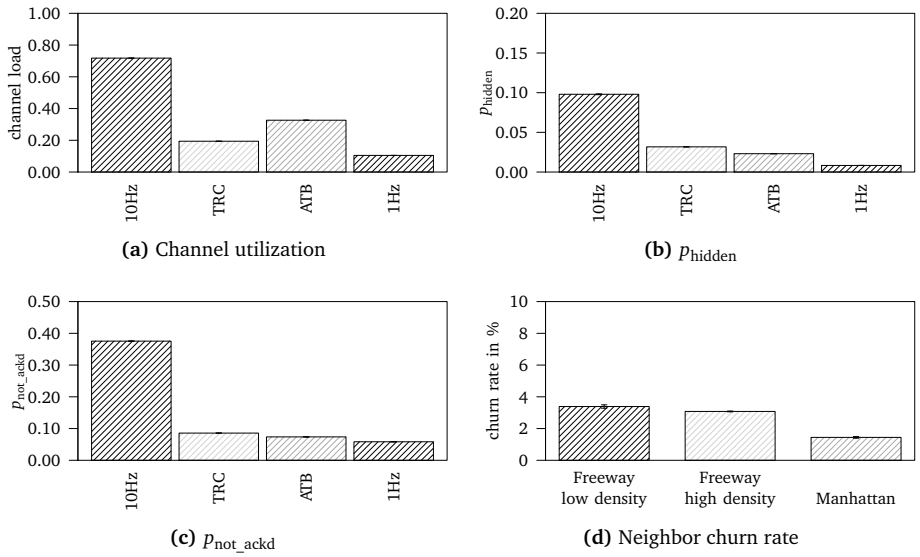
Table 5.1 – Vehicular network simulation parameters; based on [32] © 2018 IEEE.

As a next step, to assess the quality of neighbor tables, i.e., the up-to-dateness of 1-hop and 2-hop entries of each node, we compare the results to an oracle. Instead of using a simplistic oracle based on a unit disk model employing a fixed communication range (as used in Chapter 4 and section 4.5), we developed a more sophisticated method by taking advantage of an idealistic MAC and PHY. This model ignores frame collisions and delays caused by CSMA/CA and Enhanced Distributed Channel Access (EDCA) queues, and thus represents an idealistic and always unutilized communication channel for each node by still employing realistic frame reception models. A collision is defined to be a frame that could have been correctly decoded if there would not have been any interference. Thus, in our case for the oracle we still can decode the frame when it normally would have been dropped due to bit errors. We take advantage of a 10 Hz beaconing scheme to populate our oracle with

neighbors and create a database for each simulation run and repetition to contain the 1-hop neighbors $\mathbb{O}'$, and 2-hop neighbors $\mathbb{O}''$ for each vehicle over time. On average, each vehicle has 44 one-hop and 41 two-hop neighbors (freeway low density); 169 one-hop and 159 two-hop neighbors (freeway high density); and 54 one-hop and 131 two-hop neighbors (Manhattan) in our simulation setup.

We measure the fraction of missing 1-hop neighbors of a node x compared to the oracle as $p'_{\text{missing}} = \frac{|\mathbb{O}' \setminus \mathbb{X}'|}{|\mathbb{O}'|}$. Moreover we measure the fraction of outdated 1-hop neighbors of a node x as the relative amount of additional neighbors compared to the oracle as $p'_{\text{outdated}} = \frac{|\mathbb{X}' \setminus \mathbb{O}'|}{|\mathbb{X}'|}$. Similar metrics were recorded for 2-hop neighbors as p''_{missing} and p''_{outdated}. These metrics are collected every 100 ms after the warm-up period.

For multi-hop message dissemination employing our Bloom hopping protocol, both the fraction of informed nodes and the channel utilization were recorded. We compare our results to a naïve approach maintaining neighbor tables in which instead of a Bloom filter the neighbor table entries are appended as a plain list consisting of 6 B per entry. For all results, we plot the mean together with the 95 % confidence interval.



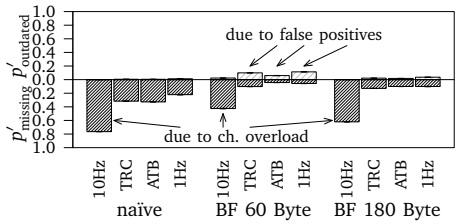
(a) Channel utilization

(b) p_{hidden}

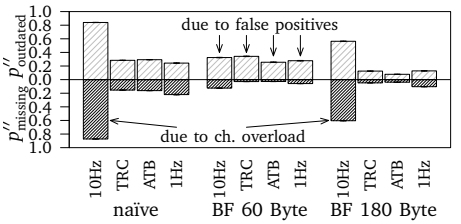
(c) $p_{\text{not\_ackd}}$

(d) Neighbor churn rate

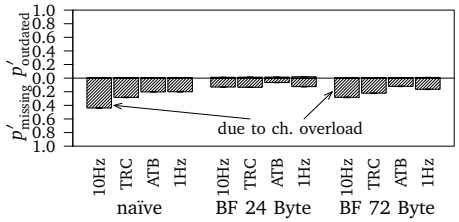
Figure 5.3 – Performance for beaconing using the high density freeway scenario as well as Neighbor churn rate for different scenarios and traffic densities (average value with 95 % confidence interval); based on [32] © 2018 IEEE.



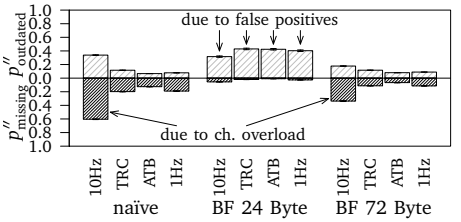
(a) Freeway: Fraction of missing and outdated 1-hop neighbors.



(b) Freeway: Fraction of missing and outdated 2-hop neighbors.



(c) Manhattan: Fraction of missing and outdated 1-hop neighbors.



(d) Manhattan: Fraction of missing and outdated 2-hop neighbors.

Figure 5.4 – Neighbor ratios (1-hop and 2-hop) for the high density freeway and Manhattan scenario: Naïve solution compared to our approach using Bloom filters with different sizes (average value and 95 % confidence interval); based on [32] © 2018 IEEE.

5.5.3 Impact on Beaconing

At first, we investigate the impact of the Bloom filter approach on the beaconing performance using a fixed size Bloom filter of 240 B. In Figure 5.3a we show the channel utilization for the high density freeway scenario. As we see, both TRC and Adaptive Traffic Beacon (ATB) are rather sensitive to the channel utilization, and configure the beacon interval to higher values in the high density scenario, although ATB is increasing the interval less aggressive (resulting in lower delays), which we already investigated in Chapter 4 and section 4.5. These results are in accordance with recent studies on beacon protocols and helped us to validate both the setup and the beacon protocol implementation [66], [72]. We can clearly see that for 10 Hz beaconing it strongly overloads the wireless channel in the high traffic density scenario. For the Manhattan scenario similar qualitative results can be observed, but in general with different beacon intervals and channel utilization due to obstacle shadowing. In particular ATB choses a slightly larger beacon interval in comparison to TRC to cope with the network dynamic caused by obstacle shadowing and hidden terminals.

Our expectation is that neighbor information is being less up-to-date for high vehicle densities; particularly for TRC and even more critical for 10 Hz beaconing.

To confirm this expectation we evaluate the metrics p_{hidden} and $p_{\text{not\_ackd}}$, which focus on measuring the channel conditions in Figures 5.3b and 5.3c, again using a Bloom filter size of 240 B and the high density freeway scenario. As we see, the performance of 10 Hz beaconing in the high density scenario is poor in comparison to all other beaconing protocols. The main cause for this is the highly overloaded channel leading to unstable neighbor tables. Also TRC has a worse performance compared to ATB and 1 Hz beaconing; this is mainly because the protocol state machine for TRC has rather large steps between the available beacon intervals which means that a neighbor might be dropped due to a quick change in the beacon configuration. However, ATB does not suffer that much from this problem, thus, p_{hidden} and $p_{\text{not\_ackd}}$ stay at a lower value. This, of course, also holds for 1 Hz beaconing. In essence, these metrics are rather sensitive to the channel load. With lower channel load, TRC performs much better (e.g., for the low density scenario), however TRC suffers from a higher number of hidden neighbors due to the inherent design of the protocol state machine. For the Manhattan scenario we observe similar effects: both protocols, TRC and 10 Hz beaconing suffer from a higher value of p_{hidden} in comparison to ATB and 1 Hz beaconing. ATB keeps the values lower as it adjusts the beacon interval according to the channel state in a careful way. As can be seen, $p_{\text{not\_ackd}}$ is even more sensitive to channel congestion as well as to oscillating beacon frequencies in comparison to the metric p_{hidden}.

Finally we assess the number of deleted neighbors per second. To do so we show the results for the low and the high density freeway scenario as well as the Manhattan scenario in Figure 5.3d using 1 Hz beaconing by using a fixed size Bloom filter of 240 B. As a result, we observe around 3 % and 3.3 % deleted neighbors per second for the low and high density freeway scenario, respectively.

For the Manhattan scenario, the value drops to around 1.5 % deleted neighbors per second, which is mainly caused by the slower driving speed of the vehicles compared to the Freeway scenario.

5.5.4 Bloom Filter Based Neighbor Table Management

As a next step we concentrate on the performance of the Bloom filter based approach for neighbor management. Here we want to assess the quality of the maintained neighbor tables for different Bloom filter sizes and beaconing protocols. Thus we start studying the capabilities of the beacon protocols to maintain the neighbor tables, by presenting the results of outdated and missing entries compared against our developed oracle. As already explained in the beginning, we compare our results with a naïve baseline neighbor management protocol that exchanges 2-hop neighbor information using a list of identifiers without the usage of Bloom filters. The beacon size of this baseline approach grows linearly with the amount of neighbors included

in each beacon. In the following we call this method *naïve*. For the Bloom filter experiments, we use two different filter sizes, one being too small, thus, suffering from false positives, and a fitting (for the particular scenario) Bloom filter size.

Results for the high density freeway and the Manhattan scenario are shown in Figure 5.4. An intuitive expectation would be to have larger and more accurate neighbor tables the more frequently we exchange neighbor information, i.e., the smaller the beacon interval gets. However, as can be seen in the results, this hypothesis cannot be confirmed. For high density scenarios, the load on the channel is getting so high such that no continuous update is possible anymore, which is caused by collisions on the wireless channel. This observation is in line with findings about beaconing in the literature, e.g., in [72].

In general, the outdated and missing fraction gives a good indication about how well the protocols are able to maintain the neighbor tables in comparison to the oracle. For too small Bloom filters, the outdated ratio increases because the Bloom filter generates too many false positives. This not only holds for 2-hop neighbors, but also has a negative impact on the fraction of outdated 1-hop neighbors (cf. Equation (5.6)). On the other hand, a too big Bloom filter directly contributes to an increasing channel load (and, thus, the collision probability). Thus, a tradeoff has to be found between the Bloom filter false positive rate and its size.

When looking at the fraction of outdated 1-hop neighbors in Figures 5.4a and 5.4c) we observe a quite low value for all protocols. This is mainly caused by entries which are removed from the list when there was no beacon received up until the next announced interval, and no further away node with lower or equal channel utilization includes this node in the Bloom filter. Particularly, for too small Bloom filters, false positive member tests cause a node to not delete 1-hop neighbors, leading to a higher amount of outdated neighbors. Similar results can be observed in the low density freeway scenario.

When we investigate the fraction of missing entries, however, we observe higher values for high vehicle densities, which holds particularly for the naïve approach. In the high density freeway scenario, for the 60 B Bloom filter size we experiences a lower missing 1-hop ratio (compared to the 180 B version) due to a smaller beacon size. This is due to more packet collisions on the wireless channel, thus, missing updates. Having a closer look at 1 Hz beaconing in the high density freeway scenario, the number of missing 1-hop neighbors is lowered by 54 % in comparison to the naïve approach, when a Bloom filter size of 180 B is used. In the Manhattan scenario, the results for 1 Hz beaconing lead to a decrease of 18 % for the missing 1-hop neighbors for a Bloom filter size of 72 B.

Results for 2-hop neighbor information show a similar trend (cf. Figures 5.4b and 5.4d), however the ratio of outdated neighbors is higher since dissemination time accumulates over 2 hops. Further, a too small Bloom filter size increases the amount

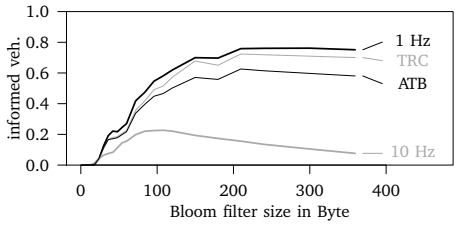
of outdated information, caused by false positive results when querying the Bloom filters. When channel load gets high, the impact gets even more visible: the naïve approach for 10 Hz beaconing completely fails due to the congested channel. For a communication channel which is completely overloaded (naïve 10 Hz beaconing), even our approach to mitigate the wrong assignment of 1-hop neighbors as 2-hop neighbors fails. In particular, the high value of p''_{outdated} is caused by 1-hop neighbors reporting nodes which we wrongly treat as 2-hop neighbors, however normally could reach within 1-hop when no permanent channel congestion occurs. When using 1 Hz beaconing, we measure a decrease of 53 % of missing and a decrease of 47 % of outdated 2-hop neighbors for a Bloom filter size of 180 B compared to the naïve approach. For the Manhattan scenario, the results for 1 Hz beaconing lead to a decrease of 40 % for the missing 2-hop neighbors when using a Bloom filter size of 72 B. The fraction of outdated neighbors caused by false positives slightly increases by 14 % compared to the naïve approach.

To summarize, in all scenarios we clearly see the potential of using Bloom filters for neighbor table management. Using a fitting Bloom filter size of 60 B (low density freeway), 180 B (high density freeway), 72 B (Manhattan), the amount of missing and outdated information stays at very small values for ATB, TRC, and 1 Hz beaconing.

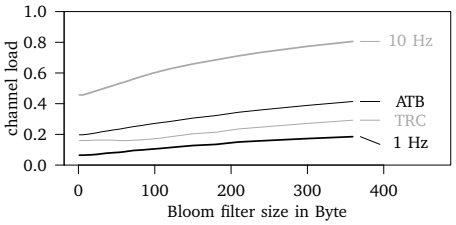
5.5.5 Bloom Hopping Performance

As a next step we study the performance of our Bloom Hopping algorithm when our Bloom filter based 2-hop neighbor table approach is used. To assess the performance we record the fraction of 2-hop nodes that receive the message and monitor the observed wireless channel utilization at the same time. We provide a parameter study for different Bloom filter sizes m and plot the results for all underlying beacon protocols which provide the needed neighbor information. We compare the performance of our Bloom Hopping protocol against the naïve approach in which we compute our forwarding set similar to Section 5.4, however without the use of Bloom filters.

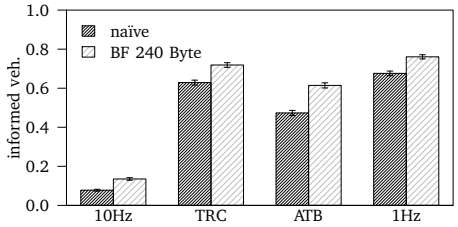
First, in Figures 5.5 and 5.6 we present the results for the high density freeway and the Manhattan grid scenario, respectively. As it can be seen, Bloom Hopping works best using 1 Hz beaconing in which it reaches about 80 % of all 2-hop nodes in all scenarios (cf. Figures 5.5a and 5.6a). The remaining and missing 20 % are caused by the fact of an increased load on the wireless channel (cf. Figures 5.5b and 5.6b). Due to that the communication distance is lowered because of interference on the wireless channel. In reality these nodes have no chance receiving the message. Similar results can be observed in the low density freeway scenario.



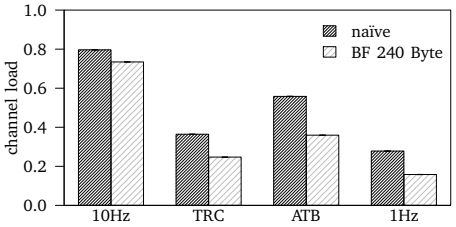
(a) Fraction of informed 2-hop neighbors vs. Bloom filter size.



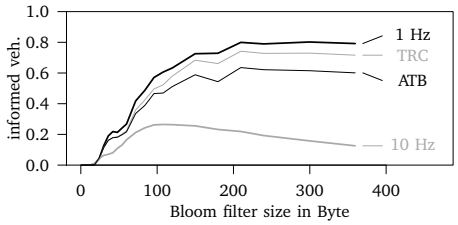
(b) Wireless channel utilization vs. Bloom filter size.



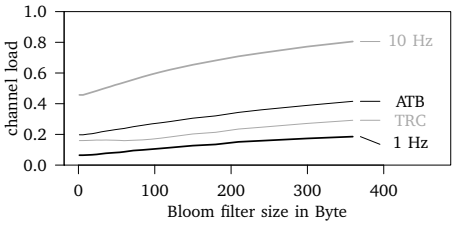
(c) Fraction of informed 2-hop neighbors using a fitting Bloom filter size.



(d) Wireless channel utilization using a fitting Bloom filter size.



(e) Fraction of informed 2-hop neighbors vs. Bloom filter size incorporating non-symmetric nodes.

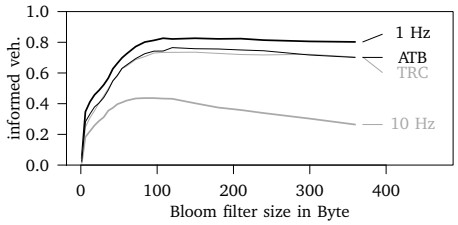


(f) Wireless channel utilization vs. Bloom filter size incorporating non-symmetric nodes.

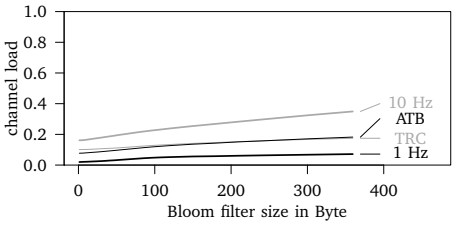
Figure 5.5 – Fraction of informed 2-hop neighbors and channel utilization for the high density freeway scenario using different Bloom filter sizes; based on [32] © 2018 IEEE.

When looking at the high density freeway scenario, we can further observe that when using Bloom Hopping and 1 Hz beaconing we reach around 12 % more 2-hop neighbors (cf. Figure 5.5c) in comparison to the naïve approach. However, at the same time we save around 43 % channel load (cf. Figure 5.5d) compared to the naïve approach. Results for the Manhattan scenario show a similar behavior (cf. Figures 5.6c and 5.6d).

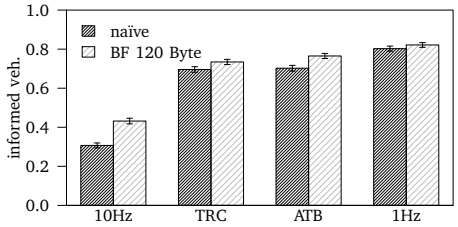
When changing to beacon protocols with adaptive intervals, like TRC or ATB, we notice a slight performance degradation in the high density freeway scenario. This is mainly caused by the increasing channel load, which leads to outdated or inaccurate neighbor information (cf. Section 5.5.4). The worst performance is achieved by 10 Hz beaconing, which simply overloads the wireless channel in all scenarios. As we observe in Figures 5.5a and 5.6a, the Bloom filter size has a great impact on



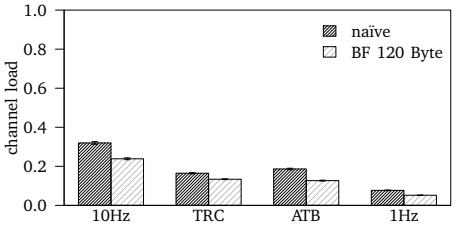
(a) Fraction of informed 2-hop neighbors vs. Bloom filter size.



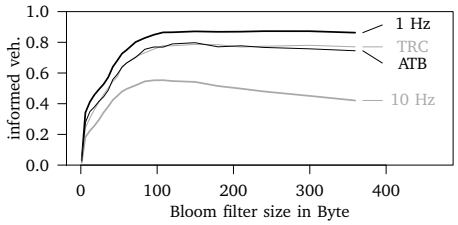
(b) Wireless channel utilization vs. Bloom filter size.



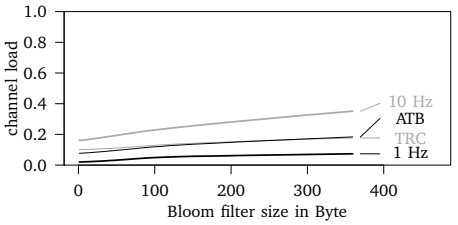
(c) Fraction of informed 2-hop neighbors using a fitting Bloom filter size.



(d) Wireless channel utilization using a fitting Bloom filter size.



(e) Fraction of informed 2-hop neighbors vs. Bloom filter size incorporating non-symmetric nodes.



(f) Wireless channel utilization vs. Bloom filter size incorporating non-symmetric nodes.

Figure 5.6 – Fraction of informed 2-hop neighbors and channel utilization for the Manhattan scenario using different Bloom filter sizes; based on [32] © 2018 IEEE.

both, the fraction of received nodes as well as the channel utilization. The amount of received neighbors increases at the beginning with increasing size of Bloom filter, thus decreasing false positives. After reaching a local maximum at around 240 B (high density freeway) and 120 B (Manhattan) the performance degrades due to increased channel load which is caused by the increasing size of the Bloom filter.

As a next step we investigate the impact of asynchronous wireless channels to the message dissemination algorithm. This way we modify the Bloom Hopping algorithm (outlined in Equation (5.10)) to include also non-symmetric nodes $\mathbb{X}$, instead of only symmetric nodes $\mathbb{X}'$ as calculated in Equation (5.1). Results show very similar values for the fraction of informed vehicles and the channel load as shown in Figures 5.5e and 5.5f for the freeway scenario. A non-symmetric communication channel occurs when node x can receive messages from node y but not vice versa,

e.g., due to interference. An intuitive expectation would be a much lower rate of informed vehicles when nodes with an asynchronous wireless link would be selected to retransmit a particular message but cannot receive the message due to interference. However, as we see in the results, this expectation cannot be confirmed. The Bloom Hopping algorithm solves the issue of non-symmetric communication channels as only those neighbors get selected to rebroadcast messages that gain additional uncovered 2-hop neighbors. A node that suffers from high interference and thus observes a lower amount of decodable messages has a limited overview of neighboring nodes and, therefore, announces only a lower amount of direct 1-hop neighbors within its Bloom Filter. Consequently, a node which announces a higher amount of neighbors can gain more towards coverage of all 2-hop neighbors and, thus, gets chosen in favor to a node only having a very small neighbor set. A very small improvement to reach further nodes is gained only in very rare cases for nodes with a non-symmetric wireless link, which can be observed in the Manhattan scenario shown in Figures 5.6e and 5.6f.

When comparing our results to other Bloom filter based message dissemination approaches like [78], we clearly see following advantage of our system: The lowered channel utilization gained by transmitting a much smaller Bloom filter instead of a large list of neighbors leads to an increased number of informed vehicles when we use our Bloom Hopping message dissemination protocol (as outlined in Figures 5.5 and 5.6). These results are an advantage over existing works (e.g., [78]), where only the overhead of beacon transmissions (i.e., the channel utilization) is lowered, but no better performance of the application could be achieved. Among this, as outlined in Section 5.3.3, the intersection operation on Bloom filters causes loss of information and, thus, increases the false positive probability of a Bloom filter and is therefore not recommended as we show in Figure 5.2. Yet, using the no longer recommended 10 Hz beaconing, which leads to a complete congestion of the wireless channel, shows an astonishing performance improvement when using our Bloom filter based neighbor management.

We conclude that our Bloom Hopping approach increases the fraction of informed nodes and at the same time lowers the channel utilization.

5.6 Lessons Learned

In this chapter we presented a novel probabilistic 2-hop neighbor management approach using Bloom filter for application in dynamic wireless networks. Compared to alternative solutions using a plain list of neighbor identities, the use of Bloom filters provides best scalability of the system that comes at the cost of a small false positive rate. We analytically explored the properties of Bloom filters for applications

in the Vehicular Ad Hoc Network (VANET) domain, and determined best suited Bloom filter sizes to keep this false positive rate very low. These findings help to prevent overloading the wireless channel. Further we explored the capabilities of our solution to build a fundamental basis for higher layer protocols, for which our designed multi-hop broadcast protocol *Bloom Hopping* builds the foundation by efficiently selecting forwarders according to the encoded Bloom filter information provided by the neighbor table algorithm.

To evaluate the performance of the developed Bloom filter based neighbor management, as well as the Bloom Hopping protocol, we provide results of an extensive simulation study. For the 1-hop and 2-hop neighbor management we compared our Bloom filter based solution with a novel oracle. To achieve this, we carefully explored theoretical reachability in the wireless network ignoring interference and delays at the MAC protocol but using a realistic path loss model. Results clearly show that the difference to the oracle is very small for the case when the Bloom filter size has been adequately chosen for the application scenario. Moreover, we demonstrated that our Bloom filter approach can easily be tied to typical beacon protocols, given that they are able to provide simple congestion control making our protocol able to be implemented in future developments of congestion aware beaconing protocols. A further finding of our simulation study is that higher beaconing rates not necessarily lead to better and more accurate neighbor information, which might be considered counter-intuitive. A fundamental limit on the performance was confirmed to be the wireless channel load. Our Bloom Hopping protocol helps to improve information dissemination and decrease channel utilization significantly by using space efficient Bloom filters.

Although we investigated our approach in a very specific application domain, namely VANETs, the concept can be applied also to other networking scenarios exhibiting a very dynamic topology.

Chapter 6

Multi-Channel Beaconing

In the previous chapters we focused on potential performance issues when IEEE 802.11 unicast communication is used, as well as the development of a purely broadcast based holistic network layer for vehicular communication, and the design of a probabilistic message dissemination protocol to cover 2-hop neighbors.

In the following chapter we go back to Media Access Control (MAC) layer functionality and focus on the scalability of network communication when taking advantage of multiple channels, which is crucial to support a large amount of nodes in a network. In particular, the IEEE 1609.4 Wireless Access in the Vehicular Environment (WAVE) standard provides multi-channel operation for vehicular communication using IEEE 802.11p in single-radio environments. Here time is divided into two phases, namely Control Channel (CCH) phase and Service Channel (SCH) phase, each having a length of 50 ms and synchronized via Global Positioning System (GPS). However, three fundamental questions are still open to provide full functionality of multi-channel operation: *when* to use *which* channel for transmission, and how to properly select a fitting channel for reception of important information. This chapter focuses on these questions and provides algorithms to allow efficient multi-channel operation in single-radio networks in the vehicular communication context. Results show that our multi-channel approach outperforms single-channel protocols in terms of channel utilization and message dissemination performance.

The content of the following chapter is based on the following peer-reviewed publications:

- F. Klingler, F. Dressler, J. Cao, and C. Sommer, "MCB - A Multi-Channel Beaconing Protocol," *Elsevier Ad Hoc Networks*, vol. 36, no. 1, pp. 258–269, Jan. 2016, © 2016 Elsevier B.V.
  My contribution in this journal article was the design, implementation and evaluation of the multi-channel message dissemination algorithm.

- F. Klingler, F. Dressler, J. Cao, and C. Sommer, "Use Both Lanes: Multi-Channel Beaconing for Message Dissemination in Vehicular Networks," in *10th IEEE/IFIP Conference on Wireless On demand Network Systems and Services (WONS 2013)*, Banff, Canada: IEEE, Mar. 2013, pp. 162–169, © 2013 IEEE.
  My contribution in this conference publication was to study the feasibility of multi-channel beaconing in vehicular networks based on IEEE 1609.4 split-phase channel switching.

- F. Klingler, "Improving Multi-Channel Beaconing in Vehicular Networks," in *3rd GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*, Ulm, Germany, Mar. 2015.
  My contribution in this Regional Workshop paper was the design and evaluation of first results of a multi-channel beaconing approach in comparison to a single-channel protocol.

6.1 Motivation

Applications in the domain of Intelligent Transportation Systems (ITS) rely on efficient communication concepts [158], for which much progress has been made in the last few years. Developments in this area lead to industry standards which define both the physical and access layer for Inter-Vehicle Communication (IVC), see Section 2.2 for details. Prominent protocol stacks here are IEEE 802.11p and IEEE 1609.4 [51], [90] or ETSI ITS-G5 [13].

Moreover, different information dissemination concepts have been explored based on these radio access technologies, which started with simple messages broadcast by individual vehicles in a periodic fashion. These one-hop broadcasts have been named *beacons*, and were later standardized as Cooperative Awareness Messages (CAMs) [87] in Europe, and Basic Safety Messages (BSMs) [159] in the U.S.; for further information please refer to Section 2.2.

To allow transmission of CAMs/BSMs in many different scenarios ranging from sparse scenarios at night time to traffic at rush hours or even traffic jams, the inter-packet interval has been identified as a critical parameter to adjust [13], [66], [72], [160], [161]. Here the main objective is to adapt the beaconing frequency to minimize communication delay, but at the same time keep wireless channel load at a low value to avoid packet collisions. However, the presented approaches rely only on a single wireless channel by making use of either the CCH or one of the available SCHs.

In this chapter, we extend these concepts and study the feasibility of using multiple channels to be used at the same time in a a Single-Radio Multi-Channel (SR-MC) fashion. Based on our investigations we developed a novel multi-channel message dissemination approach, which we call the Multi-Channel Beacon (MCB) protocol. The work we present in this chapter has evolved from our previous work presented in [33], by now employing novel channel scheduling algorithms to carefully determine when to send information within a particular CCH interval. Our new approach substantially reduces load on the channel and thus packet collisions while at the same time increases reliability and application layer performance. Furthermore, MCB is capable to also reduce the inter-beaconing interval as well as the channel utilization in comparison to single-channel approaches. Taking a traffic efficiency application as example, the presented MCB protocol is not limited to this application domain. The developed concepts of multi-channel scheduling can easily be applied to any other protocol requiring information dissemination in vehicular networks. Based on simulation studies, our evaluation shows that the use of multiple channels is not only feasible but also leads to substantial performance improvements.

We summarize our main contributions as follows:

- We present a novel Single-Radio Multi-Channel (SR-MC) beacon scheduling system (MCB) for vehicular networks that follows a split phase approach.
- We develop scheduling algorithms for MCB which can keep channel coordination overhead low by relying on a careful selection of when to send coordination information.
- By taking a typical traffic information system as an example, we clearly show that the use of multiple channels not only reduces the load of the wireless channel(s) but also lowers the information dissemination delay.

6.2 Preliminaries

As we take a traffic efficiency application as example, we classify related work in this topic into two categories, namely Traffic Information System (TIS) protocols of IVC by using beaconing, and multi-channel scheduling systems for both, single-radio and multi-radio environment. Since TIS protocols are already mostly covered in Section 2.2, here we only outline important details in that area relevant for our protocol design.

SOTIS, proposed by Wischhof et al. [85], goes one step beyond simple beaconing of CAMs [87] and BSMs [159]: each vehicle maintains a knowledge base which integrates any received traffic information items. Selected parts of these knowledge bases are periodically broadcast within beacons to neighboring vehicles.

In their investigations the authors found that static periodic beaconing is not suitable for every road traffic scenario. This is mainly caused by the wireless channels which easily get overloaded in case of traffic congestions when many vehicles simultaneously broadcast the information contained in their knowledge bases. On the other hand, when traffic density is very low as in sparse scenarios, the beaconing frequency might be too low to exploit the few communication opportunities. This way it is difficult to disseminate information in a timely manner.

To the best of our knowledge, REACT [65], is the first approach which proposes a dynamic beaconing interval selection. Here the interval between two consecutive transmissions of a vehicle is adapted to the density of the road network.

Consequently, to integrate a novel prioritization scheme, Adaptive Traffic Beacon (ATB) [66] extends this approach, where the inter-packet interval depends on both, the channel quality and the priority of the traffic information being transmitted. The goal of ATB is to send as many information as possible, but avoid overloading the wireless channel at any time, which we describe in detail in Section 2.2.1.

Applications for IVC often derive the utility of information based on sender side metrics which could lead to problems when a receiver is interested in information

which is near irrelevant for a sender. Categorizing and thus transmitting information which is of most interest for receiver is a key challenge in vehicular networks, where FairDD [162] provides an algorithm using Nash Bargaining.

FairAD [67], [86] extends these concepts and successfully combines fair and efficient information dissemination of FairDD with adaptive beaconing of ATB by retaining the advantages of both approaches. However, to this end, the protocols still operate on a single channel only, which leaves room for further improvements, in particular when traffic density is high.

Moving to related work in the context of multi-channel scheduling, problems and pitfalls of wireless communication both for SR-MC and Multi-Radio Multi-Channel (MR-MC) systems have been described in detail by Crichigno, Wu, and Shu [91]. In a more theoretical way the authors in [163] study the complexity of channel scheduling for MR-MC and prove that it is NP-hard under different interference models. An investigation about the capacity of such a network for an increasing number of nodes can be found in [164].

Based on the multi-channel operation of IEEE 1609.4, [165] studies dynamic channel intervals instead of fixed ones as proposed in the standard. In particular, the authors propose to divide the CCH interval into three phases in order to support service announcements, status beacons and peer to peer communication resource reservation. However, in their approach the authors do not provide channel selection algorithms.

Time slot reservation on a SCH and the CCH as well as collision free access for messages in these time slots is proposed by Lu et al. [166] in terms of a multi-channel MAC protocol by using two phases in the CCH interval. Channel negotiation in this approach consists of several frames to be exchanged among each vehicle for each negotiation step which adds additional channel load.

Based on an analytical model, the authors in [167] study dynamic channel intervals. In particular, the CCH interval is split into a safety and a service announcement interval for which the boundaries are dynamic and adapted according to the traffic density. Similar to approaches presented before, they do not focus on the SCH selection scheme.

Moving to asynchronous multi-channel scheduling, the approach in [168] employs a channel negotiation scheme which uses a well-known SCH (instead of the CCH) to announce specific services.

Our presented MCB protocol – in contrast to other multi-channel approaches – follows the beaconing principle, which allows to lower the complexity of channel negotiation and is at no cost to the speed of information dissemination which we demonstrate in terms of simulation studies.

We take advantage of the ATB protocol described in Section 2.2.1. It is the protocol upon which our work builds on. Further we also adopt the split phase approach of

IEEE 1609.4 [17] outlined in Section 2.2.3, however we want to highlight that MCB is not limited to be used with WAVE.

6.3 MCB Protocol

In the following section we give an overview about our designed MCB protocol, which takes advantage of additional SCHs which are available for ITS in the 5.9 GHz band. In particular, MCB is built as a split phase protocol for Single-Radio Multi-Channel (SR-MC) environments, and is able to be operated on top of other protocols using this channel switching scheme, e.g., IEEE 1609 WAVE. Our proposed protocol is an extension to our multi-channel information dissemination protocol presented earlier [33] which was based on the ATB protocol. The main difference introduced by MCB is that completely new algorithms and techniques provide now the possibility to address the shortcomings introduced by split phase multi-channel operation, mainly additional delays and split network topologies.

6.3.1 Multi-Channel Operation

In Figure 6.1 we outline the main operation principle of MCB by taking advantage of a split phase channel switching protocol. Time is divided into equal length CCH and SCH intervals, which are separated by short guard intervals to mitigate synchronization inaccuracies; the same principle that is used by IEEE 1609 WAVE. On the CCH data announcements are broadcast, which advertise a SCH on which the actual data will be transmitted during the following respective SCH interval. Nodes then can tune to the best available SCH to receive potential important information on such a SCH. MCB only performs channel switching during guard intervals to avoid message loss during switching the channel.

In general, the operation of MCB can be divided into four steps: First, the rate at which beacons are transmitted is regulated by adapting the amount of CCH/SCH

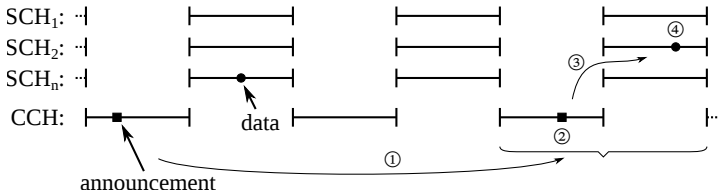


Figure 6.1 – MCB operation in four steps: (1) coarse-grained interval selection, (2) selecting a time for the announcement, (3)selecting a channel and time for the data and sending an announcement, and (4) sending the data; based on [35] © 2016 Elsevier B.V.

intervals elapse between each beacon transmission. Second, whenever a CCH is selected to transmit a beacon, MCB carefully determines when to send a data announcement within the CCH interval. This time t_{MCB} is calculated based on the priority of the payload information in a way such that more important messages are transmitted earlier in the CCH interval. Third, at the same time t_{MCB} represents the time for a transmitting node to select a SCH channel to transmit payload information by taking into account all received announcements from other nodes up until this point. In a fourth step, the node tunes its radio to the announced SCH during the guard interval between CCH and SCH, and calculates the time when to send the data within the SCH interval.

6.3.2 Step 1: Coarse Grained Interval Selection

Protocols which take advantage of a split phase approach like IEEE 1609.4 by using fixed size intervals introduce additional delays for communication on a SCH, since for each actual data transmission the used SCH needs to be announced within the CCH interval. This procedure introduces a delay between generating the information and transmitting it on the channel; for IEEE 1609.4 this delay can be up to 100 ms in the worst case. This can happen if the information is generated at the end of the CCH interval, for which the announcement gets delayed by the 50 ms SCH interval, and the data transmission gets delayed by the 50 ms CCH interval. MCB takes this into consideration by using the interval calculation of ATB (see Equation (2.10)) only to determine in which CCH slot to send a beacon. Here, in contrast to ATB which only is capable to monitor a single wireless channel, MCB takes into account also the load on SCHs by using the mean of the Signal to Noise plus Interference Ratio (SNIR) values of SCHs which the radio is tuned to, and stores these values for upcoming calculations. The time when to broadcast the announcement within the CCH interval is calculated in a second step, which we describe in the following section.

6.3.3 Step 2: When to Send the Announcement

Until now we determined the interval in which we transmit a beacon (announcement); as a next step we need to calculate the time t_{MCB} within the interval to transmit the announcement. In this announcement the SCH is indicated on which data will be sent – therefore t_{MCB} represents also the latest point in time on which this decision can take place. As already described previously, channel switching is only performed during guard intervals (in front of each CCH and selected SCH). Since we are focusing on SR-MC environments, the same SCH will be the one on which data is received and transmitted, thus this requires a trade-off between choosing a good channel for sending and a good one for receiving data. MCB delays this decision

as long as possible such that nodes can make a better selection on which channel to tune to. Intuitively, to evenly utilize all SCHs, a node can pick a random SCH for transmitting information. However, with this approach it could happen that the node then misses important information sent by other nodes on a different SCH.

MCB, on the other hand, chooses this time t_{MCB} based on the beacon priority which is related to the utility p of the most important entry of the knowledge base. The value of p is measured in the interval $[0.0, 1.0]$, where smaller values indicate more important beacons. More formally, t_{MCB} is derived by first choosing

$$t_{\text{p}} = 0.5 \times I_{\text{switching}} \times p + I_{\text{guard}}, \tag{6.1}$$

for which $I_{\text{switching}}$ and I_{guard} represents the respective intervals defined in [17], set to 50 ms and 4 ms, respectively. Therefore, depending on the priority, t_{p} defines the earliest time within the CCH interval at which the announcement might be sent. We choose a value of 0.5 such that messages having the lowest priority are delayed to the second half of the CCH interval. Next, the remaining time left in the CCH slot is derived by

$$t_{\text{left}} = I_{\text{switching}} - t_{\text{p}}. \tag{6.2}$$

To finally select the time when to broadcast the beacon, MCB uniformly distributes this time among the time interval left in the slot, upper bounded by a factor f which leads to

$$t_{\text{MCB}} = t_{\text{p}} + \mathcal{U}(0, f) \times t_{\text{left}}. \tag{6.3}$$

This factor is configured to $f = 0.5$ for messages having the highest importance, and $f = 0.8$ for all other messages. These factors ensure that announcements with the highest priority ($p = 0.0$) are transmitted before any announcements with the lowest priority ($p = 1.0$). For priorities in-between we linearly interpolate. Moreover, these factors also ensure that even for the highest delay enough time is left in the CCH interval to transmit the announcement.

This way, whenever a node selects a CCH to transmit an announcement, it is able with above time selection algorithm to take all announcements into account which are received up until t_{MCB}. Therefore, the node can make a better decision on which SCH the information should be transmitted. By exploiting this approach, MCB substantially reduces the need for explicit coordination messages among different nodes.

6.3.4 Step 3: Selecting and Announcing an SCH

Until now we know how to select a proper channel interval as well as the time within this interval to transmit a beacon. What is still open is to determine which SCH to

announce. Intuitively, the use of channel metrics fits better to decide which SCH to select for transmitting information. Although this approach is quite common, it is quite complicated to achieve. The reasons for this are as follows: First, a low utilized SCH on the sender side does not necessarily mean that this is also the case for the receiver, and thus a selection of this SCH may not be the best selection for a receiver due to the hidden terminal problem. Second, using status messages for channel negotiation in order to get information of the channel states of neighboring nodes leads to additional channel utilization which then lowers the goodput of useful data messages. Further, it is difficult to derive a good measure of the channel utilization within a very small time frame, like an SCH. As MCB is designed as a single-radio multi-channel protocol, the channel utilization can only be measured on the currently tuned channel. Thus, it is required to rely on relatively old measurements of the channel utilization, namely the last time the radio was tuned to a channel in question. However, for highly dynamic network topologies, this is not very useful, since the channel utilization can change very frequently. For MCB we choose the following channel selection algorithm for which we experienced the best results:

Each node which wants to send data in the next SCH interval needs to broadcast an announcement in the preceding CCH for it. Thus, it needs to select which channel to tune to no later than at time t_{MCB}. As we show in Figure 6.2, a consequence of this is that only announcements received up until that point can be considered for SCH selection. In essence, all announcements received after t_{MCB} are "lost" to the decision process, since a transmitting node needs to stick with the selected (and announced) SCH. Similarly, all data transmitted on a SCH other than the selected one is "lost" as far as the node which wants to transmit a beacon is concerned. Only nodes which do not intend to transmit a beacon in the next SCH interval can postpone the decision on which SCH to tune to the beginning of the guard interval of the SCH. Whenever the time for this decision (either at t_{MCB} or when the guard interval of the SCH starts) approaches, the node derives the set of all received announcements during

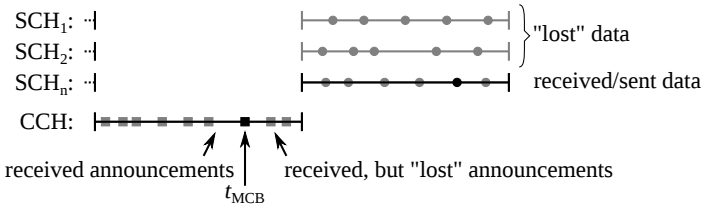


Figure 6.2 – A node selects the best fitting SCH when an announcement is sent at t_{MCB}. Announcements received after this time are "lost" to the decision process. Similarly, data sent on other channels than the selected SCH is "lost" to the node; based on [35] © 2016 Elsevier B.V.

the CCH up until this time leading to

$$\mathbb{W} = \left\{ \begin{pmatrix} \text{channel}_1 \\ \text{priority}_1 \end{pmatrix}, \begin{pmatrix} \text{channel}_2 \\ \text{priority}_2 \end{pmatrix}, \ldots, \begin{pmatrix} \text{channel}_n \\ \text{priority}_n \end{pmatrix} \right\} \tag{6.4}$$

and derives the subset $\overline{\mathbb{W}}$ as those entries having the highest priority (i.e., the lowest value of p) in $\mathbb{W}$:

$$\overline{\mathbb{W}} = \left\{ w \in \mathbb{W} : \mathrm{p}(w) = p_{\min} \right\}, \tag{6.5}$$

$$p_{\min} = \min_{v \in \mathbb{W}} \big(\mathrm{p}(v) \big), \tag{6.6}$$

for which $\mathrm{p}(\cdot)$ returns the priority of an individual announcement.

If MCB does not send a beacon, it chooses the channel that has been announced most often in $\overline{\mathbb{W}}$.

In the case a node intends to send a beacon (with priority p_{self}), it first calculates a set of announcements of potential SCHs as

$$\mathbb{C} = \begin{cases} \overline{\mathbb{W}} & \text{if } p_{\min} < p_{\text{self}} \\ \emptyset & \text{else .} \end{cases} \tag{6.7}$$

This guarantees that, although a node with less important information is still allowed to transmit, it will not only do so much less often (according to the procedure of Section 6.3.2), but it also cannot tune to a SCH channel which would cause it to lose the more important information.

The SCH c which the node announces (and tunes to in the SCH interval afterwards) is randomly drawn from $\mathbb{C}$ as

$$c = \begin{cases} \mathrm{ch}(\mathcal{U}(\mathbb{C})) & \text{if } \mathbb{C} \neq \emptyset \\ \mathcal{U}\big(\{ \mathrm{SCH}_0, \mathrm{SCH}_1, \ldots, \mathrm{SCH}_{\max} \} \big), & \text{if } \mathbb{C} = \emptyset \,, \end{cases} \tag{6.8}$$

where $\mathrm{ch}(\cdot)$ returns the announced channel of the selected announcement and $\mathcal{U}(\cdot)$ selects a set member according to a uniform probability distribution.

6.3.5 Step 4: Broadcasting the Data

Until now, the announcement for a particular SCH has been transmitted, and the radio was tuned to the announced SCH. What is still missing is the selection when to transmit the data in the SCH interval. This time is uniformly distributed among the whole SCH interval to lower the risk of packet collisions (interference) with other nodes intending to transmit data. Similar to what is transmitted in messages by the

original ATB protocol, our message contains a subset of the node's knowledge base, by taking as many entries as there is space for in a single frame.

During the SCH, all successfully received data can be integrated into the local knowledge base.

6.4 Performance Evaluation

We investigate in this section the performance of our multi-channel beaconing protocol MCB in comparison to single-channel alternatives. For this comparison, we introduce the baseline protocols, the simulation scenarios, as well as the configured simulation parameters.

For the simulation studies, we take advantage of the Veins simulation framework [122], which we outlined earlier in this thesis in Section 2.4.

6.4.1 Baseline Protocols

To assess the performance of our multi-channel protocol MCB, we focus on reliability, channel load and performance. As a baseline for single-channel beaconing protocols, we compare MCB with ATB (introduced in Section 2.2.1). As an alternative, we also investigate how well a beaconing protocol performs if the inter-beacon interval is modeled via a state machine based on the observed channel load, similar to what ETSI ITS-G5 Transmit Rate Control (TRC) proposes, and what we describe in detail in Section 2.2.2.

To come up with an comparison to a baseline multi-channel protocol, we implemented Random Channel Selector (RCS) which represents a simple message dissemination protocol randomly choosing a SCH to transmit payload information as well as randomly choosing a time within the CCH to announce information. Further, the selection on which SCH to tune to is also randomly chosen, and the calculation of message utility disabled.

6.4.2 Simulation Scenarios and Parameters

For our simulations we prepared a freeway scenario having a length of 2 km with two lanes in each direction. Two traffic flows, each configured to start at one end of the freeway, meet in between. We used two different vehicle densities for our simulations; a medium utilized freeway and a jam scenario for which the freeway is completely filled with vehicles. The configured vehicles (90 % cars, 10 % trucks) have randomly distributed dimensions and use a mobility model according to the SUMO standard *Krauss* driver model.

Parameter	MCB	ATB	TRC
min. beacon Interval I_{min}	100 ms		40 ms
def. beacon Interval I_{def}	-		500 ms
max. beacon Interval I_{max}	1 s		1 s
channel weighting w_{C}	2		-
interval weighting w_{I}	0.75		-
b_{min}	-		0.15
b_{max}	-		0.40
header length	88 bit		
knowledge base entry size	64 B		
max packet size	512 B		
dummy msg. interval	500 ms		
NIC TX power	20 mW		
NIC bitrate	18 Mbit/s		
path loss model	*freespace*, $\alpha = 2.0$		
# of used channels	1+4	1	1
freeway length	2 km		
traffic density (medium)	58 veh/km		
traffic density (congestion)	185 veh/km		

Table 6.1 – Overview of simulation parameters; based on [35] © 2016 Elsevier B.V.

Each protocol under test performs knowledge base management as detailed by the original ATB protocol outlined in Section 2.2.1.1. Moreover, each vehicle generates low priority dummy messages to model background traffic. In particular, we used this mechanism in a way such that the perceived utility of this background traffic is always lower than that of the messages we trace through the network. The utility of the messages does also not decrease over time. This change is necessary to keep the protocols comparable with each other. Otherwise outdated traffic information items get removed from the knowledge base which could falsify results.

For each simulation run, after protocol execution has reached a steady state, we select one vehicle in the middle of the freeway to start disseminating a high priority message. Moreover, each simulation is repeatedly executed with different random number generator seeds for both, protocol operation and vehicle mobility. We conduct 60 independent simulation runs to get statistically significant results. To minimize border effect, we only record data within a region of interest of 1 km. We summarize all relevant simulation parameters in Table 6.1.

6.5 Results and Discussion

To evaluate our multi-channel beaconing protocol, we focus on four metrics: First, as low level performance indicators, we investigate the mean channel utilization and, second, the packet success rate to get an overview about the sensitivity and efficiency of the protocol under consideration. As a further metric we investigate the

mean beaconing interval of the used dissemination protocol, to get more insights in the protocol behavior. Finally, we assess relative message dissemination speed to evaluate the performance on an application level.

6.5.1 Channel Utilization

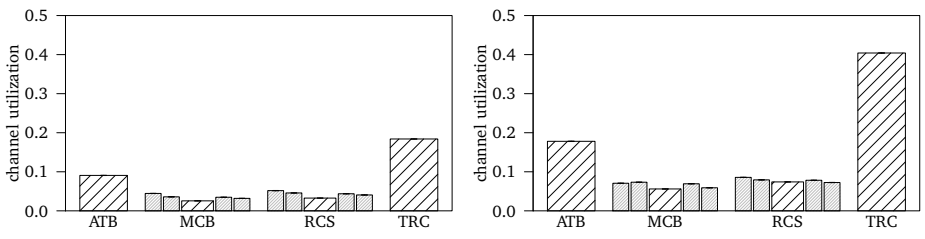
First we have a look at the channel conditions, for which the channel load experienced by each vehicle is taken into consideration. To derive the channel busy ratio, we calculate this metric as the fraction of simulation time for which the physical Clear Channel Assessment (CCA) of that vehicle would have considered the channel busy.

In Figures 6.3a and 6.3b we show the results for the medium utilized and congested freeway scenarios. We present our results in form of bar plots, in which we plot the mean channel utilization for each scenario along with 95 % confidence intervals. Further, for multi-channel protocols we show the utilization separated by channel (CCH in the center, surrounded by SCHs one to four).

Results for the congested scenario are very much comparable to those obtained for medium vehicle density, aside from an increase of channel utilization for all protocols. The results also indicate that all protocols are successful in adapting each vehicle's channel utilization to the total available capacity.

Both protocols, MCB and RCS, distribute their payload transmissions quite evenly across all SCHs. Moreover, they also keep channel utilization very low, following their aim of not overloading the channel. For the single-channel protocol ATB the channel utilization is higher, as it needs to transmit all data on a single channel. However, at the same time, it can use the CCH for 100 % of the time, which means it is more than doubling its available channel capacity compared to channel-switching protocols. This leads to a net channel load that is roughly comparable.

Even TRC shows increased channel use, but still at very reasonable values: The protocol is able to keep channel utilization below $b_{\max} = 0.40$ – even in the jam scenario.



(a) Medium traffic density **(b)** Very high traffic density (traffic jam)

Figure 6.3 – Channel utilization; based on [35] © 2016 Elsevier B.V.

When we investigate the packet success rate, the impact of these different ways of utilizing the wireless channel gets visible even more.

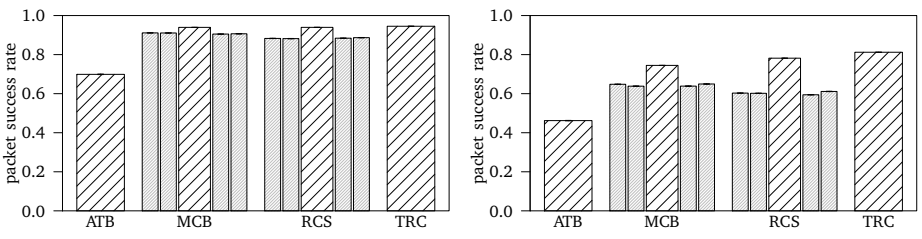
6.5.2 Packet Success Rate

To obtain the packet success rate for broadcast transmission of frames, for each frame we calculate two different reception probabilities: once using the exact SNIR value of the channel, and once ignoring interference and thus only using the Signal to Noise Ratio (SNR) value. Each frame that clears both hurdles is counted as successfully received, and any frame that only clears the second one is counted as a collision. In other words: If the frame would have been successfully received if there would not have been any interference, then we count it as a collision. We divide the first count (collisions) by the sum of both (successfully received, and collisions) to obtain the packet success rate. This gives us an indication of what fraction of packets were lost due to collisions, and consequently, how much of the used channel capacity was (not) wasted.

In Figure 6.4a we first investigate the results obtained for the medium utilized freeway. It appears that the packet success rate of MCB, RCS, and TRC is approximately equal. A slightly less efficient use of the channel makes ATB, the predecessor of MCB. For the congested scenario plotted in Figure 6.4b the same conclusions hold. However it becomes apparent that all protocols' operation is degraded by frame collisions.

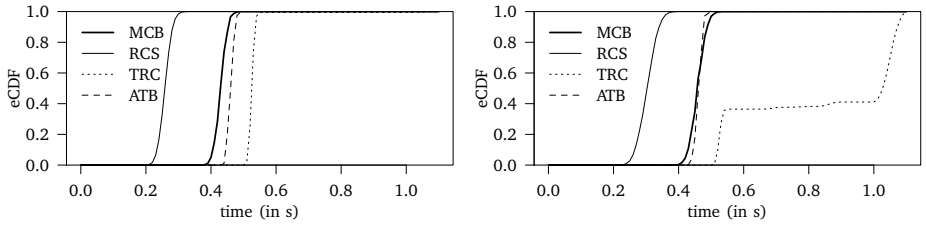
6.5.3 Beacon Interval

Since we have now discusses channel metrics, we now focus on higher layer metrics: A very important aspect here is the distribution of the inter-beacon intervals, which all protocols dynamically adapt. In Figures 6.5a and 6.5b we present the results in the form of an empirical Cumulative Distribution Function (eCDF) for each protocol and for both the medium density and the congested scenario.



(a) Medium traffic density **(b)** Very high traffic density (traffic jam)

Figure 6.4 – Packet success rate; based on [35] © 2016 Elsevier B.V.



(a) Medium traffic density **(b)** Very high traffic density (traffic jam)

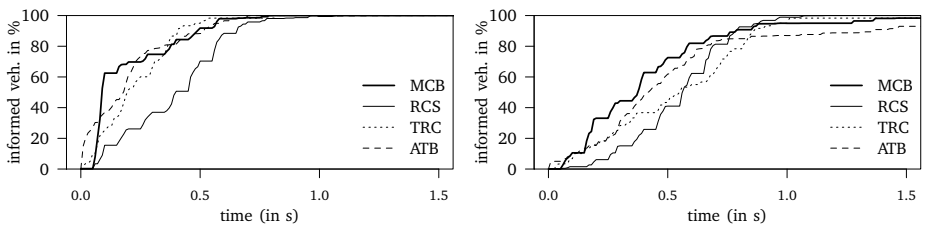
Figure 6.5 – Beacon interval; based on [35] © 2016 Elsevier B.V.

In the congested scenario plotted in Figure 6.5b, we see that TRC chooses either of two beacon intervals. The reason for this can be seen when looking at Figure 6.3b: the channel utilization indicates that the busy ratio b_t must be distributed around the second threshold $b_{\max}$ leading the state machine of TRC to switch between intervals of 500 ms and 1 s.

The ATB and MCB protocols choose very similar beacon intervals in all scenarios, according to their net channel load discussed in Section 6.5.1. Since RCS lacks of smart calculation of the message utility, it always chooses a lower beacon interval – however, with little benefit as we can see in the next section when we investigate an application layer metric.

6.5.4 Fraction of Informed Vehicles

In order to assess the actual application layer performance of the four protocols, we investigate the speed that a single piece of information spreads through the network. To perform this, we generate an information item in the middle of the freeway and insert it into the knowledge base of a single vehicle, as described in Section 6.4.2. Then we track the fraction of vehicles in the network which have received this particular information for each time instant.



(a) Medium traffic density **(b)** Very high traffic density (traffic jam)

Figure 6.6 – Fraction of informed vehicles; based on [35] © 2016 Elsevier B.V.

We show the results for the medium density and the congested scenario in Figures 6.6a and 6.6b, respectively. We plot the mean fraction of informed vehicles in all simulation repetitions for each time step after the start of message dissemination, normalized to $t = 0$.

All the previously discussed factors (channel load, beaconing interval) have an influence to this metric. As can be seen in the plots, the discussed effects lead to a substantially faster information propagation of MCB than either ATB or TRC. This is even more important, since MCB suffers from a short time lag for announcing a beacon and switching channels. Moreover, this benefit of MCB is not only visible in the congested scenario, but it is also very visible in the medium density scenario: Even here, MCB substantially decreases the time for vehicles to be informed of new information, although it suffers from the already mentioned delay to announce information. Other protocols can only catch up in the late phases of information dissemination, when around 80 % of all vehicles have already been informed. The worst performance can be seen with RCS (the baseline multi-channel protocol): although it nearly perfectly manages to distribute channel load among all SCHs (Section 6.5.1), and did not suffer from more packet loss (Section 6.5.2), and as well was transmitting beacons faster (Section 6.5.3), the speed of information propagation lagged behind all other protocols.

6.6 Lessons Learned

In this chapter we presented Multi-Channel Beacon (MCB), which represents a novel beaconing protocol for a Single-Radio Multi-Channel (SR-MC) system that makes use of the multiple SCHs available for Intelligent Transportation Systems (ITS) in order to use all available network capacity. Specifically, MCB carefully selects when to send coordination information and which SCH to use for data transmissions. This allows the protocol to make efficient use of the additional capacity afforded by otherwise unused channels.

In comparison to single-channel beaconing solutions in the scope of a simulation study, we illustrate that by reducing channel load on the CCH, information can be sent more frequently and with higher reliability. This results in the fact that MCB is not only able to compensate the overhead introduced by the necessary channel coordination, but also delivers substantially improved protocol performance compared to state of the art (that is, single-channel) beaconing protocols. In spite of operating as a single-radio system – and thus losing capacity to channel switching, to synchronization and to coordination overhead – MCB allows in a typical scenario to inform twice as many vehicles in the first 100 ms.

Chapter 7

Conclusion

In the context of this PhD thesis we focused on *Efficient Wireless Communication in Vehicular Networks*. We started with a general motivation of wireless communication in the automotive domain to achieve information dissemination among vehicles for different application domains, e.g., safety, efficiency, and entertainment-applications. Focusing explicitly on ad-hoc based wireless communication based on the IEEE 802.11p amendment of WLAN, we studied the theoretical capacity bounds of this communication protocol in terms of maximum goodput and channel utilization for different transmit data rates and frame sizes.

As a follow up, we studied available communication protocols and efforts of standardization bodies for Inter-Vehicle Communication (IVC) and found out that current approaches are not sufficient at fulfilling important requirements necessary to bring vehicular communication onto the road. This is mainly caused since *(a)* many specialized applications build upon their own protocol stack and are not designed to co-exist on the same wireless channel with many other application specific protocols, and *(b)* even for generalized protocol stacks supporting multiple different application domains, they focus on message dissemination techniques which we show to be actually harmful to the overall networking performance. Here we explicitly focus on IEEE 802.11 based unicast communication as used by ETSI ITS-G5 Geocasting, for which we show in terms of analytics, cross-validated simulations, and hardware experiments, as well as large scale Vehicular Ad Hoc Network (VANET) simulations, that this type of communication is significantly degrading the networking performance. This not only holds for packets sent by this unicast communication principle, but affects all communication using the same EDCA access category, which by default is limited to four distinct categories by each node.

Based on these findings, and the necessity to allow many application specific protocols to co-exist on the same wireless network, we developed a novel and integrated networking architecture by categorizing possible applications in the context

of vehicular communications within four distinct classes. These four classes have different requirements on the communication channel, and thus employ different paradigms for message dissemination. We selected possible candidate protocols for each of the four classes to fulfill the requirements of applications, and developed protocols where current state-of-the-art approaches were not sufficient.

This way, we developed and evaluated a Bloom filter based neighbor table maintenance protocol, which serves as a fundamental basis for our integrated class-based networking concept. Our developed probabilistic message dissemination *Bloom Hopping* allows to disseminate information among 2-hop neighbors of a node by choosing only a very small set forwarding vehicles. The main advantage of our algorithm is that it is completely independent from the underlying road topology, since it does not rely on any geographical or street-topology information. Results show that our proposed message dissemination algorithm is able to lower the necessary load on the wireless channel, and at the same time, deliver the message to more nodes in comparison to conventional approaches. This makes our proposed protocol an ideal candidate for message dissemination among direct (1-hop) and indirect (2-hop) neighbors.

Focusing on the scalability of efficient wireless communication in the vehicular networking context, we investigated the usage of multiple channels for message dissemination for single-radio systems. By using a split-phase channel switching system, we propose scheduling algorithms which carefully decide when to send information on which channel by employing a novel prioritization scheme. This allows nodes to transmit information based on a priority by ensuring that nodes trying to transmit information with a lower priority still can receive the more important information having a higher priority. Our investigations reveal in comparison to single-channel approaches that we are not only able to lower individual channel utilization by using our novel multi-channel beaconing approach, but also can improve the delay at which information is disseminated to other vehicles.

Putting everything together, the work presented in this PhD thesis provides a robust basis onto which future developments for IVC can build. We believe that our proposed networking architecture, as well as our efficient message dissemination algorithms and multi-channel scheduling approach allow a wide-range of new applications to enter the stage of ITS.

In particular, we look forward to the *IEEE 802.11 Next Generation V2X* Study Group founded in March 2018 to bring recent advances from the Physical layer (PHY) layer found in IEEE 802.11n onwards to the matured and robust IEEE 802.11p technology.

List of Abbreviations

AC	Access Category
ACK	Acknowledgment
AIFS	Arbitration Interframe Space
AP	Access Point
ARQ	Automatic Repeat Request
ATB	Adaptive Traffic Beacon
BSM	Basic Safety Message
BSS	Basic Service Set
CACC	Cooperative Adaptive Cruise Control
CAM	Cooperative Awareness Message
CBF	Contention-based Forwarding
CCA	Clear Channel Assessment
CCH	Control Channel
DCC	Decentralized Congestion Control
DENM	Decentralized Environmental Notification Message
DSC	DCC Sensitivity Control
DTN	Delay Tolerant Networking
ECC	Electronic Communications Committee
eCDF	empirical Cumulative Distribution Function
ECU	Electronic Control Unit
EDCA	Enhanced Distributed Channel Access
FCC	Federal Communications Commission
FCS	Frame Check Sequence
FOT	Field Operational Test
GPS	Global Positioning System
ISM	Industrial, Scientific and Medical
ITS	Intelligent Transportation Systems
IVC	Inter-Vehicle Communication
LOS	Line of Sight
MAC	Media Access Control

MANET	Mobile Ad Hoc Network
MCB	Multi-Channel Beacon
MR-MC	Multi-Radio Multi-Channel
OCB	“outside the context of a BSS”
OFDM	Orthogonal Frequency Division Multiplex
PHY	Physical layer
RCS	Random Channel Selector
ROI	Region of Interest
RSU	Roadside Unit
SCH	Service Channel
SNIR	Signal to Noise plus Interference Ratio
SNR	Signal to Noise Ratio
SR-MC	Single-Radio Multi-Channel
TDC	Transmit Datarate Control
TIS	Traffic Information System
TPC	Transmit Power Control
TRC	Transmit Rate Control
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-X
VANET	Vehicular Ad Hoc Network
WAVE	Wireless Access in the Vehicular Environment
WLAN	Wireless LAN

List of Figures

List of Tables

Bibliography

[1] World Health Organization (WHO), "Global Status Report on Road Safety 2015," World Health Organization (WHO), Tech. Rep., 2015.

[2] S. Joerer, "Improving Intersection Safety with Inter-Vehicle Communication," PhD Thesis (Dissertation), Institute of Computer Science, Innsbruck, Austria, Jul. 2016.

[3] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, Mar. 2000.

[4] K. Kosek-Szott, M. Natkaniec, and A. R. Pach, "A simple but accurate throughput model for IEEE 802.11 EDCA in saturation and non-saturation conditions," *Elsevier Computer Networks*, vol. 55, no. 3, pp. 622–635, Feb. 2011.

[5] S. Eichler, "Performance Evaluation of the IEEE 802.11p WAVE Communication Standard," in *66th IEEE Vehicular Technology Conference (VTC2007-Fall)*, Baltimore, MD, Oct. 2007, pp. 2199–2203.

[6] B. Cheng, A. Rostami, and M. Gruteser, "Experience: Accurate Simulation of Dense Scenarios with Hundreds of Vehicular Transmitters," in *22nd ACM International Conference on Mobile Computing and Networking (MobiCom 2016)*, New York, NY: ACM, Oct. 2016, pp. 271–279.

[7] F. Klingler, G. S. Pannu, C. Sommer, B. Bloessl, and F. Dressler, "Field Testing Vehicular Networks using OpenC2X," in *15th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys 2017), Poster Session*, Niagara Falls, NY: ACM, Jun. 2017, pp. 178–178.

[8] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions," *IEEE Communications Surveys and Tutorials*, vol. 13, no. 4, pp. 584–616, Nov. 2011.

[9] C. Sommer and F. Dressler, Vehicular Networking. Cambridge University Press, Nov. 2014.

[10] "Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description; Stage 2," 3GPP, TS 36.300 v14.7.0, Jul. 2018.

[11] "Service requirements for V2X services," 3GPP, TS 22.185 v14.4.0, Jun. 2018.

[12] "Architecture enhancements for V2X services," 3GPP, TS 23.285 v14.7.0, Jun. 2018.

[13] ETSI, "Intelligent Transport Systems (ITS); Decentralized Congestion Control Mechanisms for Intelligent Transport Systems operating in the 5 GHz range; Access layer part," ETSI, TS 102 687 V1.1.1, Jul. 2011.

[14] IEEE, "IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Resource Manager," IEEE, Std 1609.1-2006, Oct. 2006.

[15] IEEE, "IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Security Services for Applications and Management Messages," IEEE, Std 1609.2-2006, Jul. 2006.

[16] IEEE, "IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Networking Services," IEEE, Std 1609.3-2007, Apr. 2007.

[17] IEEE, "IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Multi-channel Operation," IEEE, Std 1609.4-2010, Feb. 2011.

[18] ARIB, "700 MHz Band Intelligent Transport Systems," English, ARIB, STD T109-v1.2, Dec. 2013.

[19] W. Klein Wolterink, G. Heijenk, and G. Karagiannis, "Constrained Geocast to Support Cooperative Adaptive Cruise Control (CACC) Merging," in *2nd IEEE Vehicular Networking Conference (VNC 2010)*, Jersey City, NJ: IEEE, Dec. 2010, pp. 41–48.

[20] S. Oncu, J. Ploeg, N. Van De Wouw, and H. Nijmeijer, "Cooperative Adaptive Cruise Control: Network-Aware Analysis of String Stability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1527–1537, Aug. 2014.

[21] M. Segata, B. Bloessl, S. Joerer, C. Sommer, M. Gerla, R. Lo Cigno, and F. Dressler, "Towards Communication Strategies for Platooning: Simulative and Experimental Evaluation," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 12, pp. 5411–5423, Dec. 2015.

[22] IEEE, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE, Std 802.11-2012, 2012.

[23] ETSI, "Intelligent Transport Systems (ITS); Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band," ETSI, EN 302 663 V1.2.1, Jul. 2013.

[24] G. S. Pannu, F. Klingler, C. Sommer, and F. Dressler, "QQDCA: Adapting IEEE 802.11 EDCA for Unicast Transmissions at High Topology Dynamics," in *9th IEEE Vehicular Networking Conference (VNC 2017)*, Torino, Italy: IEEE, Nov. 2017, pp. 295–302.

[25] F. Klingler, F. Dressler, and C. Sommer, "IEEE 802.11p Unicast Considered Harmful," in *7th IEEE Vehicular Networking Conference (VNC 2015)*, Kyoto, Japan: IEEE, Dec. 2015, pp. 76–83.

[26] European Telecommunications Standards Institute, "Intelligent Transport Systems (ITS); Harmonized Channel Specifications for Intelligent Transport Systems operating in the 5 GHz frequency band," ETSI, TS 102 724 V1.1.1, Oct. 2012.

[27] ETSI, "Intelligent Transport Systems (ITS); Vehicular communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality," ETSI, Tech. Rep. 302 636-4-1 V1.2.1, Jul. 2014.

[28] F. Klingler, F. Dressler, and C. Sommer, "The Impact of Head of Line Blocking in Highly Dynamic WLANs," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 8, pp. 7664–7676, Aug. 2018.

[29] S. Kühlmorgen, I. Llatser, A. Festag, and G. Fettweis, "Performance Evaluation of ETSI GeoNetworking for Vehicular Ad hoc Networks," in *81rd Vehicular Technology Conference (VTC2015-Spring)*, Glasgow, UK: IEEE, May 2015.

[30] F. Klingler, "Context-Aware and Class-Based Broadcasting in VANETs," in *International Conference on Networked Systems (NetSys 2015), PhD Forum*, Cottbus, Germany, Mar. 2015.

[31] F. Dressler, F. Klingler, C. Sommer, and R. Cohen, "Not All VANET Broadcasts Are the Same: Context-Aware Class Based Broadcast," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 17–30, Feb. 2018.

[32] F. Klingler, R. Cohen, C. Sommer, and F. Dressler, "Bloom Hopping: Bloom filter based 2-Hop Neighbor Management in VANETs," *IEEE Transactions on Mobile Computing*, 2018, available online.

[33] F. Klingler, F. Dressler, J. Cao, and C. Sommer, "Use Both Lanes: Multi-Channel Beaconing for Message Dissemination in Vehicular Networks," in *10th IEEE/IFIP Conference on Wireless On demand Network Systems and Services (WONS 2013)*, Banff, Canada: IEEE, Mar. 2013, pp. 162–169.

[34] F. Klingler, "Improving Multi-Channel Beaconing in Vehicular Networks," in *3rd GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*, Ulm, Germany, Mar. 2015.

[35] F. Klingler, F. Dressler, J. Cao, and C. Sommer, "MCB - A Multi-Channel Beaconing Protocol," *Elsevier Ad Hoc Networks*, vol. 36, no. 1, pp. 258–269, Jan. 2016.

[36] F. Dressler, F. Klingler, M. Segata, and R. Lo Cigno, "Cooperative Driving and the Tactile Internet," *Proceedings of the IEEE*, 2018, to appear.

[37] I. Turcanu, F. Klingler, C. Sommer, A. Baiocchi, and F. Dressler, "Duplicate Suppression for Efficient Floating Car Data Collection in Heterogeneous LTE-DSRC Vehicular Networks," *Elsevier Computer Communications*, vol. 123, pp. 54–64, Jun. 2018.

[38] J. Heinovski, F. Klingler, F. Dressler, and C. Sommer, "A Simulative Analysis of the Performance of IEEE 802.11p and ARIB STD-T109," *Elsevier Computer Communications*, vol. 122, pp. 84–92, Jun. 2018.

[39] F. Klingler, J. Blobel, and F. Dressler, "Agriculture meets IEEE 802.11p: A Feasibility Study," in *15th IEEE International Symposium on Wireless Communication Systems (ISWCS 2018)*, Lisbon, Portugal: IEEE, Aug. 2018.

[40] S. Loewen, F. Klingler, C. Sommer, and F. Dressler, "Backwards Compatible Extension of CAMs/DENMs for Improved Bike Safety on the Road," in *9th IEEE Vehicular Networking Conference (VNC 2017), Poster Session*, Torino, Italy: IEEE, Nov. 2017, pp. 43–44.

[41] B. Bloessl, F. Klingler, F. Missbrenner, and C. Sommer, "A Systematic Study on the Impact of Noise and OFDM Interference on IEEE 802.11p," in *9th IEEE Vehicular Networking Conference (VNC 2017)*, Torino, Italy: IEEE, Nov. 2017, pp. 287–290.

[42] F. Klingler, G. S. Pannu, C. Sommer, and F. Dressler, "Connecting Simulation and Real World: IEEE 802.11p in the Loop," in *23rd ACM International Conference on Mobile Computing and Networking (MobiCom 2017), Poster Session*, Snowbird, UT: ACM, Oct. 2017, pp. 561–563.

[43] S. Laux, G. S. Pannu, S. Schneider, J. Tiemann, F. Klingler, C. Sommer, and F. Dressler, "OpenC2X - An Open Source Experimental and Prototyping Platform Supporting ETSI ITS-G5," in *8th IEEE Vehicular Networking Conference (VNC 2016), Demo Session*, Columbus, OH: IEEE, Dec. 2016, pp. 152–153.

[44] J. Heinovski, F. Klingler, F. Dressler, and C. Sommer, "Performance Comparison of IEEE 802.11p and ARIB STD-T109," in *8th IEEE Vehicular Networking Conference (VNC 2016)*, Columbus, OH: IEEE, Dec. 2016, pp. 1–8.

[45] M. Mutschlechner, F. Klingler, F. Erlacher, F. Hagenauer, M. Kiessling, and F. Dressler, "Reliable Communication using Erasure Codes for Monitoring Bats in the Wild," in *33rd IEEE Conference on Computer Communications (INFOCOM 2014), Student Activities*, Toronto, Canada: IEEE, Apr. 2014, pp. 189–190.

[46] F. Erlacher, F. Klingler, C. Sommer, and F. Dressler, "On the Impact of Street Width on 5.9 GHz Radio Signal Propagation in Vehicular Networks," in *11th IEEE/IFIP Conference on Wireless On demand Network Systems and Services (WONS 2014)*, Obergurgl, Austria: IEEE, Apr. 2014, pp. 143–146.

[47] F. Klingler, S. Tang, X. Liu, F. Dressler, C. Sommer, and J. Cao, "Faster Distributed Localization of Large Numbers of Nodes Using Clustering," in *38th IEEE Conference on Local Computer Networks (LCN 2013)*, Sydney, Australia: IEEE, Oct. 2013, pp. 728–731.

[48] M. Segata, B. Bloessl, S. Joerer, F. Erlacher, M. Mutschlechner, F. Klingler, C. Sommer, R. Lo Cigno, and F. Dressler, "Shadowing or Multi-Path Fading: Which Dominates in Inter-Vehicle Communication?" University of Innsbruck, Institute of Computer Science, Technical Report CCS-2013-03, Jun. 2013.

[49] S. Yousefi, M. S. Mousavi, and M. Fathy, "Vehicular Ad Hoc Networks (VANETs): Challenges and Perspectives," in *6th International Conference on ITS Telecommunications (ITST 2006)*, Chengdu, China, Jun. 2006, pp. 761–766.

[50] D. Jiang and L. Delgrossi, "IEEE 802.11p: Towards an International Standard for Wireless Access in Vehicular Environments," in *67th IEEE Vehicular Technology Conference (VTC2008-Spring)*, Marina Bay, Singapore, May 2008, pp. 2036–2040.

[51] IEEE, "Wireless Access in Vehicular Environments," IEEE, Std 802.11p-2010, Jul. 2010.

[52] J. Lee, Y. Su, and C. Shen, "A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," in *33rd Annual Conference of the IEEE Industrial Electronics Society (IECON 2007)*, Taipei, Taiwan, Nov. 2007, pp. 46–51.

[53] F. Hagenauer, C. Sommer, S. Merschjohann, T. Higuchi, F. Dressler, and O. Altintas, "Cars as the Base for Service Discovery and Provision in Highly Dynamic Networks," in *35th IEEE Conference on Computer Communications (IN-*

FOCOM 2016), Demo Session, San Francisco, CA: IEEE, Apr. 2016, pp. 358–359.

[54] C. Sommer, S. Joerer, and F. Dressler, "On the Applicability of Two-Ray Path Loss Models for Vehicular Network Simulation," in *4th IEEE Vehicular Networking Conference (VNC 2012)*, Seoul, Korea: IEEE, Nov. 2012, pp. 64–69.

[55] J. Ploeg, B. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and Experimental Evaluation of Cooperative Adaptive Cruise Control," in *IEEE International Conference on Intelligent Transportation Systems (ITSC 2011)*, Washington, DC: IEEE, Oct. 2011, pp. 260–265.

[56] P. Fernandes and U. Nunes, "Platooning With IVC-Enabled Autonomous Vehicles: Strategies to Mitigate Communication Delays, Improve Safety and Traffic Flow," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 91–106, Mar. 2012.

[57] M. Sepulcre, J. Mittag, P. Santi, H. Hartenstein, and J. Gozalvez, "Congestion and Awareness Control in Cooperative Vehicular Systems," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1260–1279, Jul. 2011.

[58] ETSI, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service," ETSI, EN 302 637-2 V1.3.2, Nov. 2014.

[59] SAE, "Dedicated Short Range Communications (DSRC) Message Set Dictionary," SAE, Tech. Rep. J2735-200911, Nov. 2009.

[60] L. Wischhof, A. Ebner, and H. Rohling, "Information Dissemination in Self-Organizing Intervehicle Networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 1, pp. 90–101, Mar. 2005.

[61] G. Bansal, B. Cheng, A. Rostami, K. Sjoberg, J. B. Kenney, and M. Gruteser, "Comparing LIMERIC and DCC Approaches for VANET Channel Congestion Control," in *6th IEEE International Symposium on Wireless Vehicular Communications (WiVec 2014)*, Vancouver, Canada: IEEE, Sep. 2014, pp. 1–7.

[62] ETSI, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specification of Decentralized Environmental Notification Basic Service," ETSI, Tech. Rep. 302 637-3 V1.2.1, Sep. 2014.

[63] F. J. Ros, P. M. Ruiz, and I. Stojmenovic, "Reliable and Efficient Broadcasting in Vehicular Ad Hoc Networks," in *69th IEEE Vehicular Technology Conference (VTC2009-Spring)*, Barcelona, Spain: IEEE, Apr. 2009, pp. 1–5.

[64] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network," in *5th ACM International Conference on Mobile Computing and Networking (MobiCom 1999)*, Seattle, WA, Aug. 1999, pp. 151–162.

[65] E. Van de Velde and C. Blondia, "Adaptive REACT protocol for Emergency Applications in Vehicular Networks," in *32nd IEEE Conference on Local Computer Networks (LCN 2007)*, Dublin, Ireland: IEEE, Oct. 2007, pp. 613–619.

[66] C. Sommer, O. K. Tonguz, and F. Dressler, "Traffic Information Systems: Efficient Message Dissemination via Adaptive Beaconing," *IEEE Communications Magazine*, vol. 49, no. 5, pp. 173–179, May 2011.

[67] R. S. Schwartz, A. E. Ohazulike, C. Sommer, H. Scholten, F. Dressler, and P. Havinga, "On the Applicability of Fair and Adaptive Data Dissemination in Traffic Information Systems," *Elsevier Ad Hoc Networks*, vol. 13, Part B, pp. 428–443, Feb. 2014.

[68] G. Caizzone, P. Giacomazzi, L. Musumeci, and G. Verticale, "A power control algorithm with high channel availability for vehicular ad hoc networks," in *IEEE International Conference on Communications (ICC 2005)*, Seoul, Korea: IEEE, May 2005, pp. 16–20.

[69] T. Tielert, D. Jiang, H. Hartenstein, and L. Delgrossi, "Joint Power/Rate Congestion Control Optimizing Packet Reception in Vehicle Safety Communications," in *10th ACM International Workshop on Vehicular Internetworking (VANET 2013)*, Taipei, Taiwan: ACM, Jun. 2013, pp. 51–60.

[70] T. Tielert, D. Jiang, Q. Chen, L. Delgrossi, and H. Hartenstein, "Design Methodology and Evaluation of Rate Adaptation Based Congestion Control for Vehicle Safety Communications," in *3rd IEEE Vehicular Networking Conference (VNC 2011)*, Amsterdam, Netherlands: IEEE, Nov. 2011, pp. 116–123.

[71] G. Bansal, J. Kenney, and C. Rohrs, "LIMERIC: A Linear Adaptive Message Rate Algorithm for DSRC Congestion Control," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 9, pp. 4182–4197, Nov. 2013.

[72] C. Sommer, S. Joerer, M. Segata, O. K. Tonguz, R. Lo Cigno, and F. Dressler, "How Shadowing Hurts Vehicular Communications and How Dynamic Beaconing Can Help," *IEEE Transactions on Mobile Computing*, vol. 14, no. 7, pp. 1411–1421, Jul. 2015.

[73] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970.

[74] A. Broder and M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," *Internet Mathematics*, vol. 1, no. 4, pp. 485–509, Jan. 2005.

[75] A. Marandi, M. F. Imani, and K. Salamatian, "Practical Bloom filter based epidemic forwarding and congestion control in DTNs: A comparative analysis," *Elsevier Computer Communications*, vol. 48, pp. 98–110, Jul. 2014.

[76] F. Angius, M. Gerla, and G. Pau, "BLOOGO: BLOOm Filter Based GOssip Algorithm for Wireless NDN," in *13th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2012), 1st ACM Workshop on Emerging Name-Oriented Mobile Networking Design - Architecture, Algorithms, and Applications (NoM 2012)*, Hilton Head, SC: ACM, Jun. 2012, pp. 25–30.

[77] A. Marandi, M. F. Imani, and K. Salamatian, "Optimization of Bloom Filter Parameters for Practical Bloom Filter Based Epidemic Forwarding in DTNs," *arXiv arXiv:1208.3871*, 2012.

[78] K. Na Nakorn, Y. Ji, and K. Rojviboonchai, "Bloom Filter for Fixed-Size Beacon in VANET," in *79th IEEE Vehicular Technology Conference (VTC2014-Spring)*, Seoul, Korea: IEEE, May 2014, pp. 1–5.

[79] F. A. Silva, A. Boukerche, T. R. B. Silva, L. B. Ruiz, and A. A. Loureiro, "Geolocalized content availability in VANETs," *Elsevier Ad Hoc Networks*, vol. 36, no. 2, pp. 425–434, Jan. 2015.

[80] D. Borsetti and J. Gozalvez, "Infrastructure-assisted geo-routing for cooperative vehicular networks," in *2nd IEEE Vehicular Networking Conference (VNC 2010)*, Jersey City, NJ: IEEE, Dec. 2010, pp. 255–262.

[81] European Telecommunications Standards Institute, "Intelligent Transport Systems (ITS); Vehicular communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality," ETSI, TS 102 636-4-1 V1.1.1, Jun. 2011.

[82] L. Zhang, B. Yu, and J. Pan, "GeoMob: A mobility-aware geocast scheme in metropolitans via taxicabs and buses," in *33rd IEEE Conference on Computer Communications (INFOCOM 2014)*, Toronto, Canada: IEEE, Apr. 2014, pp. 1279–1787.

[83] R. Jiang, Y. Zhu, T. He, Y. Liu, and L. M. Ni, "Exploiting Trajectory-Based Coverage for Geocast in Vehicular Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3177–3189, Dec. 2014.

[84] C. Sommer, O. K. Tonguz, and F. Dressler, "Adaptive Beaconing for Delay-Sensitive and Congestion-Aware Traffic Information Systems," in *2nd IEEE Vehicular Networking Conference (VNC 2010)*, Jersey City, NJ: IEEE, Dec. 2010, pp. 1–8.

[85] L. Wischhof, A. Ebner, H. Rohling, M. Lott, and R. Halfmann, "SOTIS - A Self-Organizing Traffic Information System," in *57th IEEE Vehicular Technology Conference (VTC2003-Spring)*, Jeju, South Korea: IEEE, Apr. 2003, pp. 2442–2446.

[86] R. S. Schwartz, A. E. Ohazulike, C. Sommer, H. Scholten, F. Dressler, and P. Havinga, "Fair and Adaptive Data Dissemination for Traffic Information Systems," in *4th IEEE Vehicular Networking Conference (VNC 2012)*, Seoul, Korea: IEEE, Nov. 2012, pp. 1–8.

[87] ETSI, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service," ETSI, TS 102 637-2 V1.1.1, Apr. 2010.

[88] ETSI, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service," ETSI, TS 102 637-3 V1.1.1, Sep. 2010.

[89] S. Graefling, P. Maehoenen, and J. Riihijaervi, "Performance Evaluation of IEEE 1609 WAVE and IEEE 802.11p for Vehicular Communications," in *Second International Conference on Ubiquitous and Future Networks (ICUFN)*, Jeju Island, South Korea: IEEE, Jun. 2010, pp. 344–348.

[90] R. A. Uzcátegui and G. Acosta-Marum, "WAVE: A Tutorial," *IEEE Communications Magazine*, vol. 47, no. 5, pp. 126–133, May 2009.

[91] J. Crichigno, M.-Y. Wu, and W. Shu, "Protocols and architectures for channel assignment in wireless mesh networks," *Elsevier Ad Hoc Networks*, vol. 6, no. 7, pp. 1051–1077, 2008.

[92] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Transactions on Networking*, vol. 8, no. 3, pp. 281–293, 2000.

[93] D. Guo, Y. Liu, X. Li, and P. Yang, "False negative problem of counting bloom filter," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 5, pp. 651–664, Mar. 2010.

[94] T. Darwish and K. A. Bakar, "Traffic density estimation in vehicular ad hoc networks: A review," *Elsevier Ad Hoc Networks*, vol. 24, no. A, pp. 337–351, Jan. 2015.

[95] L. Garelli, C. Casetti, C.-F. Chiasserini, and M. Fiore, "Mobsampling: V2V Communications for Traffic Density Estimation," in *73rd IEEE Vehicular Technology Conference (VTC2011-Spring)*, Budapest, Hungary: IEEE, May 2011.

[96] C. Lochert, B. Scheuermann, and M. Mauve, "A probabilistic method for cooperative hierarchical aggregation of data in VANETs," *Elsevier Ad Hoc Networks*, vol. 8, no. 5, pp. 518–530, Jul. 2010.

[97] O. Papapetrou, W. Siberski, and W. Nejdl, "Cardinality estimation and dynamic length adaptation for Bloom filters," *Springer Distributed and Parallel Databases*, vol. 28, no. 2-3, pp. 119–156, Dec. 2010.

[98] M. C. Jeffrey and J. G. Steffan, "Understanding Bloom Filter Intersection for Lazy Address-set Disambiguation," in *23rd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA 2011)*, San Jose, CA: ACM, Jun. 2011, pp. 345–354.

[99] S. J. Swamidass and P. Baldi, "Mathematical Correction for Fingerprint Similarity Measures to Improve Chemical Retrieval," *Journal of Chemical Information and Modeling*, vol. 47, no. 3, pp. 952–964, 2007.

[100] C. Sommer, D. Eckhoff, R. German, and F. Dressler, "A Computationally Inexpensive Empirical Model of IEEE 802.11p Radio Shadowing in Urban Environments," in *8th IEEE/IFIP Conference on Wireless On demand Network Systems and Services (WONS 2011)*, Bardonecchia, Italy: IEEE, Jan. 2011, pp. 84–90.

[101] C. Sommer and F. Dressler, "Using the Right Two-Ray Model? A Measurement based Evaluation of PHY Models in VANETs," in *17th ACM International Conference on Mobile Computing and Networking (MobiCom 2011), Poster Session*, Las Vegas, NV: ACM, Sep. 2011.

[102] D. Eckhoff, A. Brummer, and C. Sommer, "On the Impact of Antenna Patterns on VANET Simulation," in *8th IEEE Vehicular Networking Conference (VNC 2016)*, Columbus, OH: IEEE, Dec. 2016, pp. 17–20.

[103] C. Campolo, C. Sommer, F. Dressler, and A. Molinaro, "On the Impact of Adjacent Channel Interference in Multi-Channel VANETs," in *IEEE International Conference on Communications (ICC 2016)*, Kuala Lumpur, Malaysia: IEEE, May 2016, pp. 2626–2632.

[104] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. K. Haneveld, T. Parker, O. Visser, H. S. Lichte, and S. Valentin, "Simulating Wireless and Mobile Networks in OMNeT++ – The MiXiM Vision," in *1st ACM/ICST International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2008): 1st ACM/ICST International Workshop on OMNeT++ (OMNeT++ 2008)*, Marseille, France: ACM, Mar. 2008.

[105] K. Wessel, M. Swigulski, A. Köpke, and D. Willkomm, "MiXiM – The Physical Layer: An Architecture Overview," in *2nd ACM/ICST International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2009): 2nd ACM/ICST International Workshop on OMNeT++ (OMNeT++ 2009)*, Rome, Italy: ACM, Mar. 2009.

[106] G. Pei and T. R. Henderson, "Validation of OFDM Error Rate Model in ns-3," Boeing Research & Technology, Seattle, WA, Tech. Rep., 2010.

[107] L. E. Miller, "Validation of 802.11a/UWB Coexistence Simulation," National Institute of Standards and Technology (NIST), Tech. Rep., Oct. 2003.

[108] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent Development and Applications of SUMO - Simulation of Urban MObility," *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, Dec. 2012.

[109] P. Bhagwat, P. Bhattacharya, A. Krishna, and S. Tripathi, "Enhancing throughput over wireless LANs using channel state dependent packet scheduling," in *15th IEEE Conference on Computer Communications (INFOCOM 1996)*, San Francisco, CA: IEEE, Mar. 1996, pp. 1133–1140.

[110] E. Schoch, F. Kargl, M. Weber, and T. Leinmüller, "Communication Patterns in VANETs," *IEEE Communications Magazine*, vol. 46, no. 11, pp. 119–125, Nov. 2008.

[111] M. Kihl, M. Sichitiu, T. Ekeroth, and M. Rozenberg, "Reliable Geographical Multicast Routing in Vehicular Ad-Hoc Networks," in *Wired/Wireless Internet Communications*, Springer, 2007, pp. 315–325.

[112] A. Böhm, M. Jonsson, K. Kunert, and A. Vinel, "Context-Aware Retransmission Scheme for Increased Reliability in Platooning Applications," in *6th IFIP/IEEE International Workshop on Communication Technologies for Vehicles (Nets4Cars 2014-Spring)*, Offenburg, Germany: Springer, May 2014, pp. 30–42.

[113] F. Li and Y. Wang, "Routing in Vehicular Ad hoc Networks: A Survey," *IEEE Vehicular Technology Magazine*, vol. 2, no. 2, pp. 12–22, Jun. 2007.

[114] J. Bernsen and D. Manivannan, "Unicast routing protocols for vehicular ad hoc networks: A critical comparison and classification," *Elsevier Pervasive and Mobile Computing*, vol. 5, no. 1, pp. 1–18, Feb. 2009.

[115] M. L. Sichitiu and M. Kihl, "Inter-Vehicle Communication Systems: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 2, pp. 88–105, 2008.

[116] L. Urquiza-Aguiar, C. Tripp-Barba, and Á. R. Muir, "Mitigation of packet duplication in VANET unicast protocols," *Elsevier Ad Hoc Networks*, vol. 52, pp. 63–77, Dec. 2016.

[117] B. Han, A. Schulman, F. Gringoli, N. Spring, B. Bhattacharjee, L. Nava, L. Ji, S. Lee, and R. Miller, "Maranello: Practical Partial Packet Recovery for 802.11," in *7th USENIX Conference on Networked Systems Design and Implementation (NSDI 2010)*, San Jose, CA, Apr. 2010, pp. 205–218.

[118] Y. Xie, I. W.-H. Ho, and L. F. Xie, "Stochastic Modeling and Analysis of Unicast Performance in 802.11p VANETs," in *10th International Conference on Information, Communications and Signal Processing (ICICS)*, Singapore: IEEE, Dec. 2015.

[119] R. Reinders, M. Van Eenennaam, G. Karagiannis, and G. Heijenk, "Contention window analysis for beaconing in VANETs," in *7th International Wireless Communications and Mobile Computing Conference (IWCMC 2011)*, Istanbul, Turkey: IEEE, Jul. 2011, pp. 1481–1487.

[120] R. Lisovy, M. Sojka, and Z. Hanzálek, "IEEE 802.11p Linux Kernel Implementation," Industrial Informatics Research Center, Czech Technical University, Prague, Czech Republic, Technical Report, Dec. 2014.

[121] W. Alasmary and W. Zhuang, "Mobility impact in IEEE 802.11p infrastructureless vehicular networks," *Elsevier Ad Hoc Networks*, vol. 10, no. 2, pp. 222–230, Mar. 2012.

[122] C. Sommer, R. German, and F. Dressler, "Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis," *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 3–15, Jan. 2011.

[123] M. Segata, B. Bloessl, S. Joerer, C. Sommer, M. Gerla, R. Lo Cigno, and F. Dressler, "Towards Inter-Vehicle Communication Strategies for Platooning Support," in *7th IFIP/IEEE International Workshop on Communication Technologies for Vehicles (Nets4Cars 2014-Fall)*, Saint-Petersburg, Russia: IEEE, Oct. 2014, pp. 1–6.

[124] T. K. Mak, K. P. Laberteaux, and R. Sengupta, "A Multi-channel VANET Providing Concurrent Safety and Commercial Services," in *2nd ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2005)*, Cologne, Germany: ACM, Sep. 2005, pp. 1–9.

[125] E. Blanton and M. Allman, "On making TCP more robust to packet reordering," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 1, pp. 20–30, Jan. 2002.

[126] G. Bianchi, A. Di Stefano, C. Giaconia, L. Scalia, G. Terrazzino, and I. Tinnirello, "Experimental assessment of the backoff behavior of commercial IEEE 802.11b network cards," in *26th IEEE Conference on Computer Communications (INFOCOM 2007)*, Anchorage, AK: IEEE, May 2007, pp. 1181–1189.

[127] M. Vanhoef and F. Piessens, "Advanced Wi-Fi Attacks Using Commodity Hardware," in *30th Annual Computer Security Applications Conference (ACSAC 2014)*, New Orleans, LA: ACM, Dec. 2014, pp. 256–265.

[128] N. Wisitpongphan, O. K. Tonguz, J. S. Parikh, P. Mudalige, F. Bai, and V. Sadekar, "Broadcast Storm Mitigation Techniques in Vehicular Ad Hoc Networks," *IEEE Wireless Communications*, vol. 14, no. 6, pp. 84–94, Dec. 2007.

[129] J. Jeong, A. Petrescu, T. Oh, D. Liu, and C. Perkins, "Problem Statement for IP Wireless Access in Vehicular Environments)," IETF, Internet-Draft (work in progress) draft-jeong-ipwave-problem-statement-00.txt, Jun. 2017.

[130] Y.-T. Yu, T. Punihaole, M. Gerla, and M. Sanadidi, "Content routing in the Vehicle Cloud," in *IEEE Military Communications Conference (MILCOM 2012)*, Orlando, FL: IEEE, Oct. 2012, pp. 1–6.

[131] F. Farnoud and S. Valaee, "Reliable Broadcast of Safety Messages in Vehicular Ad Hoc Networks," in *28th IEEE Conference on Computer Communications (INFOCOM 2009)*, Rio de Janeiro, Brazil: IEEE, Apr. 2009, pp. 226–234.

[132] N. Liu, M. Liu, W. Lou, G. Chen, and J. Cao, "PVA in VANETs: Stopped cars are not silent," in *30th IEEE Conference on Computer Communications (INFOCOM 2011), Mini-Conference*, Shanghai, China: IEEE, Apr. 2011, pp. 431–435.

[133] C. Sommer, D. Eckhoff, and F. Dressler, "IVC in Cities: Signal Attenuation by Buildings and How Parked Cars Can Improve the Situation," *IEEE Transactions on Mobile Computing*, vol. 13, no. 8, pp. 1733–1745, Aug. 2014.

[134] D. Jia, K. Lu, J. Wang, X. Zhang, and X. Shen, "A Survey on Platoon-Based Vehicular Cyber-Physical Systems," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 263–284, 2016.

[135] F. Dressler, H. Hartenstein, O. Altintas, and O. K. Tonguz, "Inter-Vehicle Communication – Quo Vadis," *IEEE Communications Magazine*, vol. 52, no. 6, pp. 170–177, Jun. 2014.

[136] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements," IEEE, Std 802.11e-2005, 2005.

[137] K. Lee, U. Lee, and M. Gerla, "Geo-Opportunistic Routing for Vehicular Networks," *IEEE Communications Magazine*, vol. 48, no. 5, pp. 164–170, May 2010.

[138] B. Korte and J. Vygen, Combinatorial Optimization: Theory and Algorithms, 5th, ser. Algorithms and Combinatorics 21. Berlin Heidelberg: Springer, 2012.

[139] R. Michoud, A. Orozco, and G. Llano, "Mobile ad-hoc routing protocols survey for the design of VANET applications," in *IEEE Colombian Intelligent Transportation Systems Symposium (CITSS 2012)*, IEEE, Aug. 2012.

[140] J. Petit, F. Schaub, M. Feiri, and F. Kargl, "Pseudonym Schemes in Vehicular Networks: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 1, pp. 228–255, 2015.

[141] European Telecommunications Standards Institute, "Intelligent Transport Systems (ITS); Security; Threat, Vulnerability and Risk Analysis (TVRA)," ETSI, TS 102 893 V1.1.1, Mar. 2010.

[142] O. K. Tonguz, N. Wisitpongphan, and F. Bai, "DV-CAST: A distributed vehicular broadcast protocol for vehicular ad hoc networks," *IEEE Wireless Communications*, vol. 17, no. 2, pp. 47–57, Apr. 2010.

[143] O. Abumansoor, A. Boukerche, B. Landfeldt, and S. Samarah, "Privacy Preserving Neighborhood Awareness in Vehicular Ad Hoc Networks," in *14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2011), 7th ACM Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet 2011)*, Miami, FL: ACM, Jul. 2011, pp. 17–20.

[144] F. J. Ros, P. M. Ruiz, and I. Stojmenovic, "Acknowledgment-Based Broadcast Protocol for Reliable and Efficient Data Dissemination in Vehicular Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 1, pp. 33–46, Jan. 2012.

[145] K. Lee, J. Haerri, U. Lee, and M. Gerla, "Enhanced Perimeter Routing for Geographic Forwarding Protocols in Urban Vehicular Scenarios," in *IEEE Global Telecommunications Conference (GLOBECOM 2007), 2nd IEEE Workshop on Automotive Networking and Applications (AutoNet 2007)*, Washington, DC: IEEE, Nov. 2007, pp. 1–10.

[146] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating Sets and Neighbor Elimination-Based Broadcasting Algorithms in Wireless Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 1, pp. 14–25, Jan. 2002.

[147] A. Khan, I. Stojmenovic, and N. Zaguia, "Parameterless broadcasting in static to highly mobile wireless ad hoc, sensor and actuator networks," in *22nd IEEE International Conference on Advanced Information Networking and Applications (AINA 2008)*, Okinawa, Japan, Mar. 2008, pp. 620–627.

[148] H. A. Omar, W. Zhuang, and L. Li, "VeMAC: A TDMA-Based MAC Protocol for Reliable Broadcast in VANETs," *IEEE Transactions on Mobile Computing*, vol. 12, no. 9, pp. 1724–1736, Sep. 2013.

[149] J. Wu and H. Li, "On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks," in *3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM 1999)*, Seattle, WA: ACM, Aug. 1999, pp. 7–14.

[150] Y.-T. Yu, M. Gerla, and M. Y. Sanadidi, "Scalable VANET Content Routing Using Hierarchical Bloom Filters," *Wiley Wireless Communications and Mobile Computing*, vol. 15, no. 6, 1001–1014, Apr. 2015.

[151] A. Bujari, "A Network Coverage Algorithm for Message Broadcast in Vehicular Networks," *ACM/Springer Mobile Networks and Applications (MONET)*, Apr. 2016.

[152] S. Duquennoy, O. Landsiedel, and T. Voigt, "Let the tree Bloom: Scalable Opportunistic Routing with ORPL," in *11th ACM Conference on Embedded Networked Sensor Systems (SenSys 2013)*, Rome, Italy: ACM, Nov. 2013.

[153] K. Lin and P. Levis, "Data Discovery and Dissemination with DIP," in *6th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN 2008)*, St. Louis, MO: IEEE, Apr. 2008, pp. 433–444.

[154] A. Reinhardt, O. Morar, S. Santini, S. Zöller, and R. Steinmetz, "CBFR: Bloom Filter Routing with Gradual Forgetting for Tree-structured Wireless Sensor Networks with Mobile Nodes," in *13th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2012)*, San Francisco, CA: IEEE, Jun. 2012.

[155] P.-H. Hsiao, "Geographical Region Summary Service for Geographical Routing," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 4, pp. 25–39, Oct. 2001.

[156] K. J. O'Dwyer and D. Malone, "Bitcoin Mining and its Energy Footprint," in *25th IET Irish Signals & Systems Conference (ISSC 2014)*, Limerick, Ireland: IET, Jun. 2014.

[157] A. Kirsch and M. Mitzenmacher, "Less hashing, same performance: Building a better Bloom filter," *Wiley Random Structures & Algorithms*, vol. 33, no. 2, pp. 187–218, May 2008.

[158] M. Gerla and L. Kleinrock, "Vehicular networks and the future of the mobile internet," *Elsevier Computer Networks*, vol. 55, no. 2, pp. 457–469, Feb. 2011.

[159] SAE International DSRC Committee, "DSRC Message Communication Minimum Performance Requirements: Basic Safety Message for Vehicle Safety Applications," SAE, Draft Std. J2945.1 Revision 2.2, Apr. 2011.

[160] R. K. Schmidt, T. Leinmüller, E. Schoch, F. Kargl, and G. Schäfer, "Exploration of Adaptive Beaconing for Efficient Intervehicle Safety Communication," *IEEE Network Magazine*, vol. 24, no. 1, pp. 14–19, Jan. 2010.

[161] M. van Eenennaam, W. Wolterink, G. Karagiannis, and G. Heijenk, "Exploring the solution space of beaconing in VANETs," in *1st IEEE Vehicular Networking Conference (VNC 2009)*, Tokyo, Japan: IEEE, Oct. 2009.

[162] R. S. Schwartz, A. E. Ohazulike, and H. Scholten, "Achieving Data Utility Fairness in Periodic Dissemination for VANETs," in *75st IEEE Vehicular Technology Conference (VTC2012-Spring)*, Yokohama, Japan: IEEE, May 2012.

[163] W. Cheng, X. Cheng, T. Znati, X. Lu, and Z. Lu, "The Complexity of Channel Scheduling in Multi-Radio Multi-Channel Wireless Networks," in *28th IEEE Conference on Computer Communications (INFOCOM 2009)*, Rio de Janeiro, Brazil, Apr. 2009, pp. 1512–1520.

[164] P. Kyasanur and N. H. Vaidya, "Capacity of Multi-Channel Wireless Networks: Impact of Number of Channels and Interfaces," in *11th ACM International Conference on Mobile Computing and Networking (MobiCom 2005)*, Cologne, Germany: ACM, Aug. 2005, pp. 43–57.

[165] L. Liu, W. Xia, and L. Shen, "An Adaptive Multi-Channel MAC Protocol with Dynamic Interval Division in Vehicular Environment," in *1st International Conference on Information Science and Engineering (ICISE), 2009*, Nanjing: IEEE, Dec. 2009, pp. 2534–2537.

[166] N. Lu, Y. Ji, F. Liu, and X. Wang, "A Dedicated Multi-Channel MAC Protocol Design for VANET with Adaptive Broadcasting," in *IEEE Wireless Communications and Networking Conference (WCNC)*, Sydney, Australia: IEEE, Apr. 2010.

[167] Q. Wang, S. Leng, H. Fu, and Y. Zhang, "An IEEE 802.11p-Based Multichannel MAC Scheme With Channel Coordination for Vehicular Ad Hoc Networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 449–458, Jun. 2012.

[168] L. De Martini and J. Härri, "Short paper: Design and evaluation of a multichannel mechanism for vehicular service management at 5.9GHz," in *5th IEEE Vehicular Networking Conference (VNC 2013)*, Boston, MA: IEEE, Dec. 2013, pp. 178–181.

Acknowledgments

I want to thank all my colleagues from the CCS-Team whom I worked with during the last few years. In particular those which supported my since beginning of my research activities.

Further, I want to thank my family and friends, especially my parents Brigitte and Sebastian, for encouraging me through my whole studies and supporting me all the years of my education.

Most importantly, I would like to say a big thank you to my wife Sandra. Without your great support, Sandra, this PhD thesis would not have been possible.

Thanks again to all of you!

Danksagung

Ich möchte allen meinen Kollegen vom CCS-Team danken, mit denen ich in den letzten Jahren zusammenarbeiten durfte. Besonders jenen, die mir schon seit Beginn meiner Forschungsarbeiten mit Rat und Tat zur Seite gestanden sind.

Weiteres möchte ich mich bei meiner Familie, meinen Verwandten und meinen Freunden bedanken, im Speziellen bei meinen Eltern Brigitte und Sebastian, die mich in den Jahren der Ausbildung und während meiner Studienzeit tatkräftig unterstützt haben.

Als wichtigste Person möchte ich hier meine Frau Sandra erwähnen. Ohne Deine Unterstützung, Sandra, wäre diese Dissertation nicht möglich gewesen.

Danke euch allen!