Content Location in Sparse Vehicular Ad-Hoc Networks

—

Feasibility, Implementation, Optimization on the Case of a City Scenario

Inaugural-Dissertation

zur
Erlangung des Doktorgrades der
Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Christian Wewetzer

aus Braunschweig

März 2009

Aus dem Institut für Informatik
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Referent:	Prof. Dr. Martin Mauve Heinrich-Heine-Universität Düsseldorf
Koreferent:	Prof. Dr. Michael Schöttner Heinrich-Heine-Universität Düsseldorf

Tag der mündlichen Prüfung: 5.5.2009

Abstract

Vehicular ad-hoc networks (VANETs) have many applications improving road safety, traffic efficiency, and driving experience. A commonality of all VANET applications is that they rely on information exchanged among VANET nodes. What differs is the communication paradigm underlying these applications: required information may be distributed proactively (push-based communication), or it may be requested on demand (pull-based communication). This thesis is concerned with a special problem in pull-based communication, where a vehicle in need of certain content is unaware of the existence and location of a suitable information source. To tackle this problem, we propose and study distributed services allowing for content location in VANETs.

As a starting point, we look for existing solutions to the search problem that have been proposed in other domains. We identify flooding and hash tables—distributed either among nodes or over geographical regions—as representative search schemes and introduce mathematical models that allow for a calculation of the search latency and success rate of these schemes in VANETs. The models combine random variables of information propagation speed and information propagation distance. Unfortunately, the expected success rates are low in our sample scenario even for flooding-based search, motivating us to introduce a network of base stations at the most frequented road intersections. The base stations improve communication in the VANET and allow content search with high success rate. Nevertheless, flooding as search approach with the highest success rate causes very high network load; index-based methods cause less network load but have a lower success rate. A goal of this thesis is to investigate whether the index-based schemes can be improved to achieve similar success rates as flooding and whether the performance of our theoretical models can be achieved at all in practice. To answer these questions, we design, implement, and evaluate an index-based content location service that is tailored to infrastructure-supported VANETs.

As basis of the network layer supporting the content location service, we first define an underlying routing algorithm. The algorithm is a position-based greedy algorithm extended to combat specific routing challenges occuring in infrastructure-supported

VANETs. Simulative evaluation shows that the algorithm performs close to optimal in the considered sample scenario.

In the next step we design and implement two search schemes. The first scheme relies on an index that is distributed among interlinked base stations, thereby differing from the theoretically modeled index-based search methods. Vehicles periodically register their search keys at the next available base station. Base stations are often involved in communication between distant vehicles, thus the cost in querying the search index is low. The performance of search with help of such an index can thus be regarded as best case performance of index-based search in infrastructure-supported VANETs. The second scheme is a flooding scheme that includes message suppression strategies to save network load but that nevertheless achieves success rates close to that of the theoretical model of flooding if parameterized correctly. We implement and optimize both search schemes in a parameter study in the sample scenario. With the best parameter sets, index-based search achieves a slightly lower success rate than flooding but saves a significant amount of network load. The implemented distribution of the index among base stations results in a latency that is lower than that of the theoretical models, where the index is distributed among vehicles. But due to imperfect routing and aging position information, our implemented scheme cannot achieve the success rate of the theoretical models. The performance of the implemented flooding scheme is close to that of our theoretical model, supporting its validity.

After showing that the proposed index-based scheme is a good practical choice for content search in infrastructure-supported VANETs, we are concerned with the network load required for maintenance of such a decentralized index. In commonly used inverted indices, contributing nodes calculate an identifier in form of a hash value for each information item and register (identifier, keyword set)-pairs in the search index. For better space-efficiency, a well-defined description scheme of information items could be established, so that nodes can compute the description of an information item they are interested in; it then suffices to register only the hash key of this description. Our goal is to reduce the amount of data required for this key. As first idea, nodes could simply resort to sending shorter hash keys. The downside of such short hash keys is an increased chance of false-positive look ups in the search index: if a looked-up information item is not registered but has the same hash key as a registered item, other content than intended is requested. As alternative, we propose the encoding of each nodes' information items into Bloom filters because these—if compressed—achieve space-optimality. We prove that lists of hash-keys are always less space-efficient than equivalent compressed Bloom filters and that there is a break-even point from which on these lists of hash-keys

are also less space-efficient than uncompressed Bloom filters. Furthermore, we propose a local opportunistic aggregation scheme in which vicinal nodes build a common Bloom filter, leading to an even lower false-positive rate of the search index. Lastly, we demonstrate the effectiveness of Bloom filters and of the aggregation scheme in a simulation study.

Zusammenfassung

Spontane drahtlose Kommunikationsnetze zwischen Fahrzeugen—sogenannte Vehicular Ad-Hoc Networks (VANETs)—ermöglichen vielfältige Anwendungen zur Verbesserung der Verkehrssicherheit, der Verkehrseffizienz und des Fahrerlebnisses. Eine Gemeinsamkeit dieser Anwendungen ist, dass sie auf Informationen basieren, die zwischen den Netzknoten des VANET ausgetauscht werden. Bei genauerem Betrachten unterscheiden sich diese Anwendungen allerdings im zugrundeliegenden Kommunikationsparadigma: Anwendungen können benötigte Informationen proaktiv an andere Netzknoten im VANET verteilen (Push-basierte Kommunikation) oder können Informationen erst dann von anderen Netzteilnehmern anfragen, wenn sie benötigt werden (Pull-basierte Kommunikation). Diese Arbeit betrachtet ein spezielles Problem der Pull-basierten Kommunikation, bei dem ein Fahrzeug mit Interesse an einer bestimmten Information nicht weiss, ob und wo eine geeignete Informationsquelle existiert. Um diesem Problem zu begegnen, wird in dieser Arbeit ein verteilter Dienst entwickelt, der die Suche nach Informationen im VANET ermöglicht.

Zum Einstieg wird zunächst in verwandten Forschungsgebieten nach Lösungsansätzen für das Suchproblem recherchiert. Als repräsentative Ansätze ergeben sich Fluten und Hash-Tabellen, welche über die einzelnen Netzknoten oder über geographische Gebiete verteilt sein können. Zur Beurteilung der Eignung dieser Ansätze für VANETs werden mathematische Modelle entwickelt, die eine Berechnung der Sucherfolgsquote und Suchlatenz ermöglichen. Die Modelle kombinieren Zufallsvariablen von Informationsausbreitungsgeschwindigkeit und Informationsausbreitungsdistanz. Für das betrachtete Beispielszenario ergeben sich selbst für Fluten nur geringe Erfolgsaussichten, was zu der Idee führt, die Kommunikation im VANET durch ein Netzwerk von Funkstationen—platziert an den meistbefahrenen Kreuzungen im Szenario—zu unterstützen. Dies führt zu deutlich verbesserten Erfolgsaussichten bei der Informationssuche. Dennoch haben die betrachteten Verfahren Nachteile: Fluten bietet die größten Erfolgschancen, erzeugt aber eine hohe Netzlast. Indexbasierte Ansätze verursachen zwar eine niedrigere Netzlast, dafür sind ihre Erfolgsaussichten deutlich geringer als die der Suche per Fluten.

Es ergibt sich die Frage, ob die indexbasierten Verfahren derart verbessert werden können, dass ihre Erfolgsrate mit der von Fluten vergleichbar ist und ob die anhand der theoretischen Modelle errechneten Leistungswerte in einem realen System überhaupt erreicht werden können. Um diese Fragen zu beantworten, ist ein Ziel der Arbeit die Implementierung und Evaluation eines für infrastrukturgestützte VANETs optimierten, indexbasierten Suchverfahrens.

Als Grundlage für die unterhalb des Suchverfahrens benötigte Netzwerkschicht wird jedoch zuerst ein Routingalgorithmus definiert. Der Algorithmus ist positionsbasiert und arbeitet nach dem Greedy-Prinzip. Er besitzt Erweiterungen, die den speziellen Herausforderungen in infrastrukturgestützten VANETs begegnen. Die Evaluation des Algorithmus zeigt, dass er im betrachteten Beispielszenario nahezu optimal arbeitet.

Im nächsten Schritt werden zwei Suchverfahren konzipiert und implementiert. Das erste Suchverfahren basiert auf einem Index, der über die miteinander vernetzten Funkstationen verteilt ist, wodurch er sich von den zuvor theoretisch modellierten indexbasierten Lösungen unterscheidet. Fahrzeuge registrieren ihre Informationen regelmäßig an ihrer nächstgelegenen Funkstation, von wo aus sie jeweils an die gemäß eines Verteilungsschemas zuständige Funkstation weitergeleitet werden. Funkstationen sind für die Speicherung des Suchindex ideal, weil ihre Positionen fest und sie immer im Netz verfügbar sind. Darüber hinaus beinhaltet ein effizienter Kommunikationspfad zwischen weiter entfernten Fahrzeugen in den meisten Fällen sowieso eine Funkstation, weswegen der zu erwartende zusätzliche Kommunikationsaufwand für die Abfrage des Suchindex niedrig ist. Die Leistungsfähigkeit eines solchen Suchindex ist daher repräsentativ für den “Best Case” der indexbasierten Suche in infrastrukturgestützten VANETs. Das zweite Verfahren basiert auf Fluten und enthält Strategien zur Vermeidung unnötiger Nachrichtensendevorgänge mit dem Ziel der Netzlastreduktion. Korrekt konfiguriert erreicht das Verfahren dennoch die Erfolgsquote des theoretischen Modells einer Suche basiert auf Fluten; somit kann dessen Sucherfolgsquote als Referenz für das erste Verfahren dienen. Die beiden Suchverfahren werden mittels einer Parameterstudie für ein Beispielszenario konfiguriert und miteinander verglichen. In der besten untersuchten Konfiguration erreicht das indexbasierte Verfahren eine etwas niedrigere Erfolgsquote als Fluten, spart dabei aber einen großen Anteil an Netzlast ein. Die Verteilung des Index über Basisstationen führt zu einer deutlich geringeren Suchlatenz als die in den theoretischen Modellen betrachtete Verteilung über Fahrzeuge. Das implementierte indexbasierte Verfahren kann aber aufgrund des nicht perfekten Routingalgorithmus und alternder Positionsinformationen der beteiligten Fahrzeuge nicht ganz die Erfolgsquoten

der theoretischen Modelle erzielen. Die für Fluten ermittelten Leistungswerte sind nah an denen des theoretischen Modells und stützen damit dessen Validität.

Nachdem gezeigt wurde, dass das indexbasierte Verfahren in der Praxis eine gute Lösung des Suchproblems in infrastrukturgestützten VANETs bietet, wird abschließend die Netzlast für die Verwaltung eines Suchindex betrachtet. In den üblicherweise verwendeten inversen Indizes berechnen die Netzknoten für jede Information einen Schlüssel in Form eines Hashwertes und registrieren (Schlüssel, Stichwortliste)-Paare im Suchindex. Um die Größe dieser Registrierungsinformationen zu reduzieren, kann ein wohldefiniertes Beschreibungsschema für Informationen eingeführt werden, welches es an einer Information interessierten Knoten erlaubt, die entsprechende Beschreibung zu bestimmen; dann reicht es aus, wenn Informationsquellen nur die Hashwerte ihrer Informationsbeschreibungen registrieren. In dieser Arbeit wird untersucht, wie die bei der Übertragung der Informationsbeschreibungen aufkommende Datenmenge reduziert werden kann. Ein erster Ansatz ist, dass die Netzknoten einfach kürzere Hashwerte versenden. Der Nachteil dieses Ansatzes ist eine erhöhte Wahrscheinlichkeit von falschen Treffern bei der Abfrage des Suchindex: wenn eine angefragte Information nicht registriert ist, aber den gleichen Hashwert wie eine registrierte Information besitzt, wird eine falsche Information angefordert. Als alternative Lösungsmöglichkeit wird in dieser Arbeit die Kodierung der Informationen in Bloom-Filter vorgeschlagen, weil diese in komprimierter Form die theoretisch minimale Größe annehmen. Es wird bewiesen, dass Hashwerte bei gleicher Wahrscheinlichkeit falscher Treffer immer mehr Platz als komprimierte Bloom-Filter benötigen, und gezeigt, dass es einen Punkt gibt, ab dem Hashwerte ebenfalls mehr Platz als unkomprimierte Bloom-Filter benötigen. Darüber hinaus wird ein lokales, opportunistisches Aggregationsverfahren vorgeschlagen, in welchem benachbarte Netzknoten ihre Informationen in einem gemeinsamen Bloom-Filter registrieren. Dies führt zu einer weiteren Reduktion der Wahrscheinlichkeit falscher Treffer im Suchindex. Abschließend wird die Effizienz der Bloom-Filter-basierten Registrierung und der Aggregation in einer Simulationsstudie überprüft.

Acknowledgments

This thesis is the outcome of my research as a member of the industrial Ph. D. programme of Volkswagen Group, Wolfsburg. I am very grateful for having the opportunity to work in this environment of industrial research. Completing a doctoral degree is very challenging, and I owe thanks to many people that supported me on my way.

First of all, I would like to express my deep gratitude to my advisor, Prof. Dr. Martin Mauve. He agreed to be my advisor when I only had some nebulous ideas in my pocket. He open-mindedly motivated and supported me in shaping my research topic and in every further step of my work. He was available for discussions and feedback on very short notice, and I left each of our conversations with the feeling that I learned something. I also thank Prof. Dr. Michael Schöttner for agreeing to be referee for this thesis.

In the initial phase of my research, I quickly made contact with the other gifted researchers forming the Computer Networks Research Group at the Heinrich Heine University of Düsseldorf. I would like to thank all group members for our lively and fruitful discussions. I am especially grateful to Jun.-Prof. Dr. Björn Scheuermann for his assistance with some challenging mathematical parts of this thesis as well as for his inspiration and collaboration on the probabilistic search indices ("wouldn't it be better to use short hash values?"). Furthermore, I am thankful to Dr. Christian Lochert for insights on routing as well as for his help with the simulator interlinking environment. I thank Dr. Wolfgang Kiess for initial conversations on thesis writing and for his critical paper review.

I am very grateful for the support I received from my colleagues at the Volkswagen Driver Information Systems Research department. My boss Dr. Bernd Rech supported me unbureaucratically and in every possible way. I had countless discussions with my mentor, Dr. Andreas Lübke, whom I thank for his ideas and comments; his positivity encouraged me to push forward my research. I further thank Dr. Murat Caliskan for intense fundamental conversations especially throughout the initial phase of my research, for his critical reviews, and for introducing me to my advisor—helping me to pave the path of my work. At a later stage of my research, I had the luck to meet a new

colleague, Dr. Gregor Gärtner, whom I was happy to find interested in my work. His positive attitude motivated me, and he taught me much about problem solving and scientific writing. Furthermore, I acknowledge Markus Kerper for proofreading parts of this thesis on very short notice. Finally, I thank my dear friend Stefan Neumann for conversations and caffeine whenever I needed them.

I could not have completed my thesis without the support and understanding of my family. When I came home from work, my girlfriend Sandra Schrader was there to put me back on track with unmatched affection and empathy. My families (the Wewetzers and the Schraders) encouraged[1] me in all my efforts and gave me confidence. I dedicate this thesis to my grandfathers Horst Wewetzer and Heinrich Rossel; I started my Ph. D. with the support of both of them—and hope they see me finishing it where they are now.

[1] Dad: I am not sure all this would have happened if you had not bought that C64.

Confidentiality Note

Publications about the content of this work require the written consent of Volkswagen AG.

The results, opinions and conclusions expressed in this thesis are not necessarily those of Volkswagen AG.

Contents

List of Figures

List of Tables

List of Abbreviations

ACK	Acknowledgment
AIFS	Arbitration Interframe Space
AODV	Ad-hoc On-demand Distance Vector Routing
BSS	Basic Service Set
BSSID	Basic Service Set ID
C2CCC	Car-to-Car Communication Consortium
CA	Collision Avoidance
CAN	Content-Addressable Network
CBF	Contention-Based Forwarding
CCK	Complementary Code Keying
CDF	Cumulative Distribution Function
CGGC	Cached Greedy Geocast
CSMA	Carrier Sense Multiple Access
CTS	Clear To Send
CW	Contention Window
DCF	Distributed Coordination Function
DECT	Digital Enhanced Cordless Telecommunication
DHT	Distributed Hash Table
DIFS	DCF Interframe Space
DSR	Dynamic Source Routing
DSSS	Direct-Sequence Spread Spectrum
DTN	Delay Tolerant Network
EDCA	Enhanced Distributed Channel Access
EDF	Empirical Distribution Function
EIRP	Effective Isotropic Radiated Power

ETSI	European Telecommunications Standards Institute
FH	Frequency Hopping
FIFO	First In, First Out
FPR	False-Positive Rate
GHT	Geographic Hash Table
GPCR	Greedy Perimeter Coordinator Routing
GPRS	Greedy Perimeter Stateless Routing
GPS	Global Positioning System
GSR	Geographic Source Routing
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IR	Infrared
ISM	Industrial Scientific Medical
ISO	International Standards Organization
LAN	Local Area Network
LLC	Logical Link Control
MAC	Medium Access Control
MANET	Mobile Ad-Hoc Network
OD	Origin-Destination
OFDM	Orthogonal Frequency Division Multiplex
OSI	Open Systems Interconnection
PCF	Point Coordination Function
PDF	Probability Density Function
RTS	Ready To Send
SAE	Society of Automotive Engineers
SIFS	Short Interframe Space
SMP	Short Message Protocol
TC ITS	Technical Committee on Intelligent Transportation Systems
TCP	Transmission Control Protocol
THL	Truncated Hash List

TTL	Time To Live
UDP	User Datagram Protocol
VADD	Vehicle-Assisted Data Delivery
VANET	Vehicular Ad-Hoc Network
WAVE	Wireless Access in Vehicular Environments
WLAN	Wireless Local Area Network
WWW	World Wide Web

Chapter 1

Introduction

Locating and retrieving content has always been one of the most frequent tasks of users in the World Wide Web (WWW). In the beginning of the WWW, users unaware of the address of a suitable web server could potentially browse the few web pages for specific content, but with the enormous growth of the web, such an approach is impractical today. Instead, users rely on results provided by search engines. In this thesis, we are concerned with the feasibility, implementation, and optimization of such a search engine for vehicular ad-hoc networks (VANETs). We envision that there will be a demand for such a search engine as the amount of data available in, and gathered by vehicles will grow, and so will the amount of such data harvesters in the VANET.

In the remainder of this chapter, we first motivate the need for a search engine in VANETs (Section 1.1). This is followed by the elaboration and the definition of the underlying search problem in the VANET context (Section 1.2). Section 1.3 concludes the introduction with a summary of the thesis contributions and a thesis outline.

1.1 Motivation

The emergence of VANETs enables many kinds of novel applications improving traffic safety, traffic efficiency, and driving experience. For example, vehicles involved in an accident may inform approaching vehicles, so that drivers are alerted of the incident. Vehicles may share information about travel times and parking spots, so that drivers may select the most efficient route to their destination. Communities may evolve in vehicular networks, with vehicles communicating to share content of interest or information about certain events.

The common part of all applications of VANETs is that they produce and consume information. But at a closer look, these applications differ in the way information is exchanged among vehicles. Consider as one example an accident scenario, where a vehicle detecting the accident immediately broadcasts a warning message since it intends to inform all nearby vehicles as fast as possible to avoid any further collisions. In other words, the generated warning message is *pushed* into the VANET. Push-based communication is reasonable here because the transmitted message is small and the contained information is of value for all nearby vehicles. On the other hand, there are cases when vehicles require data of interest to no (or only very few) other vehicles. As example of this, passengers of a vehicle might be interested in finding other vehicles with special properties, e.g. driving to the same destination or being a special vehicle like a tow truck or ambulance. Passengers might be interested in content stored in other vehicles, such as fuel consumption data or available media. Proactively broadcasting this content in a VANET would be inefficient; instead, it is more bandwidth-efficient if this type of content is requested on-demand. Content is then *pulled* from an information source. This thesis is concerned with such pull-based communication.

In pull-based communication, whenever a vehicle requires a certain information item, it has to contact a suitable information source. If the vehicle in need of information is aware of the location of an information source, it may send a message to that particular position. Otherwise, if the vehicle knows that an information source is located somewhere nearby, it may flood a request message within a certain area around its position. But, if the vehicle has no indication about the location—or even the existence—of a suitable information source, it first needs to locate one. If all vehicles in such a situation accomplished this by flooding requests into the whole VANET, the resulting network load would quickly congest the network and thereby degrade the performance of all applications on the wireless channel. To avoid this, this thesis proposes a mechanism for vehicles to efficiently locate and query information sources in VANETs.

1.2 Problem Definition: Search in VANETs

To come to a precise problem definition, we review existing definitions of the term *search engine* in the Internet domain. We afterwards interpret these results in the context of VANETs, breaking the search process into single steps. We then state which of the steps are addressed in this thesis.

According to webopedia[1], a search engine is

> *a (potentially distributed) algorithm mapping keys to information sources*

or

> *a program that searches documents for specified keywords and returns a list of the documents where the keywords were found.*

Answers.com[2] defines a search engine as

> *a software program that searches a database and gathers and reports information that contains or is related to specified terms*

or (related to the Internet) as

> *a website whose primary function is providing a search engine for gathering and reporting information available on the Internet or a portion of the Internet.*

Searchengineshowdown[3] offers

> *While the term more properly refers to any software used to search any database, on the Internet the phrase usually refers to the very large databases of Web sites that are automatically built by robots. These Internet search engines use a software robot (or spider) that seeks out and index the words on Web pages.*

In short, a Web search engine is a (possibly distributed) algorithm that looks up *information sources* owning an *information item* associated with a certain search *key*. This *lookup* requires a (possibly distributed) data base of key-to-information source ID mappings (in the Internet, the most common form of IDs are IP addresses), called *search index*. The data base may either be created by the search engine itself (using *robots*) or it may be created by information sources actively registering their locally held keys (*content registration*). To initiate a lookup, a node (the *originator*) sends a *search request* for a set of keys to the search engine. Based on the returned lookup results, the originator may send one or more *content requests*. Knowing the IP address of the target node, each node receiving the content request is able to route it with help of its routing table. When an information source receives a content request, it transfers the requested content to the originator (*content retrieval*).

[1] http://www.webopedia.com/TERM/s/search_engine.html
[2] http://www.answers.com/topic/search-engine
[3] http://www.searchengineshowdown.com/defs/se.html

In contrast to the Internet, the most widespread routing methods in VANETs are position-based, as the highly dynamic network topology makes maintenance of routing tables over several hops impractical [LW07]. This way, vehicles do not make routing decisions based on the IDs of the intended targets, but based on their relative geographical location to other vehicles. Consequently, if a vehicle wants to request content, it must know the location of a suitable information source. As vehicles are mobile and move away from their registered position, the ID of the information source is required in addition to its location information to choose the right vehicle among those near the target position.

There are two principal design options for a VANET service allowing vehicles to obtain ID and location of a search key. The first option is to set up two separate services that are consecutively queried by the originator: one service returning IDs of information sources for a given key, the other service returning a location for a given ID. The second option is to set up a single service directly mapping keys to (ID, location)-tuples.

Regarding the advantages and disadvantages of these design options, the most important point is that communication in a sparse VANET is heavily influenced by delays. Furthermore, vehicles are often reachable only for a short amount of time. Consequently, the whole search process should be designed for minimum latency. This clearly favors the single-service approach, as less messages have to be consecutively exchanged than with separate services. Thus, in this thesis we aim for the single-service solution: a VANET search engine looking up (ID, location)-tuples for given search keys. What is more, due to the communication latency in the VANET, in this thesis we assume that the search engine itself issues a content request for the looked-up key. The reason is that this involves less communication than the alternative of returning the tuple list to the originator, and having the originator issue a content request afterwards.

Accordingly, we define the problem to be solved by a VANET search engine: given a key h, look up a tuple (S_h, L_{S_h}), where S_h and L_{S_h} are the ID and the location of the information source, and request the corresponding content from S_h.

This thesis is concerned with three questions related to such a search engine:

- Given a VANET scenario, what is the theoretically achievable success rate and what is the latency of search?
- How close can search schemes come to this bound in practice?
- How can nodes bandwidth-efficiently register content in a search index?

This thesis is not concerned with the decision which out of many available information sources should be queried. In our evaluation, we ensure that there is at least one information source available for a requested key; the search engine issues content requests to each available source. Furthermore, this thesis does not investigate efficient content retrieval. Whenever a vehicle replies to the content request of another vehicle in our evaluations, the answer is just one small data packet to see whether any content can be transferred at all.

1.3 Thesis Contributions and Outline

This thesis has three main contributions, investigating the above-mentioned questions.

Our first contribution is a feasibility study of search in sparse VANETs. We derive stochastic models for search success rate and latency of existing search methods and combine these with a simulative evaluation of information propagation speed in a sample scenario. The information propagation speed is determined with help of a flooding scheme that achieves best possible success rate and lowest communication delay as long as there is no congestion on the medium. The feasibility study results in a ranking of existing ideas, an upper bound of the achievable success rate, and a lower bound of the achievable latency for the considered scenario. Furthermore, the study motivates the need for infrastructure support to allow for satisfactory search success rates even in sparse VANETs.

Our second contribution is the conceptuation, implementation and parameter study of an infrastructure-supported search scheme. The ideas underlying the search engine result from the feasibility study. To have a reference for the search scheme, the flooding scheme defined in the feasibility study is extended to be more bandwidth-efficient, while maintaining the same success rate and latency. The parameter study shows the sensitivity of the search methods to their individual parameters as well as their relative performance. Furthermore, it allows for a comparison with our theoretical results.

Our third contribution is a bandwidth-efficient content registration algorithm. In the initial concept of our search engine, vehicles register hash keys of their content in the search index. To save bandwidth, vehicles could register shorter hash values at the risk of more false-positive lookups. Instead, we propose that each vehicle encodes its registration information into a compressed Bloom filter, because the bandwidth required to transmit such a filter is minimal in theory. As part of this contribution, we prove

the better bandwidth efficiency of compressed and uncompressed Bloom filters in comparison with hash keys. Finally, we introduce an aggregation scheme where vehicles cooperatively build a common Bloom filter, leading to a lower false-positive rate of the index compared to atomic filters. The Bloom filter registration scheme is again evaluated in a simulation study, underlining its feasibility and bandwidth-efficiency when compared with standard hash values.

This thesis is structured as follows. In Chapter 2, the technical aspects and deployment scenarios of VANETs are presented. Chapter 3 outlines the simulation environment and scenario that are used in subsequent evaluations. In Chapter 4, a feasibility study of search in sparse VANETs is conducted. The results of this chapter, together with the routing algorithm proposed in Chapter 5, are the groundwork for design, implementation, and evaluation of a search engine in infrastructure-supported VANETs (Chapter 6). Bandwidth-efficient content registration is investigated in Chapter 7. Chapter 8 concludes this thesis with a summary of results and suggestions for future work.

Chapter 2

VANETs

This chapter is devoted to VANETs as underlying communication technology of this thesis. The first part is an outline of fundamental technical aspects of VANETs (Section 2.1). The second part (Section 2.2) briefly discusses what network densities can be expected after market introduction of VANETs.

2.1 Technical Aspects

The worldwide harmonization and standardization process of VANETs is still ongoing, and the final technical aspects have not yet been agreed upon. Nevertheless, there is a strong trend indicating that communication in VANETs will be based on an adaption of the IEEE 802.11 (wireless local area network) standard [CSLSC97] for vehicular environments in a communication architecture that allows for IP-based and non-IP-based communication.

This section has a layer-oriented bottom-up structure. The well-known ISO/OSI model [ITU94] (Figure 2.1) has been developed as a basis for the definition of communication standards and here serves as a starting point for introduction of VANET communication architectures. In classical layer-oriented approaches following the ISO/OSI model, newly created messages are handed downwards, whereby each layer adds some content to the message it received from the layer above. When processing a received message, these contents are successively removed from bottom to top, until only the application layer data remains. Higher layers have no access to information appended and removed by lower layers; for example, applications cannot access position information if exchanged at network layer level. Consequently, if such information is required in multiple layers, it has to be included in each respective message part. This redundant encoding leads to overhead in the resulting packet.

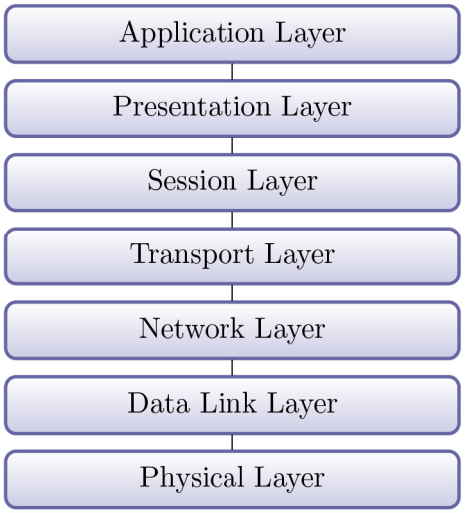


Figure 2.1: ISO/OSI layer model [ITU94]

Turning to VANETs, additional overhead in packets is critical: first, wireless networks generally suffer from low bandwidth; second, relative speeds of vehicles are high and the communication range is only in the order of a few hundred meters, consequently the available communication time between vehicles is low; third, the longer the transmitted packet, the higher is the likeliness of an erroneous transmission. Thus, more lightweight (cross-layer) approaches with fast media access are preferable. In this context, several projects and gremia have proposed communication architectures for VANETs that permit lightweight communication. Here, we focus on the architectures defined in IEEE 1609 [CSLSC07a] and by the Car-to-Car Communication Consortium (C2CCC)[1] as representative and well-known examples.

Figure 2.2 shows the Wireless Access in Vehicular Environment (WAVE) protocol stack. WAVE is a set of standards that is developed by IEEE. The standards define architecture, communication model, management functions (such as security), and physical layer access. While the protocol stack allows for connection-oriented communication based on TCP/IP, it introduces the WAVE short message protocol (SMP), enabling lightweight message exchange as an alternative to IP-based communication. The management plane provides cross-layer information and functionality such as application registration, channel usage monitoring, and a message information base (MIB).

Figure 2.3 shows the architecture as envisioned by the C2CCC. Similar to the IEEE

[1] industrial consortium initiated by automotive manufacturers, later joined by suppliers and research institutions, http://www.car-to-car.org

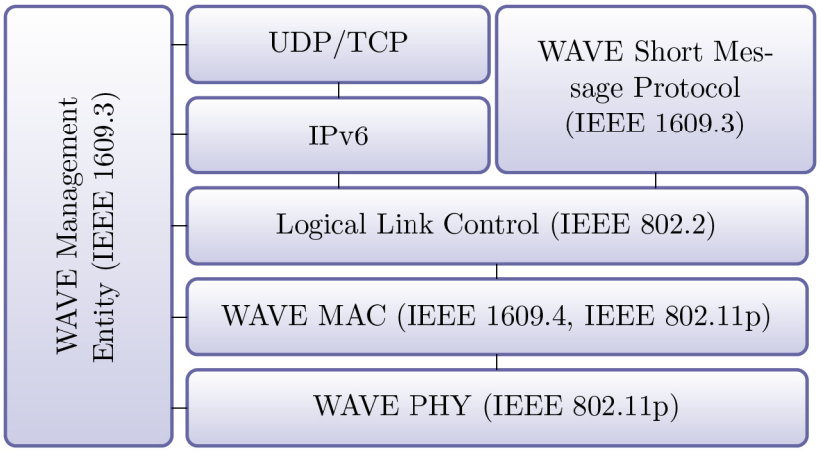


Figure 2.2: WAVE protocol stack (without application layer) [CSLSC07a], consisting of two planes: Management Plane on the left and Data Plane on the right

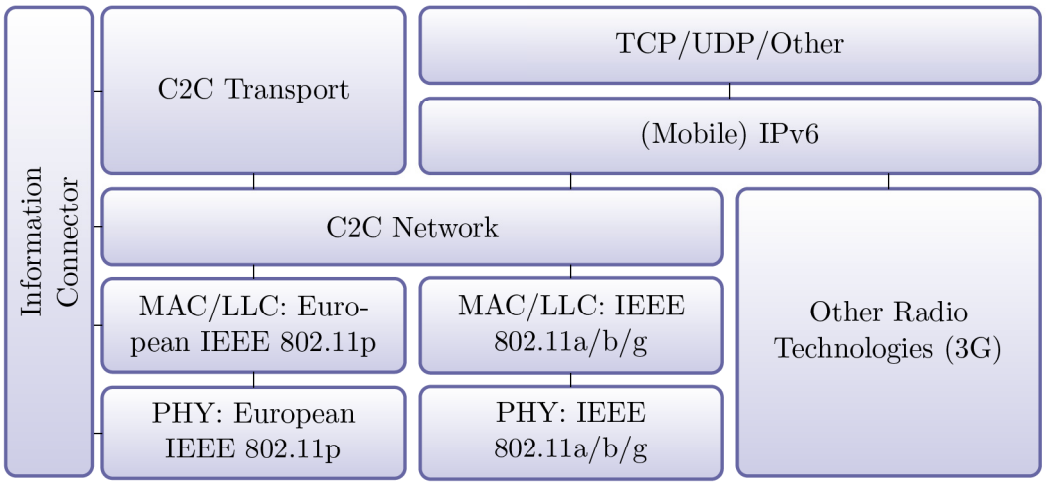


Figure 2.3: C2CCC protocol architecture (without application layer). Allows for IP and non-IP-based communication. Includes cellular networks in addition to 802.11-based systems.

1609 architecture, it allows for non-IP-based communication. IP packets may be sent through the C2C Network and in other radio networks, i.e. cellular networks as additional communication technology. The information connector on the left hand side of the drawing is also similar to the management entity of the IEEE 1609 variant: it provides information that is commonly available to all layers, e.g. position information for use in routing (C2C Network) as well as on the application layer.

Each architecture layer will now be discussed in a separate section, with the exception of the LLC; currently no necessity of deviations from the IEEE 802.2 LLC for wired LAN has been identified, thus it can be expected to behave as described in [CSLSC98].

2.1.1 Physical Layer

This section details the physical layer of the IEEE 802.11p standard [CSLSC08], which is evolving as the common physical layer of VANETs. The standard is an extension of the original IEEE 802.11 standard, whose main properties we briefly review in the following. Table 2.1 includes information about frequency spectrum, modulation, and achievable bit rates of the IEEE 802.11 legacy standard and some of its sub-standards.

Standard	Freq. band	Modulation	Supported bitrates	Max. EIRP (Ger.)
802.11	2.4 GHz	IR/FH/DSSS	up to 2 MBit/s	0.1 W
802.11a	5 GHz	OFDM	up to 54 MBit/s	1 W
802.11b	2.4 GHz	DSSS-CCK	up to 11 MBit/s	0.1 W
802.11g	2.4 GHz	OFDM	up to 54 MBit/s	0.1 W
802.11p	5.9 GHz	OFDM	up to 27/54* MBit/s	2 W

Table 2.1: Characteristics of IEEE 802.11 legacy standard and its sub-standards. (*) Maximum bitrate of 802.11p is 27 MBit/s on a 10 MHz channel, 54 MBit/s on a 20 MHz channel

The original IEEE 802.11 standard enables data rates of up to 2 MBit/s and includes three modulation schemes, namely infrared (IR), frequency hopping (FH) and direct-sequence spread spectrum (DSSS). The IEEE 802.11b standard [CSLSC99b] introduces complementary code keying (CCK) to extend the DSSS modulation defined in the original standard, thereby allowing data rates up to 11 MBit/s. The original standard uses the 2.4 GHz Industrial Scientific Medical (ISM) frequency band. This band is heavily used (e.g. by DECT phones, microwave ovens, Bluetooth devices), which negatively impacts IEEE 802.11 performance. This lead to the proposal of IEEE 802.11a [CSLSC99a] as an alternative, introducing an orthogonal frequency division multiplex (OFDM)-based

physical layer in the 5 GHz frequency band. The disadvantage of transmitting in a higher frequency band is that waves of this wavelength are more subject to attenuation and interference by radio obstacles. To compensate these losses, some countries permit higher effective isotropic radiated power (EIRP), e.g. 1 W in Germany. IEEE 802.11g [CSLSC03] introduces OFDM (and therefore higher data rates) in the 2.4 GHz band.

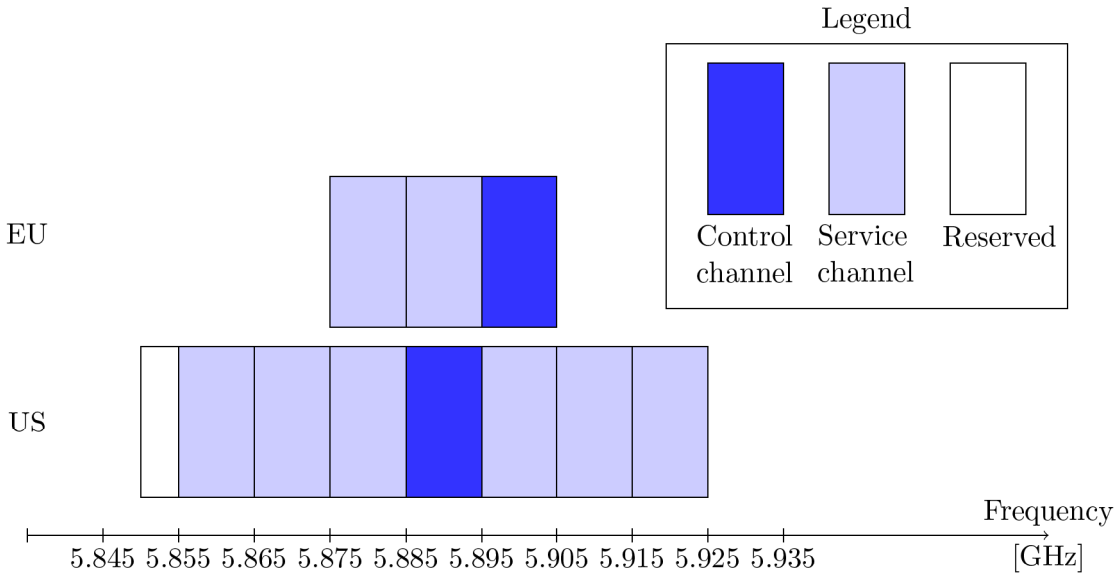


Figure 2.4: Designated frequencies for VANETs in Europe and the US (dark blue: control channel; light blue: service channel)

The legacy standard defines a channel bandwidth of 20 MHz. IEEE 802.11p supports channel bandwidths of 10 MHz and 20 MHz, using OFDM as modulation scheme. 10 MHz channels are introduced to reduce the OFDM inter-symbol interference due to multi-path propagation. In case of a 10 MHz channel, the maximum bitrate is 27 MBit/s; in case of a 20 MHz channel, this maximum bitrate doubles to 54 MBit/s. The standard is developed for use by vehicular safety and traffic efficiency applications, and thus requires a dedicated frequency band to minimize interference with other systems. The current status of frequency allocation for the European Union (EU) and the United States (US) is depicted in Figure 2.4. In the US, the 75 MHz frequency band between 5.85 GHz and 5.925 GHz has been allocated. In Europe, a 30 MHz frequency band has been allocated between 5.875 GHz and 5.905 GHz. The channel allocation scheme shows that the EU and US both subdivide the frequency band into one control channel and several service channels, each with a channel bandwidth of 10 MHz. The control channel

is used for safety messages and service announcements. Service data is exchanged on separate channels. The maximum allowed transmission power is 2 W in Europe, which partly compensates for the reduction of transmission range that results from transmitting in the 5.9 GHz frequency band instead of the 2.4 GHz band.

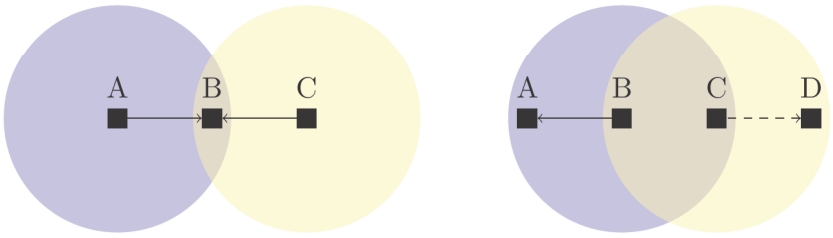
Furthermore, compared to the legacy standard, IEEE 802.11p sets higher requirements on receiver filtering capabilities: while receivers are listening to a transmission on one channel, they have to reject signals transferred on other channels more accurately.

2.1.2 MAC Layer

It is the task of the medium access control (MAC) layer to coordinate access to the shared radio channel between multiple nodes. The medium access protocol defined in the IEEE 802.11 standard is based on carrier sense multiple access (CSMA) [KT75]. The standard defines two MAC schemes: the point coordination function (PCF), which is used when an access point is available, and the distributed coordination function (DCF), which is used in ad-hoc networks. Since this thesis focuses on ad-hoc networks, only the DCF is detailed here. We start with an outline of the basic DCF. Subsequently, we point out two extensions of the DCF that are optionally available for unicast packets.

In the basic DCF, each node willing to transmit a packet first checks whether the channel is free (carrier sense) for a fixed period of time (DCF interframe space, DIFS). If it finds the channel free, it sends the packet. Otherwise, the node waits until the channel is free plus another DIFS. At this point in time, the node defers its transmission for a random multiple of the slot time (16 μs according to the standard) that lies between zero and the contention window (CW). This random number is called backoff timer. The backoff timer mechanism avoids simultaneous transmissions that would happen in case multiple nodes with packets to send were waiting for the end of the ongoing transmission. During the deferral time, the node checks whether the channel becomes busy. If this happens, it may not further decrement its backoff timer, but first has to wait for the end of the current transmission plus the DIFS. When the backoff timer expires, the node sends the packet. If the node had to defer its transmission, it now has wait for another random backoff period before sending the next packet.

There are two fundamental channel access problems in VANETs—and in mobile ad-hoc networks (MANETs) in general—that the basic DCF scheme cannot handle: the hidden node problem and the exposed node problem. Both are illustrated in Figure 2.5. The hidden node problem occurs when a node receives two simultaneous transmission from



(a) Hidden node problem: node A and C each assess the channel as free and transmit simultaneously, causing a collision at node B.

(b) Exposed node problem: node B sends to A, C notices a busy channel and unnecessarily defers a packet it wishes to send to node D.

Figure 2.5: Hidden node problem and exposed node problem

nodes that are outside each others radio range (in Figure 2.5(a), these are nodes A and C). The exposed node problem occurs when a node that is ready to send a packet notices that the channel is busy but its own packet is intended for a node not influenced by the ongoing transmission (in Figure 2.5(b), node D could still receive a packet from node C, although C senses a busy channel). The legacy standard does not combat the exposed node problem, but includes an extension of the DCF scheme that is able to alleviate the hidden node problem for unicast messages.

This optional extension for avoiding the hidden station problem is a ready to send/clear to send (RTS/CTS) scheme called *collision avoidance* (CSMA/CA): at the point where a node would usually transmit the packet itself, it instead sends an RTS message. After waiting for a short interframe space (SIFS) that is shorter than the DIFS, the intended receiver replies with a CTS message, if there is no other node (potentially outside the range of the node that sent the RTS message) from which it is receiving or will shortly receive a packet. This way, the receiving node may avoid a collision caused by receiving two simultaneous transmissions.

Another optional extension for unicast messages is the acknowledgment of successfully received packets by the receiver. Like CTS messages, acknowledgments are sent after deferring for a SIFS. If the sender of a unicast message does not receive such an acknowledgment, it doubles the CW and chooses a new random value for the backoff timer to repeat the transmission. This way, the retransmission mechanism—if successful—is capable of handling errors locally and concealing them from the network layer.

The IEEE standards 802.11, 802.11a, 802.11b, and 802.11g do not include priority concepts for data messages. Priorities play an important role in vehicular networks, because

safety applications may generate emergency messages; these should have preference over less important messages. Thus, the IEEE 802.11p standard additionally specifies a concept of message priorities and prioritized channel access. This scheme is similar to that of enhanced distributed channel access (EDCA), which is defined in the IEEE 802.11e standard [CSLSC05]. In EDCA, each message, according to its priority, is inserted into one out of four First In, First Out (FIFO) priority queues. The queues have different CWs, so that the queue with the highest priority—on average—has the lowest backoff timer. Furthermore, EDCA requires nodes not only to wait for the DIFS, but also for another arbitration interframe space (AIFS), before starting or resuming their backoff timers. The AIFS values are different for each queue, just like the sizes of the CWs. If the backoff timers of two queues expire simultaneously, the message waiting in the higher-priority queue is transmitted. In short, EDCA introduces a virtual internal contention in addition to the external contention for the channel. Moving from the standard EDCA to that of IEEE 802.11p, the latter differentiates between control channel and service channels; each channel has its set of four priority queues. The values of AIFS and CW are specified as part of the IEEE 1609.4 standard [CSLSC06].

Another key amendment of the IEEE 802.11p standard is the WAVE mode. The legacy standard requires nodes to setup membership of a common group of nodes called basic service set (BSS, or independent BSS in case of ad-hoc mode) before being able to exchange data. Taking into account the short time during which vehicles are within each others' radio range, this setup is too time-consuming in VANETs. It is eliminated with introduction of the IEEE 802.11p WAVE mode: a station in WAVE mode may send and receive packets using a wildcard BSSID, which the legacy standard restricted to management frames. Furthermore, IEEE 802.11p introduces the WAVE BSS that allows nodes to join a service just by overhearing the service beacon. While associated with a WAVE service, a node may still send and receive packets with the wildcard BSSID, which is important for safety applications.

2.1.3 Network Layer

While the lower layers enable hop-to-hop delivery of packets, end-to-end delivery of packets is the task of the network layer. This includes the fundamental function of routing: a packet received from the higher layers contains the address of the intended receiver(s); the network layer has to find a suitable path through the network.

Considering current standardization activities, the IEEE P1609.3 standard [CSLSC07b] describes a common network layer interface for vehicular networks, but it does not stan-

dardize a certain routing protocol. Similarly, the manifesto of the C2CCC [Con07] distinguishes four possible forwarding types (Geographical Unicast, Topologically-Scoped Broadcast, Geographical Broadcast, and Geographical Anycast), but likewise does not stipulate any specific routing protocol. We therefore move on to research results in this area.

The choice of a routing protocol strongly depends on the network type and the intended receiver. Flooding-based strategies are reasonable if a message is to be sent to all nodes in a certain region of the network. This is typically the case in vehicular safety applications, where nodes within a certain region are to be notified of an event [TM07, EGH+06].

In the most simple flooding scheme, a message contains the payload plus a time to live (TTL) field indicating the remaining period of time or number of hops it may be propagated. All nodes receiving this message decrement the TTL and broadcast the message until the TTL is zero. This scheme causes the well-known broadcast storm problem [TNCS02] because even nodes that have already sent the message resend it upon reception. These rebroadcasts can be avoided by introducing message identifiers and having nodes discard previously received messages. But even in that case, each node broadcasts the message once, which is often not required to distribute the message to all nodes. As further improvement, [TNCS02] suggests several rebroadcast suppression schemes. These are discussed in Section 6.1.

When a message has only one intended target (unicast), more bandwidth-efficient routing algorithms than flooding may be applied. Unicast routing is a well-researched topic in MANETs. In this research direction, Mauve et al. in [MWH01] classify routing algorithms into topology-based and position-based. Topology-based algorithms maintain and use information about existing links in the network. These algorithms are not suitable for VANETs, because the occurring links are highly dynamic [LW07]. Position-based routing strategies make forwarding decisions based on the geographical positions of nodes. In VANETs, most research is on position-based routing protocols. We review these separately in Chapter 5, where we outline a unicast and anycast routing algorithm for infrastructure-supported VANETs.

2.1.4 Application Layer

Applications for VANETs are usually subdivided into three types: traffic safety, traffic efficiency, and infotainment [Con07]. Each application type poses different requirements to the underlying communication system (see Table 2.2) [WCML07].

Application type	Network load	Latency	Comm. paradigm	Comm. distance
Traffic				
Safety	Low	Low	Push	Close
Efficiency	Moderate	Moderate	Push, Pull	Close/Medium
Infotainment				
Download	High	High	Pull	Varies
Streaming	High	Low	Pull	Varies

Table 2.2: Application types and their requirements regarding the communication system, derived from [WCML07]

Safety applications require fast and reliable information dissemination, on the other hand the network load caused by these applications is low. Nevertheless, many safety applications rely on information about positions and movements of vicinal vehicles; this information is obtained from periodic single-hop broadcast messages (beacons), causing significant channel load if disseminated at a high frequency [TM07]. The C2CCC has drafted a standard format for this message (Cooperative Awareness Message), which is currently being standardized within the European Telecommunications Standards Institute (ETSI) Technical Committee on Intelligent Transportation Systems (TC ITS). ETSI TC ITS has not yet defined the broadcast interval of this message, but current discussions indicate that the interval will be below one second. In the US, the Society of Automotive Engineers (SAE) has standardized a message set [DSR07] containing application layer messages such as the BasicSafetyMessage. Like the Cooperative Awareness Message, the BasicSafetyMessage contains the senders' position, movement vector, and status of several actors (e.g. blinker, brake pedal, headlight). Based on these periodically broadcast messages, safety applications create other types of messages in case an abnormal event is detected (push-based communication).

Traffic efficiency applications aim for an optimal use of road capacity. Applications of this type include information systems about traffic conditions [WER+03], parking spots [CGM06], as well as merging assistants [ALG+05], taxi dispatch systems [HHCW05], and tolling. The bandwidth required by such applications can be assumed to be moderate, noticing that information gathered by individual vehicles may be aggregated [LSM07, IW07]. While information exchanged by these applications is outdated after a short time, traffic efficiency applications still have lower requirements on the latency of the communication system than safety applications. Information required by these applications is typically exchanged proactively among all vehicles (push-based), which is reasonable because it is of interest to most nodes. Nevertheless, very specific information for efficiency applications, such as availability of a certain parking spot or

detailed fuel consumption data of a certain vehicle type, are only of interest to few vehicles and are thus better requested on demand.

Infotainment applications provide enhanced driving experience to driver and passengers. Such applications may be further divided into downloading and streaming applications. Streaming applications require continuous low latency-channel access, whereas downloading applications have weaker communication requirements. Examples of applications include point-of-interest notification, community-based applications, and multimedia, with the latter posing high bandwidth requirements on the VANET. Published applications include virtual flea markets [LPAG06] and in-vehicle advertising [NDZ+05]. Latest developments show a trend towards a highly customizable and individual infotainment: each driver or passenger has its own preferred type of infotainment, influenced e.g. by driving situation and age group. Consequently, information required by these applications cannot not be broadcast proactively, but should be requested on-demand (pull-based communication). In this direction, Fiore et al. [FCC05] have investigated a pull-based information-sharing application for VANETs. In their work, content is located via flooding.

Content search schemes as investigated by this thesis are a necessary building block for pull-based applications that require content whose existence or location is unknown: they allow to locate and query content within other vehicles. This thesis thereby focuses on efficient location and querying of the content; the transfer of large amounts of data is not investigated. For concepts on data transfer in VANETs, the reader is referred to [LPY+06, NDP+05].

2.2 Equipment Density in Real-World VANETs

In this section, expected equipment densities in real-world VANETs are discussed. The term *equipment density* originates from a paper by Lochert et al. [LSCM07], where it is defined as the *average number of equipped* (carrying a wireless radio unit) *vehicles per radio range of road*. We call VANETs with low equipment density *sparse*, and VANETs with high equipment density *dense*.

The equipment densities occurring during the first years of market introduction are determined by two factors, namely the traffic density and the penetration rate. The traffic density is defined as the number of vehicles per km of road [HBS01]. The penetration rate is the number of equipped vehicles divided by the total number of vehicles.

As a numerical example of equipment density, Lochert et al. give a value between 2.25 and 5 averaged over their metropolitan simulation area at a penetration rate of 100 %. This number is exceeded in particular at intersections. In a two-laned freeway scenario, [HBS01] gives a traffic density of 16–23 vehicles per km under normal conditions and a traffic density of 32–45 vehicles per km in case of heavy congestion. These numbers double when the two lanes in opposite direction are taken into account. With a radio range of about a quarter-kilometer and a penetration rate of 100 %, the equipment densities in such a scenario are about 8–11.5 (normal conditions) and 16–23 (heavy congestion) vehicles per radio range.

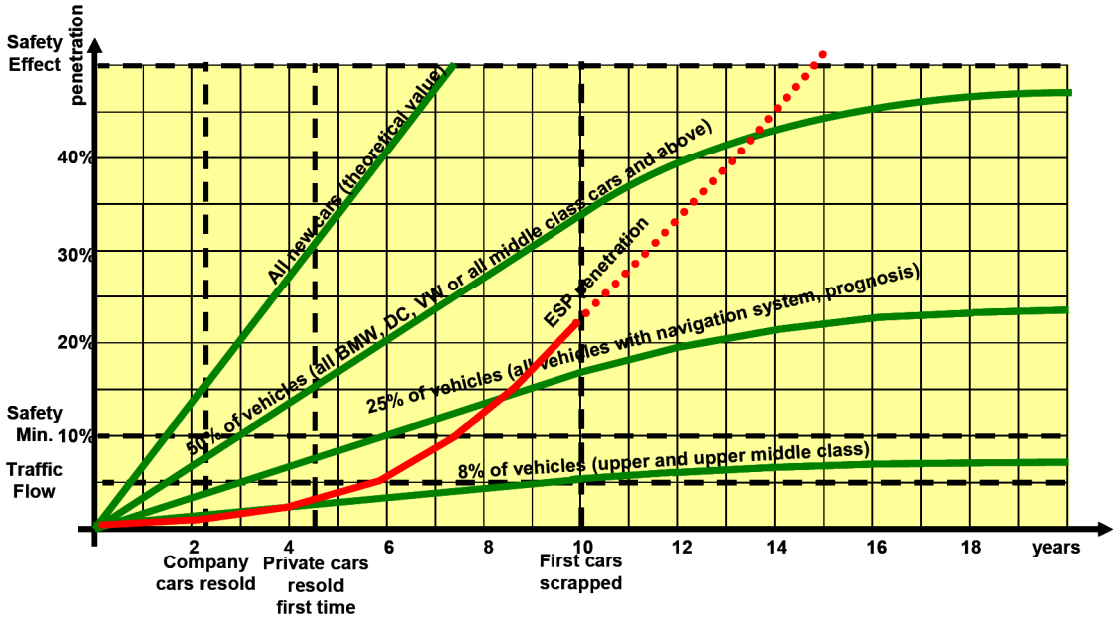


Figure 2.6: Expected penetration rates after market introduction of VANETs [MMP+05]

Figure 2.6 shows the growth of penetration rate for several market introduction strategies of VANETs. The strategies “all upper class vehicles” and “all vehicles with navigation systems” both lead to penetration rates of less than 20 % even after ten years. If the German car manufacturers equipped all their new vehicles with VANET technology, even after ten years the penetration rate would be only 35 %. This way, the most likely market introduction strategies all lead to a low penetration rate even after several years.

Even when the penetration rate is high, there will be scenarios where the VANET is sparse: in rural areas or during nighttime, a low equipment density can be expected due to low traffic density. Hence, sparse VANETs are not only a phenomenon occurring

within the first years after market introduction, but will also occur at higher penetration rates.

Unlike in dense VANETs with end-to-end connectivity, latency in communication is a problem that arises in sparse VANETs [WBM+07]: vehicles on the message path between sender and receiver have to cache and carry the message if no suitable next hop is available, and forward it if a new suitable neighbor has come into their radio range. The influence of latency has to be taken into account when developing services and applications for sparse VANETs, such as the search engine proposed in this thesis. Thus, expected latency is a core criteria of the feasibility study in Chapter 4 and strongly impacts our concept of index-based search in Chapter 6.

Chapter 3

Scenario and Simulation Environment

This thesis makes extensive use of simulation. The underlying scenario and simulation environment, which are unchanged throughout the thesis, are introduced here. The simulation environment is outlined in Section 3.1. This is followed by a characterization of the investigated scenario in Section 3.2. The key properties are summarized in Section 3.3.

3.1 Simulation Environment

Our simulation environment consists of three components: a vehicular mobility pattern, a model of the communication network, and a model of the topology of the simulation area (radio obstacles).

Mobility patterns may be obtained either from real movement traces [JHP+03, WBM+07] or self-generated using a mobility model [Fio06, CBD02, BHM07]. Artificial patterns may lead to fundamentally different results than realistic ones [MPGW06, BHM07]. For example, the frequently occurring chains of vehicles along roads do not occur in patterns based on the well-known random waypoint model [SJ04].

To obtain realistic vehicular mobility patterns, we employ the microscopic traffic simulator VISSIM. In VISSIM, the behavior of vehicles is implemented according to the model of Wiedemann [Wie74], which includes physical and psychological aspects of traffic. A VISSIM scenario contains sources where vehicles join the scenario, and drains where vehicles leave the scenario. The number of vehicles that join and leave the scenario within a certain period of time at a certain source or drain can be assigned in form of an origin-destination (OD) matrix. Each VISSIM simulation is initialized with a random seed; different random seeds produce different movement patterns, but all of them follow

the OD matrix. The vehicular traces obtained through VISSIM are used as movement patterns of mobile wireless nodes in a network simulator.

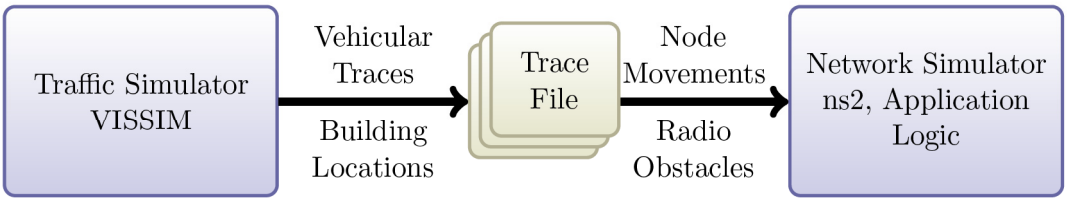


Figure 3.1: Simulation environment (offline coupling): vehicular traces are generated in VISSIM and, together with a list of buildings in the scenario, exported to files. This data is then used as movement pattern of mobile nodes and radio obstacles in ns2.

In our simulation environment, VISSIM is interlinked [LCS+05] with the network simulator ns-2[1]. Generally, three ways of interlinking a traffic simulator with a network simulator may be distinguished [WHWL08]: offline coupling, online one-way coupling, and online two-way coupling (feedback loop). In offline coupling, the traffic simulator is run and the resulting movement pattern is recorded into a trace file. This is preferable if complex scenarios are simulated and the resulting computation times of the traffic simulator are high. In online one-way coupling, updates of vehicle positions are sent to the network simulator after each simulation step. This way, the import of (occasionally very large) trace files can be avoided. In online two-way coupling, a VANET application may influence the behavior of vehicles in the traffic simulator. This allows studying the effects of VANET applications on traffic, e.g. the change of travel time [Mat05] or the reduction of fuel consumption [WHWL08] through a navigation system with real-time traffic information. As this thesis does not evaluate effects on traffic, and the simulated scenario is complex, offline coupling is used (Figure 3.1).

ns-2 comes with a model of the IEEE 802.11 physical and MAC layer, not including the IEEE 802.11p modifications. The simulator configuration is shown in Table 3.1. The channel model selected is the two-ray ground model: depending on the distance d from sender, the received power $p_r(d)$ calculates to

$$p_r(d) = \begin{cases} \frac{p_t g_t g_r \lambda^2}{(4\pi)^2 d^2 l} & \text{if } d \leq \frac{4\pi h_t h_r}{\lambda} \\ \frac{p_t g_t g_r h_t^2 h_r^2}{d^4 l} & \text{if } d > \frac{4\pi h_t h_r}{\lambda} \end{cases}, \tag{3.1}$$

where λ is the carrier wavelength, p_t is the power of the transmitted signal, g_t and g_r

[1] http://www.isi.edu/nsnam/ns

Parameter	Value
Channel	
Wireless network type	IEEE 802.11
Channel model	Two-Ray-Ground
Channel bitrate	11 MBit/s
Frequency	2.472 GHz
Radio	
Radio range	225 m
Transmit power	0.1 W
Receive threshold	$1.8423 \cdot 10^{-10}$ W
Carrier sense threshold	$8.1 \cdot 10^{-12}$ W
Capture threshold	10 dB
System loss l	1
Antenna	
Type	Antenna/OmniAntenna
Gain g	1
Height h	1.5 m

Table 3.1: ns-2 parameters

are the antenna gains of sender and receiver, l is the system loss, and h_t and h_r are the antenna heights of sender and receiver.

We set a transmit power of 0.1 W, which is the maximum allowed at 2.472 GHz in Germany. The selected receive threshold allows to receive packets from other nodes up to 225 m away (see Figure 3.2). The selected carrier sense threshold allows to sense other transmissions over distances up to 500 m, which is more than twice the receive threshold. This avoids hidden node situations in topologies like the one shown in Figure 2.5(a), because the transmission of the distant node can then be sensed.

There is no model of channel bit error rate in ns-2. Instead, ns-2 regards a packet as correctly received if the received signal power is greater than the receive threshold. If multiple signals arrive simultaneously, only the packet with the highest power is considered; this packet is received correctly ("captured") if the ratio between its received power and the signal strength sum over all other concurrently received transmissions is greater than the capture threshold (10 dB in our case).

We enable the ns-2 option of acknowledgments (ACKs) for unicast messages. ns-2 furthermore has the option of RTS/CTS for unicast messages. For three reasons, we disable RTS/CTS. First, the main benefit of RTS/CTS is that it avoids hidden node situations as that in Figure 2.5(a), but these situations do not occur in our setup because

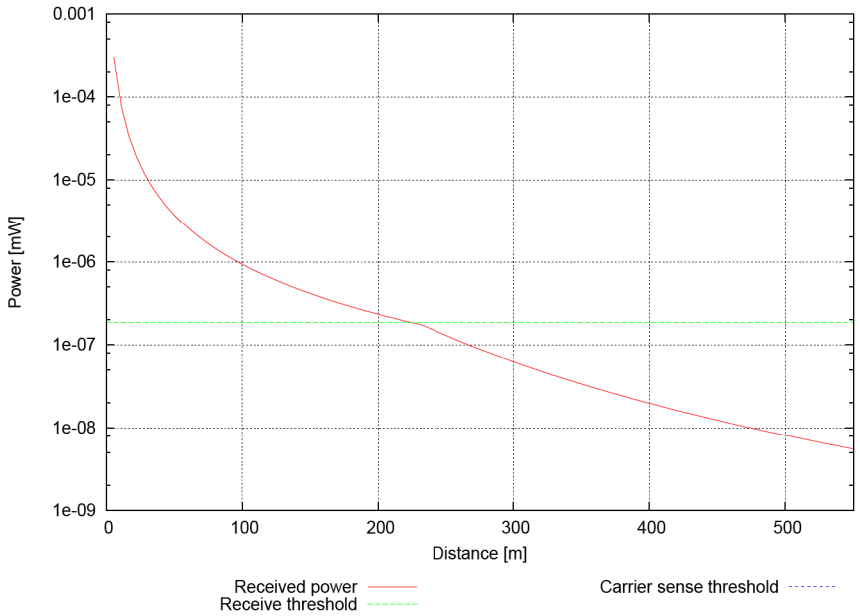


Figure 3.2: Received signal power over distance. Packets may be received up to a distance of 225 m away from the sender. Transmissions of senders up to 500 m away may be sensed.

of the large carrier sense range. Second, RTS/CTS messages themselves require channel capacity and are typically only used for large packets, but packets in our case are small. Third, measurements by Bicket et al. [BABM05] as well as an analyses by Xu et al. [XGB03, XS01] raise doubts about the effectiveness of the RTS/CTS mechanism as a whole.

The simulation environment also includes a model of radio obstacles in the simulation area: radio transmission between two nodes is impossible if their line of sight is obstructed by a radio obstacle. Radio obstacles are modeled as polygons in the (x,y)-plane and are assumed to be of infinite height. In the simulator interlinking environment, the list of polygons is passed to ns-2 together with the movement patterns.

3.2 Scenario

The investigated scenario is a 4 km · 4 km square around the city center of Braunschweig, Germany. The road network as well as the radio obstacles are depicted in Figure 3.3. The obstacles prohibit short-cuts of packets, i.e. the forwarding of a packet to a node at an adjacent road without reaching the corresponding intersection.

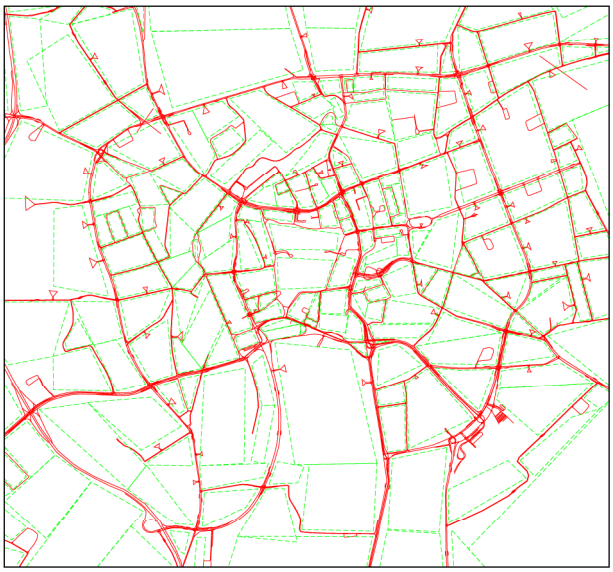


Figure 3.3: Simulation scenario: city center of Braunschweig; polygons indicate radio obstacles

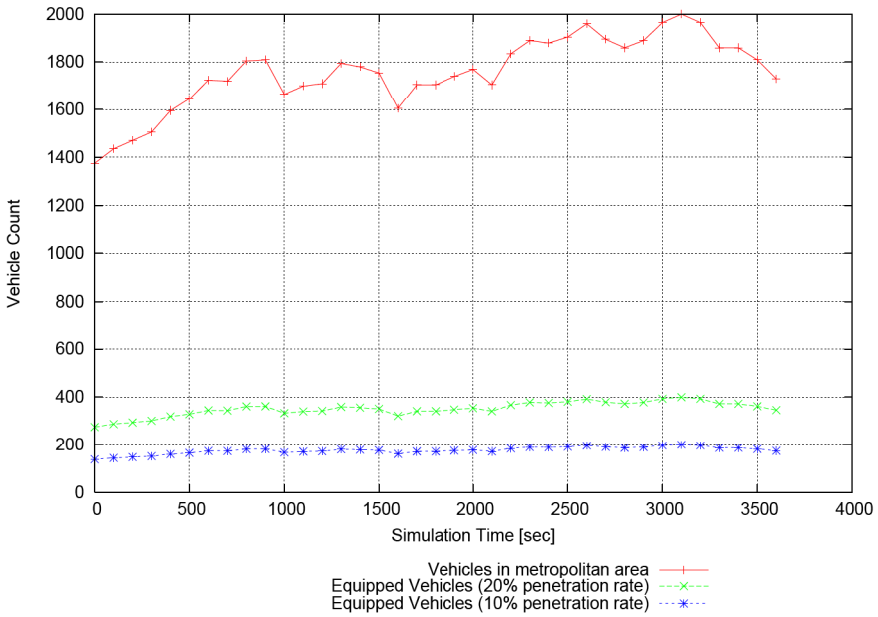


Figure 3.4: Count of vehicles in scenario over simulation time

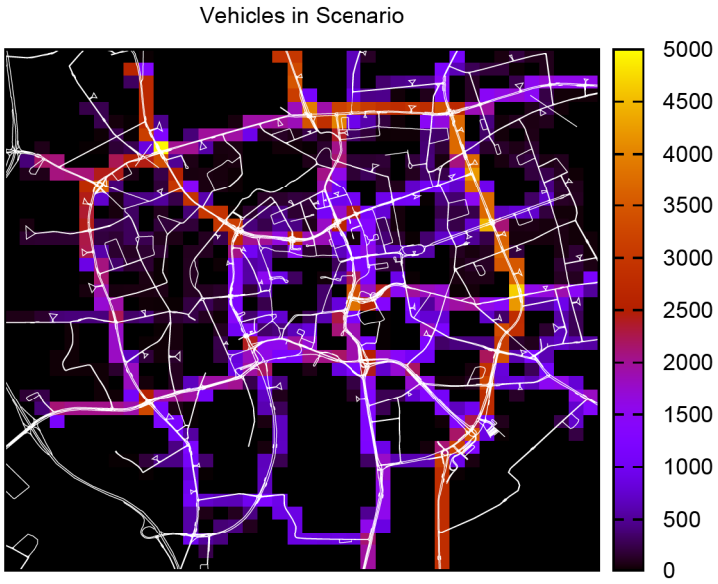


Figure 3.5: Traffic density in simulation area. Colors indicate total number of vehicles that crossed a square during simulation time. The traffic situation is highly imbalanced.

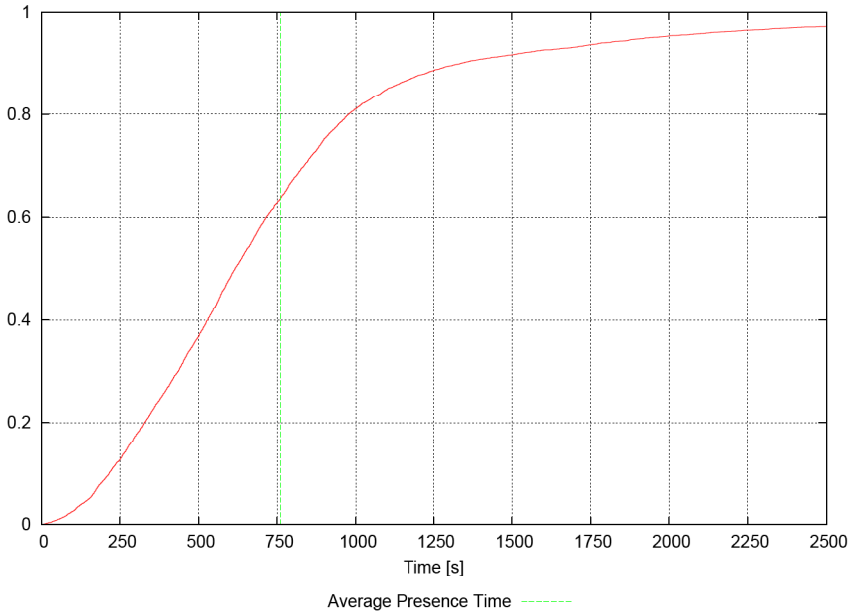


Figure 3.6: EDF: vehicle presence time. The average presence time is only about 750 s.

Figures 3.4, 3.5, and 3.6 have been included to give the reader a better understanding of the traffic conditions in the scenario. Figure 3.4 shows that there are between 1400 and 2000 vehicles in the scenario at the same time. Figure 3.5 shows that traffic in the metropolitan area is highly imbalanced: while there are areas that are crossed by many vehicles (especially intersections), others are rarely visited. Figure 3.6 depicts the empirical distribution function (EDF) of vehicle presence time. The curve indicates that there is high fluctuation in traffic: vehicles are only present in the scenario for about 750 s on average. The average vehicle speed in the scenario is about 7.5 m/s.

3.3 Chapter Summary

In this chapter we have presented the simulation environment and sample scenario that are used in evaluations throughout this thesis. The simulation environment is an interlinking between the traffic simulator VISSIM and the network simulator ns-2. The traffic simulator creates realistic vehicular mobility patters that we use in simulation of the wireless vehicular network. The simulated network is based on CSMA for channel access and has a maximum transmission range of 225 m at a data rate of 11 MBit/s. The scenario is a city scenario of 4 km · 4 km size. Up to 2000 vehicles are present in the metropolitan area at the same time, for about 750 s on average. The traffic situation in the metropolitan area is highly imbalanced.

Chapter 4

Feasibility of Search in Sparse VANETs

In this chapter we assess existing search schemes regarding their success rate and latency in a VANET scenario. We define the terms *success rate* and *search latency* from the originator point of view. In the Internet domain, the originator regards a search request as successful if it receives lookup results containing the addresses of one or more matching information sources. In contrast, our approach in the VANET domain is that the search engine does not return lookup results to the originator, but instead directly issues content requests to registered information sources (see Section 1.2); here, the originator regards a search as successful if it receives the requested content. We call this probability the *success rate*. Correspondingly, we define *search latency* as the time period between the originator creating the search request and receiving the requested content.

We study three different search schemes: flooding-based search, search based on a geographic hash table, and search based on a distributed hash table. One way of comparing these schemes would be to implement and evaluate all of them by means of simulation. We refrain from this as our first goal is to determine theoretical bounds of these schemes; instead we mathematically analyze abstractions of the schemes in combination with a simulation study.

After this analysis, we comment on the expected *network load* caused by these schemes. We define this *network load* as the amount of data transmitted over the VANET to forward search requests and content requests, plus the amount of data transmitted for maintaining the search index. The network load for transmitting the reply is not further considered here as we assume the reply is always delivered in the same way.

In the next section (Section 4.1), we survey related work, elaborate the three search schemes, and apply these concepts to the VANET domain. Afterwards, we determine

their success rates and latencies (Section 4.2). We discuss search network load in Section 4.3. We draw conclusions from our findings in Section 4.4 and summarize the chapter in Section 4.5.

4.1 Related Work

We now review research results on content location schemes in the context of the Internet (Section 4.1.1). Researchers have adapted some of these schemes for the domain of mobile wireless networks. We review these adaptions along with other applicable schemes in such networks (Section 4.1.2) and illustrate how the index-based concepts could be applied in VANETs (Section 4.1.3). Thereon, we examine work on determining information propagation speed in VANETs (Section 4.1.4). Lastly, we summarize our key findings (Section 4.1.5).

4.1.1 Content Location in the Internet

A service that provides search functionality in the Internet essentially has to be distributed: the enormous amount of data as well as the large number of concurrent search requests make it impossible to provide search functionality using only a single machine. It is common to classify distributed system architectures into client-server and peer-to-peer architectures [CDK05]. In client-server architectures, one or more nodes (servers) provide services to other nodes (clients) in the network. In peer-to-peer architectures, there is no such hierarchy—all nodes equally provide service functionality. This section continues with an overview of content location schemes in each architectures type.

Thinking of client-server search architectures, Internet search engines like Google [PBMW99] come to mind. These search engines rely on a search index that is distributed and replicated within a large cluster of servers, allowing a high number of requests to be processed in parallel. To register the content of a web server, its address and keywords describing the hosted content have to be stored in the search index. If content and address of the web server are fixed, there is no need for repetitive registration in the search index to locate an information source. This is different from a search index of content in a VANET: here, the index contains locations of vehicular information sources. As information about these locations ages quickly and vehicles may completely disappear, vehicles are required to periodically re-register in the search index.

What methods of search are applicable in peer-to-peer architectures depends on the degree of organization in the underlying network. Such networks may be divided into two organizational classes: unstructured networks and networks structured for search. There are surveys of content location techniques in both network classes: Li and Wu give such an overview in [LW04], Tsoumakos and Roussopoulos review searching techniques in unstructured networks [TR03]. In the following, we take a closer look at possible search approaches in each network class.

In unstructured networks, network nodes have no knowledge about information stored in other nodes. In this case, commonly used search methods are flooding, e.g. by breadth-first-search as applied in Gnutella [Rit01], heuristically directed search [YGM02], and random walks [LCC+02, BA05]. Heuristics based on historical information may not improve search performance if the network is highly dynamic. Random walks ignore historical information and thus have disadvantages in case this historic information may give an indication about possible paths to information sources. Flooding has the disadvantage of causing high network load. Regardless of this downside, flooding has an interesting property relevant for our work: all nodes are contacted as fast as possible. As long as flooding-based search does not congest the network, it leads to the lowest possible search latency (lower bound) and to the highest possible search success rate (upper bound).

Structured networks are organized with help of meta information that is distributed among nodes. This way, an overlay network is constructed on top of the physical network, enabling more intelligent techniques of content location. One well-known type of overlay is a distributed hash table (DHT). DHTs establish a virtual ID space: each node joining a DHT acquires its virtual ID (bootstrapping) and is then responsible for a range of hash values (keys) around its own ID. DHTs support two basic operations, namely store and query. Nodes use the store operation to register their keys in the DHT. They use the query operation to look for keys they are interested in. To perform these operations, messages have to be routed to the responsible nodes. This is done with help of the DHT (overlay-routing) and requires each node to maintain a table of virtual neighbors (overlay neighbors). The neighbor table is designed in a way that allows each node to forward a message associated with a certain hash key to a node closer to the key in the key space than itself, so it is guaranteed that the message reaches the responsible node.

A key advantage of DHTs over unstructured networks is that neighbor tables can be constructed in a way that allows to reach *any* node in the network with only very few

overlay-routing hops. For example, the Chord DHT [MKKB01] is set up so that any node can be reached in at most $O(log\, n)$ hops, where n is the total number of nodes in the network. Other DHTs with similar routing properties have been published, among them Pastry [RD01], and CAN [RFH+01]. A big drawback of DHTs is that virtual neighbors and physical neighbors are uncorrelated. This leads to messages zigzagging the physical network, causing a much longer routing path to the responsible node than the shortest possible path. The relation between the physical path taken by a message and its shortest possible physical path is called *stretch*. A high stretch may be tolerable in networks with reliable communication and sufficient bandwidth capacity, but is inacceptable in MANETs.

4.1.2 Content Location in MANETs

Services in MANETs are typically designed following the peer-to-peer paradigm, thereby achieving high robustness and fault tolerance. Like in the previous section, we overview applicable search methods in unstructured MANETs and in structured MANETs. Lastly, we discuss the relation between a MANET location service and a VANET search service.

Message flooding schemes can be used to distribute content requests in unstructured MANETs. Several researchers have developed controlled flooding schemes for such networks. Vahdat and Becker have introduced a three-step message transmission scheme [VB00]. In the first step, upon noticing a new contact in its radio range, a node advertises its locally stored messages with help of a summary vector. In the second step, the new contact returns a similar vector indicating which messages it is interested in. In the third step, the node transmits the requested messages. Harras et al. present a similar scheme, in which nodes periodically send beacons to advertise messages [HABR05]. Beacon receivers return an acknowledgment if they are interested in the advertised messages, that are then sent by the advertiser. The message propagation scheme we introduce and apply in the simulation study in Section 4.2.2.1 employs this beacon-based advertisement.

Moving to structured MANETs, efforts have been made to reduce the routing stretch of DHTs, allowing them to work efficiently in these networks [PDH04, ZS06, CF05, ZS05]. Alternatively, instead of distributing a hash table dynamically among nodes in the network—causing a routing stretch—the hash table may be distributed over geographic regions (geographic hash table, GHT [RKY+02, GZ06]). In this case, the

nodes in each region are responsible for a fixed part of the key space. The geographic distribution scheme is assumed to be known by the nodes. Thus, whenever a node wants to store or query a key, it can compute the responsible region and send a message to that region. For this purpose, it uses position-based routing, which is more efficient than the overlay routing of DHTs because it avoids the routing stretch. Nevertheless, GHTs have another drawback: when a node moves from one region to another, it has to maintain the part of the hash table associated with the new region, which it needs to receive from a node inside that area. This causes high network load for maintenance in networks with rapid node movement, such as VANETs.

As mentioned in Section 1.2, a VANET search engine may also be regarded as an extended location service. Location services are a well-studied field of MANET research [FK06]. With help of a location service, a node may find the location of another node, given that it knows the other node's ID. Such location awareness is essential for MANET communication based on position-based routing. The assumption underlying MANET location services is that a node already knows which node it wants to communicate with. In contrast, the VANET search schemes investigated in this chapter help nodes to locate specific content in the network, and thus may be used when a node has no indication if and where certain content is available. In short, while a location service in the MANET context is a *node location service*, the VANET search engine—as we define it (see Section 1.2)—is a *content location service*. Another difference to location services is that the VANET search schemes studied in this chapter do not return the location information to the originator but instead directly issue a content request to the information source.

MANET location services rely on periodic registrations of the network nodes. This is the same for the index-based content location schemes investigated in this chapter, which implies an important challenge: the number of information items available in the network may by far exceed the number of nodes. Network load of such registration messages has to be taken into account, as high bandwidth usage due to an inefficient registration process may degrade performance of other applications on the wireless channel. Bandwidth-efficient registration is the topic of Chapter 7.

4.1.3 Index-based Content Location in VANETs

Let us sketch a DHT and a GHT for content location in VANETs. Each vehicle contributing to the search index makes its own information items $(I_1, I_2, \ldots, I_n)$ available for

search. Commonly used search engines are based on inverted indices mapping keywords to documents [BL85]. This way, vehicles would register (document identifier, keyword set)-tuples in the search index. For better bandwidth-efficiency, we here assume that vehicles instead apply a well-defined description scheme for their information items, allowing vehicles interested in a particular item I to compute the respective description D_I. Thus, to register their information items, we assume that vehicles store their node ID, position, and hash keys of all information descriptions $h(D_I)$, $I \in I_1, I_2, \ldots, I_n$ at the respectively responsible nodes in the hash table. This has to be performed repeatedly since their positions vary over time and they leave the VANET after some time.

Based on the hash table, we assume the following search technique (Figure 4.1): the originator sends a search request to the responsible vehicle/region. A responsible vehicle receiving the request checks for registered information sources. If no information source is found, a negative reply is returned to the originator. If a suitable information source is found, the responsible vehicle issues a content request to the registered position. Alternatively, the position information of the information source could be returned to the originator of the search request, which then in turn contacts the information source. This has the major drawback of increased search latency, thus we do not further consider this alternative.

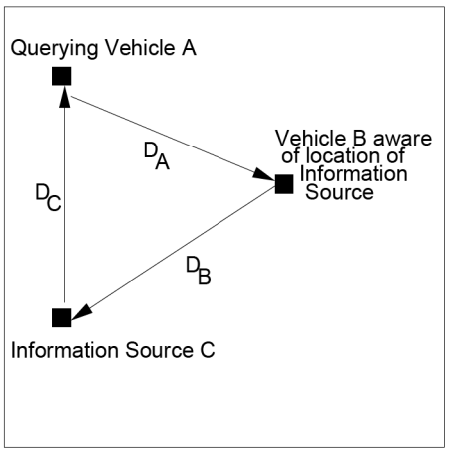


Figure 4.1: Three vehicles involved in searching information in a structured VANET

Comparing GHT- and DHT- to flooding-based search shows an important difference in the number of subsequently involved vehicles. In GHT and DHT, this number is three because the originator first issues a search request that has to reach a vehicle maintaining the corresponding part of the search index. This vehicle then sends a content request to an information source which replies to the originator. In flooding, this number is

only two because the originator directly issues a content request that is sent to the information source (and—in parallel—to all other vehicles). This has a big impact on the expected search latency and therefore on success rate, as our following study will show.

4.1.4 Measuring Information Propagation Speed in VANETs

As part of our feasibility study, we determine the distribution of information propagation speed in a VANET scenario. Lochert et al. have conducted a similar simulation study on information propagation speed [LSCM07]. Like us, the authors also investigate the effect of base stations and find that, while the improvement through stand-alone base stations is marginal, interconnected base stations have a very positive effect on information propagation speed. Our work extends their study by assuming an underlying distribution of information propagation speed that we approximate through a log-normal distribution function. Furthermore, while Lochert et al. focus on delivery of a single message, we calculate the distribution of the total delay after multiple sequential message deliveries. Finally, vehicles in the simulation study of Lochert et al. propagate information via periodic single-hop broadcast, using a broadcast interval of 1 s. This causes additional latency in intra-partition forwarding, and thus their results underestimate the achievable information propagation speed. In contrast, vehicles in our simulation employ more aggressive intra-partition forwarding, thereby achieving a latency closer to the minimum.

4.1.5 Summary

Subsuming the surveyed results, there exist several decentralized content location schemes designed for the Internet or MANETs with low mobility, classifiable into flooding-based and hash-based (GHT, DHT). Nevertheless, the network characteristics of sparse VANETs, especially the high latency in communication, leave it unclear whether any of these schemes achieves an adequate success rate.

To answer this question, we model and analyze the different content location schemes with help of a simulation study of information propagation speed in a VANET scenario. Currently published studies underestimate the achievable information propagation speeds, and thus need to be refined to give more realistic results.

4.2 Quantitative Analysis of Search Latency

In this section we compare latencies of GHT-, DHT-, and flooding-based search. The success rates of the search schemes follow directly from the results: we compute the fraction of searches having infinite latency—these are the searches that fail. We study the search schemes under the assumption that exactly one information source is available at a random location inside a metropolitan area; we do not investigate the search for unavailable content. Next, we outline the methodology behind our study (Section 4.2.1) before presenting the results (Section 4.2.2).

4.2.1 Methodology

We are interested in the latencies of GHT-, DHT-, and flooding-based search. To calculate these, we apply the formula

$$T = \frac{D}{V}, \tag{4.1}$$

where D, V, T is the distance, speed, and time, respectively. In our context, D is the geographical distance that needs to be covered by messages to complete a search, and V is the information propagation speed. Naturally, both D and V vary from one search to another; therefore, we model them as continuous random variables; then, T of course is also a random variable. Our goal is to calculate the cumulative distribution function (CDF) of T, i.e. $P(T \leq t)$.

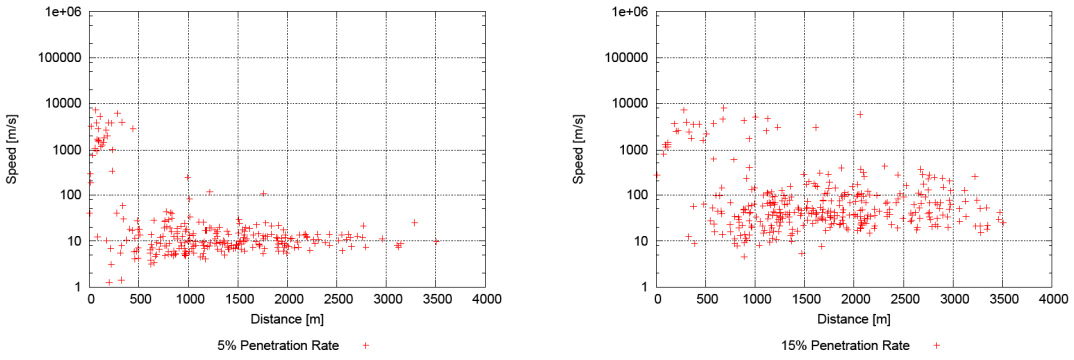


Figure 4.2: Simulation results: information propagation speed over communication distance for penetration rates 5 % and 15 % (logarithmic scale on y-axis). High speed when the receiver is in radio range of the sender. High speed is more probable over short distances than over long distances. Higher penetration rate leads to higher propagation speed on average.

For our calculation, we treat D and V as independent. This is a simplification of reality, where these variables are not totally uncorrelated: high information propagation speed is more probable over short distances than over long distances, because a shorter communication chain of vehicles suffices (see Figure 4.2). In the extreme case that the target of a message is in radio range of the originator (D is small), information propagation speed is almost infinite since the message can be delivered immediately. For our calculation, we regard this case as a negligible exception because the examined area is large in relation to the radio range: in our case study, the area size is $4\,\mathrm{km} \cdot 4\,\mathrm{km} = 16\,\mathrm{km}^2$; the radio range is about 225 m, thus the radio of a vehicle covers an area of $\pi(0.225\,\mathrm{km})^2 \approx 0.159\,\mathrm{km}^2$, leading to a relation of about 101 to 1.

Our assumed distribution of V is a result of samples measured over all communication distances within the simulation scenario; consequently, on average information propagation speed over short distances is slightly underestimated while that over long distances is slightly overestimated.

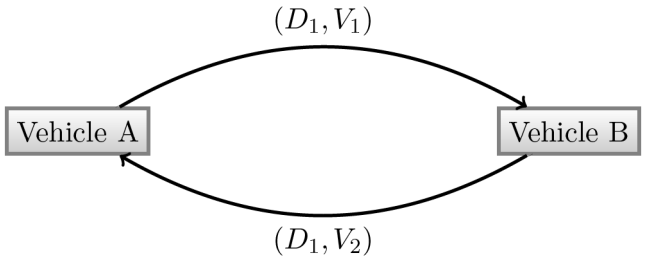


Figure 4.3: Model of flooding-based search. Vehicle A floods a content request that reaches vehicle B covering distance D_1 with speed V_1. Vehicle B returns the reply covering about the same distance D_1 with speed V_2.

In flooding, there are two deliveries over approximately the same distance, as illustrated in Figure 4.3. Therefore we model the search latency of flooding T_{fl} by

$$T_{fl} = \frac{D_1}{V_1} + \frac{D_1}{V_2}. \tag{4.2}$$

GHT- and DHT-based search require three subsequent deliveries (Figure 4.4). The search latency of a GHT is thus modeled as

$$T_{GHT} = \frac{D_1}{V_1} + \frac{D_2}{V_2} + \frac{D_3}{V_3}. \tag{4.3}$$

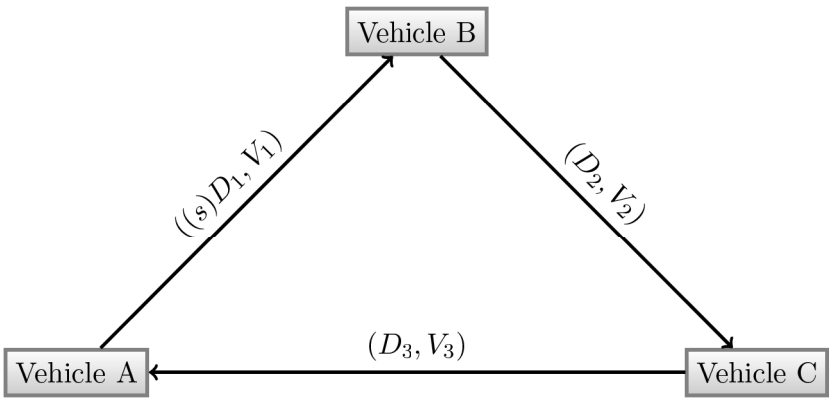


Figure 4.4: Model of hash-based search, assuming independent delivery processes. Vehicle A directs a search request to vehicle B. With speed V_1, the search request covers a distance D_1 in case of a GHT or $s \cdot D_1$ in case of a DHT with stretch s. Vehicle B issues a content request towards vehicle C, covering distance D_2 with speed V_2. Vehicle C issues a search reply towards vehicle A, covering distance D_3 with speed V_3.

DHT-based search is a special case of the above: the overlay stretch s increases the communication distance on the delivery AB (the DHT is only used for the first delivery because afterwards there is enough information to perform position-based routing on the remaining two deliveries). Thus,

$$T_{DHT} = \frac{sD_1}{V_1} + \frac{D_2}{V_2} + \frac{D_3}{V_3}. \tag{4.4}$$

Our goal is to determine the CDFs of T_{fl}, T_{GHT}, and T_{DHT}. The CDFs can be calculated if the probability density functions (PDFs) of all D_i and V_i are known. These PDFs are determined in the next section. This is followed by the calculation of the desired CDFs.

4.2.1.1 Calculating the PDFs of distance and speed

To determine f_{V_i}, we perform a simulation study of the scenario introduced in Chapter 3. Repeatedly during simulation, we choose a random vehicle to send a message to another random vehicle and measure the geographical distance D between the vehicles and the delay T until the message arrives. This way, we obtain a set of n samples of information propagation speeds $\overline{V}_k = D_k/T_k$, $k = 1 \ldots n$. These samples follow a certain

probability distribution, resulting from a certain probability density. We approximate this probability density through a PDF.

We classify each sample of information propagation speed into one of three categories: almost infinite ($T \approx 0$), failed ($T = \infty$), and delayed. The results of our experiments, presented in Section 4.2.2, will show that the PDF of the delayed deliveries can reasonably be approximated through the PDF of a log-normal distribution:

$$\overline{f}_V(v, \mu, \sigma) = \frac{1}{v\sigma\sqrt{2\pi}} e^{-\frac{(ln(v)-\mu)^2}{2\sigma^2}}. \tag{4.5}$$

We then set $f_{V_i} = \overline{f}_V(v, \mu, \sigma)$. It is important to note that this approximation only holds for the delayed deliveries. Nevertheless, the complete CDF of search latency has to contain failed and almost infinitely fast searches as well. Their respective probabilities have to be calculated in a different way (Section 4.2.1.2), as the PDF given by Equation 4.5 does not cover failed and infinitely fast message deliveries.

Now we turn to the calculation of the PDF of expected geographical distance between sender and receiver (f_{D_i}). To enable an analytical calculation of this function, we assume that the metropolitan area is square and vehicles are uniformly distributed. Consequently, the PDF of the distance between two random vehicles is the PDF of the distance between two random points in a square, $\overline{f}_D(d)$. This function in the unit square is well-known [Wei] to be

$$\overline{f}_D(d) = \begin{cases} 2d(d^2 - 4d + \pi) & \text{for } 0 \leq d \leq 1 \\ 2d(4\sqrt{d^2-1} - (d^2 + 2 - \pi) - 4\arctan(\sqrt{d^2-1})) & \text{for } 1 \leq d \leq \sqrt{2} \end{cases} \tag{4.6}$$

The PDF is depicted in Figure 4.5. It can be adapted to square areas of any size through a transformation of random variables: for an $l \cdot l$ square, we set $D_l = g(D) = l \cdot D$, knowing the PDF $\overline{f}_D(d)$ of D. Then, it follows that [SSS00],

$$\overline{f}_{D_l} = \frac{\overline{f}_D(g^{-1}(D_l))}{|g'(g^{-1}(D_l))|}, \tag{4.7}$$

where we have

$$g^{-1}(D_l) = \frac{D_L}{l}; g'(D) = l, \tag{4.8}$$

therefore,

$$\overline{f}_{D_l}(d) = \frac{\overline{f}_D(\frac{d}{l})}{l}. \tag{4.9}$$

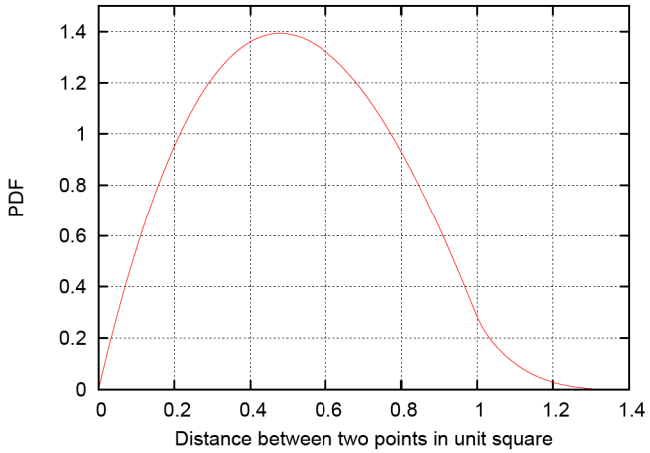


Figure 4.5: PDF: distance of two points in unit square

We set $f_{D_i}(d_i) = \overline{f}_{D_l}(d)$.

4.2.1.2 Calculating the CDF of Search Latency

We now calculate the CDF of search latency. For this we have to take into account that through simulation we only have a CDF of information propagation speed for delayed deliveries. Thus, we will not be able to calculate the probability of infinitely fast or failed searches, cases that essentially are part of a complete CDF; we have to find another way to determine these. We do so by conducting a case differentiation.

Recall the three distinct outcomes of a message delivery. First, the message may be delivered almost *infinitely fast* if the target is in radio range of the originator or if there is a multi-hop link between originator and target. We call the probability of this case P_∞. Second, the message may *never* be delivered because the VANET is too sparse to allow transport of the message into the target area. We call the probability of this case P_0. Third, the message may be delivered with a certain *delay* if there is store-and-forwarding. We call the probability of this case P_d. We determine P_∞, P_0, and P_d in our simulation study. From the first two probabilities, we can calculate probabilities of search requests to fail ($P(flooding_0)$, $P(hash_0)$) and probabilities of search requests to be completed immediately ($P(flooding_\infty)$, $P(hash_\infty)$). The rationale of this case differentiation is to construct the final CDF $F_T(T \leq t)$ of the search latency T so that it evaluates to $P(flooding_\infty)$ ($P(hash_\infty)$, respectively) at $t = 0$ and to never exceed $1 - P(flooding_0)$ ($1 - P(hash_0)$, respectively) (cf. Figure 4.6).

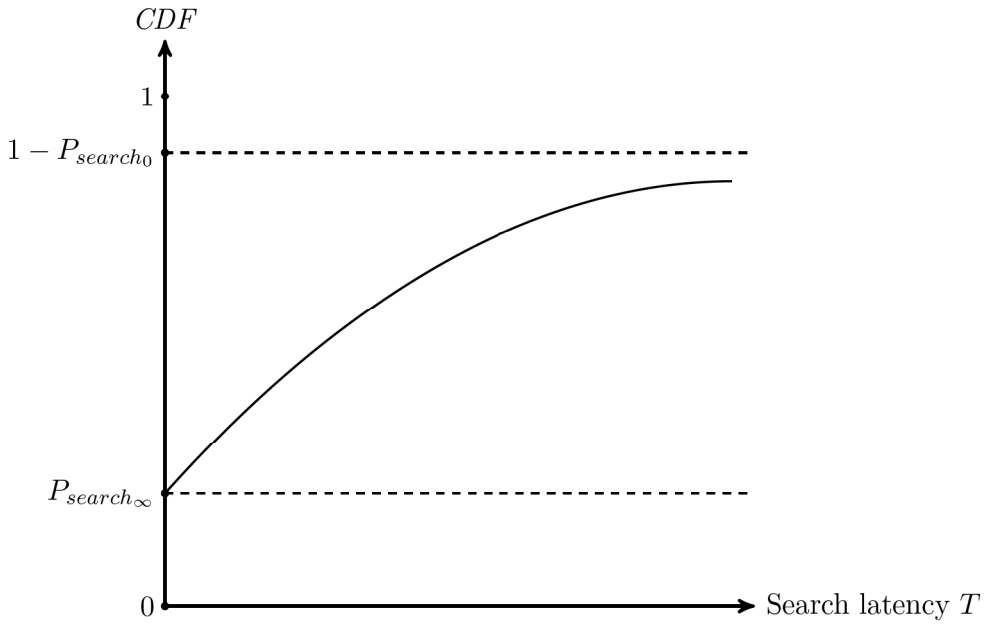


Figure 4.6: Idea of case differentiation

AB	BA	Outcome
0	X	0
X	0	0
d	d	d
d	∞	d
∞	d	impossible
∞	∞	∞

Table 4.1: Outcomes of flooding. 0: failed delivery; d: delayed delivery; ∞: immediate delivery; X: "don't care" = any of $\{0,d,\infty\}$

We start with the calculation of these probabilities for flooding. Table 4.1 shows the possible outcomes of flooding-based search. A 0 indicates a failed delivery, ∞ indicates an infinitely fast delivery and d indicates a delivery with latency $0 < l < \infty$. The X symbolizes a "don't care". Flooding fails if delivery of either the request or the reply fails.

To determine the probabilities $P(flooding_0)$, $P(flooding_\infty)$, and $P(flooding_d)$, we model the two deliveries as stochastic processes. We assume independent processes because of the high dynamics of vehicular traffic, with one obvious exception: if the request is delivered immediately, we assume that the response can be delivered immediately because there is no node movement in between. In all other cases we define the outcomes of the two delivery processes as independent. Next, we calculate the probabilities of each outcome with help of P_0 and P_∞.

Search is completed immediately if and only if the request is delivered immediately, therefore

$$P(flooding_\infty) = P_\infty. \tag{4.10}$$

Search fails if either the first delivery fails or if the first delivery succeeds with delay and the second delivery fails:

$$P(flooding_0) = P_0 + (1 - P_0 - P_\infty)P_0 \tag{4.11}$$

Here, it is important to note that we assume equal failure probabilities P_0 for the two deliveries. Since we determine P_0 from samples of one-way deliveries, the likeliness of a failed search is slightly underestimated: in reality, higher failure probability of the second delivery can be expected because the target has to be present in the VANET for a longer period of time than that of the first delivery.

If search does not fail and is not completed immediately, it succeeds with delay:

$$P(flooding_d) = 1 - P(flooding_\infty) - P(flooding_0) \tag{4.12}$$

Having calculated $P(flooding_0)$ and $P(flooding_\infty)$ that define the bounds of the CDF, what remains is the calculation of the CDF itself in between, i.e. in the case of a delayed search. Looking at Table 4.1 again, there is a probability that—although search is delayed—the reply may be delivered infinitely fast ($P(req_d \cap rep_\infty | flooding_d)$). This has to be taken into account when calculating the CDF, therefore we again conduct a case differentiation distinguishing the cases that this is true and false. The idea of

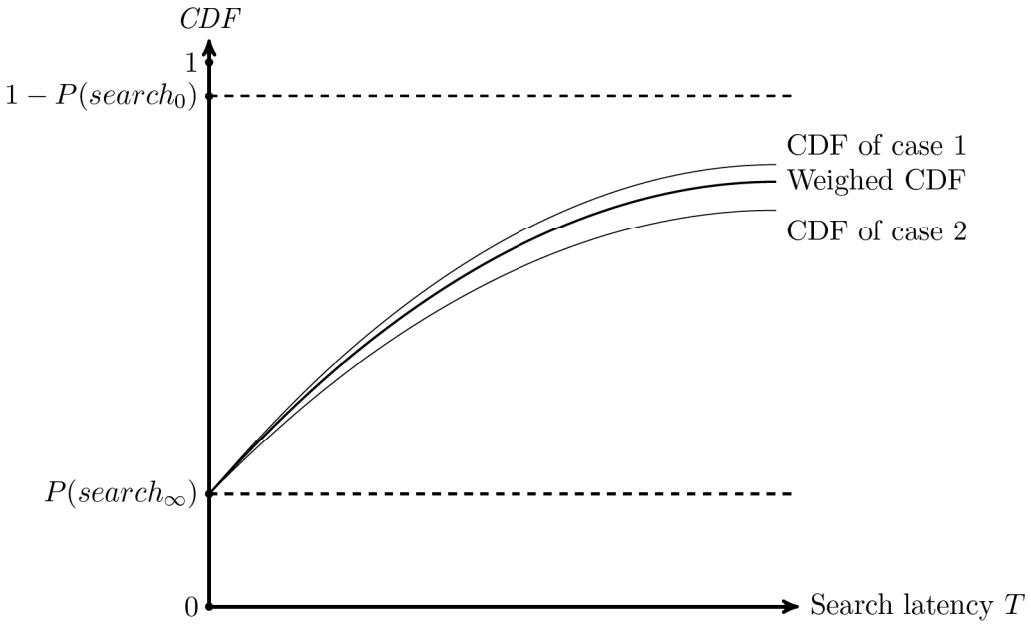


Figure 4.7: Final CDF as weighed CDF of two possible cases

how we determine the final CDF is shown in Figure 4.7. We calculate the CDF of each case and combine them using the weights $P(req_d \cap rep_\infty | flooding_d)$ and $1 - P(req_d \cap rep_\infty | flooding_d)$:

$$\begin{aligned} P(T_{fl_d} \leq t) = {} & P(req_d \cap rep_\infty | flooding_d) P(T_{fl_{d1}} \leq t) \\ & + (1 - P(req_d \cap rep_\infty | flooding_d)) P(T_{fl_{d2}} \leq t), \end{aligned} \tag{4.13}$$

where $P(T_{fl_{d1}} \leq t)$ is the CDF of delayed flooding-based search latency with an immediately delivered reply and $P(T_{fl_{d2}} \leq t)$ is the CDF of flooding-based search latency with delayed request and delayed reply.

To obtain the complete CDF, this combined CDF is then shifted and scaled according to the values of $P(flooding_0)$ and $P(flooding_\infty)$:

$$P(T_{fl} \leq t) = P(flooding_\infty) + \frac{P(T_{fl_d} \leq t)}{(1 - P(flooding_0)) - P(flooding_\infty)} \tag{4.14}$$

We first determine $P(req_d \cap rep_\infty | flooding_d)$, then calculate $P(T_{fl_{d1}} \leq t)$ and

$P(T_{fl_{d2}} \leq t)$.

$$\begin{aligned}
P(req_d \cap rep_\infty | flooding_d) &= \frac{P(req_d \cap rep_\infty \cap flooding_d)}{P(flooding_d)} \\
&= \frac{P(flooding_d \cap req_d \cap rep_\infty)}{P(flooding_d)} \\
&= \frac{P(flooding_d | req_d \cap rep_\infty) P(req_d \cap rep_\infty)}{P(flooding_d)} \\
&= \frac{1 P(req_d \cap rep_\infty)}{P(flooding_d)} \\
&= \frac{P(req_d) P(rep_\infty)}{P(flooding_d)} \text{ (independent deliveries)} \\
&= \frac{(1 - P_0 - P_\infty) P_\infty}{P(flooding_d)} \qquad (4.15)
\end{aligned}$$

If this holds (immediately delivered reply), the total delay is simply

$$T_{fl_{d1}} = \frac{D_1}{V_1}. \qquad (4.16)$$

Otherwise, the delay is

$$T_{fl_{d2}} = \frac{D_1}{V_1} + \frac{D_1}{V_2}. \qquad (4.17)$$

Since the PDFs of D_i and V_i are known (Section 4.2.1.1), we can calculate [SSS00] $P(T_{fl_{d1}} \leq t)$ (CDF of case 1 in Figure 4.7) and $P(T_{fl_{d2}} \leq t)$ (CDF of case 2 in Figure 4.7) through the CDFs of $T_{fl_{d1}}$ and $T_{fl_{d2}}$:

$$P(T_{fl_{d1}} \leq t) = F_{T_{fl_{d1}}}(t) = \iint_R f_{D_1,V_1}(d_1, v_1) dd_1 dv_1,$$

$$\text{with } R = \{(d_1, v_1) | \frac{d_1}{v_1} \leq t\} \qquad (4.18)$$

$$P(T_{fl_{d2}} \leq t) = F_{T_{fl_{d2}}}(t) = \iiint_R f_{D_1,V_1,V_2}(d_1, v_1, v_2) dd_1 dv_1 dv_2,$$

$$\text{with } R = \{(d_1, v_1, v_2) | \frac{d_1}{v_1} + \frac{d_1}{v_2} \leq t\} \qquad (4.19)$$

According to our assumption of independence, the joint PDFs in these equations are simply given by the products of the PDFs of the single random variables.

In case of hash-based search, we proceed accordingly. Because of the three message deliveries, the case differentiation becomes slightly more complex. Table 4.2 shows the possible outcomes. Note that in case of exactly two infinitely fast deliveries AB and

AB	BC	CA	Outcome
0	X	X	0
X	0	X	0
X	X	0	0
d	d	d	d
∞	d	d	d
d	∞	d	d
d	d	∞	d
∞	∞	d	d
d	∞	∞	d
∞	d	∞	d
∞	∞	∞	∞

Table 4.2: Outcomes of hash-based search. 0: failed delivery; d: delayed delivery; ∞: immediate delivery; X: "don't care" = any of {0,d,∞}

BC, it is also conceivable to assume that the total outcome is an infinitely fast delivery: a routing algorithm that tracks the latency of previous deliveries could propagate the message backwards from C to B and then to A. We here do not assume such an algorithm and suppose that the message is forwarded from C to A.

We again start with the calculation of $P(hash_0)$, $P(hash_\infty)$, and $P(hash_d)$ with help of Table 4.2.

$$P(hash_\infty) = (P_\infty)^3 \tag{4.20}$$

$$P(hash_0) = P_0 + (1 - P_0)P_0 + (1 - P_0)^2 P_0 \tag{4.21}$$

$$P(hash_d) = 1 - P(hash_\infty) - P(hash_0) \tag{4.22}$$

Please note that, like in case of flooding-based search, the failure probability of hash-based search is slightly underestimated.

Next, we determine the CDF of successful delayed hash-based search latency, again conducting a case differentiation: we have to take into account the possibility of immediate deliveries between the three involved vehicles; looking at the delayed successful searches in Table 4.2, we must take into account the possibility of exactly one ($P(1\infty \cap 2d|hash_d)$) and of exactly two ($P(2\infty \cap 1d|hash_d)$) immediate deliveries. The weighed CDF of successful delayed hash-based search latency is then

$$\begin{aligned} P(T_{hash_d} \leq t) = {} & P(2\infty \cap 1d|hash_d)P(T_{hash_{d1}} \leq t) \\ & + P(1\infty \cap 2d|hash_d)P(T_{hash_{d2}} \leq t) \end{aligned}$$

$$+ (1 - P(1\infty \cap 2d|hash_d) - P(2\infty \cap 1d|hash_d))P(T_{hash_{d3}} \leq t). \quad (4.23)$$

Analogusly to flooding, the final CDF of hash-based search latency then calculates to

$$P(T_{hash} \leq t) = P(hash_\infty) + \frac{P(T_{hash_d} \leq t)}{(1 - P(hash_0)) - P(hash_\infty)}. \quad (4.24)$$

Again, we start with the calculation of the conditional probabilities. Analogously to $P(req_d \cap rep_\infty|flooding_d)$, these are

$$P(2\infty \cap 1d|hash_d) = \frac{3(P_\infty)^2(1 - P_\infty - P_0)}{P(hash_d)}, \quad (4.25)$$

$$P(1\infty \cap 2d|hash_d) = \frac{3P_\infty(1 - P_\infty - P_0)^2}{P(hash_d)}. \quad (4.26)$$

In the first case,

$$T_{hash_{d1}} = \frac{D_1}{V_1}. \quad (4.27)$$

In the second case,

$$T_{hash_{d2}} = \frac{D_1}{V_1} + \frac{D_2}{V_2}. \quad (4.28)$$

Otherwise,

$$T_{hash_{d3}} = \frac{D_1}{V_1} + \frac{D_2}{V_2} + \frac{D_3}{V_3}. \quad (4.29)$$

With knowledge of the D_i and V_i, the CDFs of the random variables $T_{hash_{di}}$ are

$$P(T_{hash_{d1}} \leq t) = F_{T_{hash_{d1}}}(t) = \iint\limits_R f_{D_1,V_1}(d_1, v_1)dd_1dv_1,$$
$$\text{with } R = \{(d_1, v_1)|\frac{d_1}{v_1} \leq t\} \quad (4.30)$$

$$P(T_{hash_{d2}} \leq t) = F_{T_{hash_{d2}}}(t) = \iiiint\limits_R f_{D_1,D_2,V_1,V_2}(d_1, d_2, v_1, v_2)dd_1dd_2dv_1dv_2,$$
$$\text{with } R = \{(d_1, v_1, v_2)|\frac{d_1}{v_1} + \frac{d_1}{v_2} \leq t\} \quad (4.31)$$

$$P(T_{hash_{d3}} \leq t) = F_{T_{hash_{d3}}}(t) =$$
$$\int_R \cdots \int f_{D_1,D_2,D_3,V_1,V_2,V_3}(d_1,d_2,d_3,v_1,v_2,v_3)dd_1dd_2dd_3dv_1dv_2dv_3,$$
$$\text{with } R = \{(d_1,d_2,d_3,v_1,v_2,v_3)|\frac{d_1}{v_1}+\frac{d_2}{v_2}+\frac{d_3}{v_3} \leq t\} \quad (4.32)$$

The DHT has to be treated separately because the message AB is delivered using overlay routing: given a stretch s, the distance covered by the message is s times the geographical distance. In case of one infinitely fast delivery, the probability that delivery AB is the infinitely fast one is $\frac{1}{3}$. In case of two infinitely fast deliveries, the probability that delivery AB is one of them is $\frac{2}{3}$. Thus, the weighed CDF of delayed successful DHT-based search latency is

$$\begin{aligned} P(T_{DHT_d} \leq t) = {} & P(2\infty \cap 1d|hash_d)(\frac{2}{3}P(T_{hash_{d1}} \leq t) + \frac{1}{3}P(T_{DHT_{d1}})) \\ & + P(1\infty \cap 2d|hash_d)(\frac{1}{3}P(T_{hash_{d2}} \leq t) + \frac{2}{3}P(T_{DHT_{d2}})) \\ & + (1 - P(1\infty \cap 2d|hash_d) - P(2\infty \cap 1d|hash_d))P(T_{DHT_{d3}} \leq t). \quad (4.33) \end{aligned}$$

The CDF of DHT-based search is then given by

$$P(T_{DHT} \leq t) = P(hash_\infty) + \frac{P(T_{DHT_d} \leq t)}{(1 - P(hash_0)) - P(hash_\infty)}. \quad (4.34)$$

The CDFs of $P(T_{DHT_{di}} \leq t)$ are identical to $P(T_{hash_{di}} \leq t)$, the only difference being the integration range due to the distance increase of factor s in the first delivery.

$$P(T_{DHT_{d1}} \leq t) = F_{T_{DHT_{d1}}}(t) = \iint_R f_{D_1,V_1}(d_1,v_1)dd_1dv_1,$$
$$\text{with } R = \{(d_1,v_1)|\frac{sd_1}{v_1} \leq t\} \quad (4.35)$$

$$P(T_{DHT_{d2}} \leq t) = F_{T_{DHT_{d2}}}(t) = \iiiint_R f_{D_1,D_2,V_1,V_2}(d_1,d_2,v_1,v_2)dd_1dd_2dv_1dv_2,$$
$$\text{with } R = \{(d_1,v_1,v_2)|\frac{sd_1}{v_1}+\frac{d_2}{v_2} \leq t\} \quad (4.36)$$

$$P(T_{DHT_{d3}} \leq t) = F_{T_{DHT_{d3}}}(t) =$$
$$\int_R \cdots \int f_{D_1,D_2,D_3,V_1,V_2,V_3}(d_1,d_2,d_3,v_1,v_2,v_3)dd_1dd_2dd_3dv_1dv_2dv_3,$$
$$\text{with } R = \{(d_1,d_2,d_3,v_1,v_2,v_3)|\frac{sd_1}{v_1} + \frac{d_2}{v_2} + \frac{d_3}{v_3} \leq t\} \quad (4.37)$$

According to our assumption of independence, the joint PDFs in these equations are products of the PDFs of the single random variables.

In the next section we present the results of our simulation study of information propagation speed and the derived CDFs of search latency. Due to the complexity of the resulting integrals, we refrain from calculating analytical solutions and instead give numerical results.

4.2.2 Simulation Study of Information Propagation Speed, Resulting CDF of Search Latency

In this section, achievable information propagation speeds are determined for the sample scenario (see Chapter 3) and the respective CDFs of search latency are derived. The underlying methodology is detailed in Section 4.2.2.1, before results are presented in Section 4.2.2.2 and Section 4.2.2.3.

4.2.2.1 Methodology

The PDF of information propagation speed is the missing piece for computation of the CDF of search latency. The idea behind the simulation study is to obtain samples of information propagation speeds and to assume an underlying standard PDF whose parameters are determined with help of the samples.

We obtained samples by repeatedly selecting a random vehicle that sends a message to another random vehicle in the simulation area. Determining the achievable information propagation speed for a delivery requires a message propagation algorithm that finds the fastest route. In the following we first define an according optimality criterion for the propagation algorithm, then present a message flooding scheme and discuss how closely it approximates an optimal solution.

Definition: We call a message flooding scheme *optimal within a time horizon d* if for each message, all nodes reachable from the message originator during the time horizon d

(through single- or multi-hop communication) receive the message with the respectively shortest possible latency.

b: beacon interval
unknown($Message\ m$): message received for the first time
send($Message\ m, Node\ t, Delay\ w$): defer for w, then send m to t
broadcast($Message\ m, Delay\ w$): defer for w, then broadcast m
advertise($ID\ m.id$): broadcast $m.id$
store($Message\ m$): store in local message queue, advertise in next beacon
timeout($Timer\ b$): beacon timer expired

receiveMessage($Message\ m$)
1: **if** (unknown(m)) **then**
2: broadcast(m, w_2)
3: store(m)
4: **end if**

timeout($Timer\ b$)
1: advertise($m.id$)

receiveAdvertise($ID\ m.id, Node\ s$)
1: **if** (unknown($m.id$)) **then**
2: send($m.id, s, w_1$)
3: **end if**

receiveBroadcastInstruction($ID\ m.id, Node\ s$)
1: send(m, s)

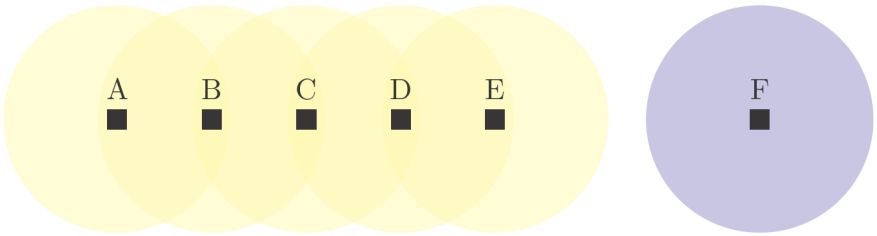
Algorithm 4.1: Advertisement-based message flooding scheme

For our evaluation, we applied Algorithm 4.1, which works as follows. A node advertises a message after each timeout of timer b. Nodes in radio range of the advertiser overhear the advertisement and send a broadcast instruction for the message after a delay of w_1 if it is unknown to them. Upon receiving the instruction, the advertiser sends the message. New owners become advertisers and once broadcast the message after a delay of w_2 so that their respective neighbors receive it.

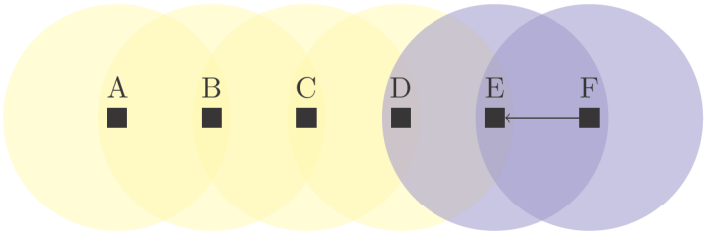
Theorem 4.1. *Assuming a message transmission time of zero (thereby not taking into account possible collisions on the wireless channel), Algorithm 4.1 is optimal for* $w_1 \to 0, w_2 \to 0, b \to 0$.

Proof Since $b \to 0$, all nodes ever in radio range of a message owner receive the advertisement of the message. Upon receiving the advertisement, nodes that do not own a copy of the message send a broadcast instruction with delay $w_1 \to 0$, triggering the

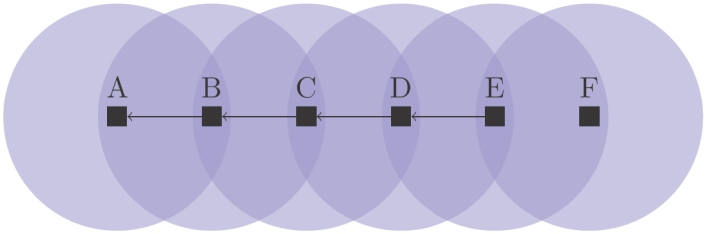
sending of the message at the advertiser (receiveBroadcastInstruction), and receive it (receiveMessage) within zero time, according to our assumption of a message transmission time of zero. Nodes in range of the new owner immediately receive the message, because the new owner broadcasts it with delay $w_2 \to 0$. Consequently, all nodes ever within radio range of a message owner receive the message with the shortest possible delay. ⊔



(a) Vehicle F carrying a message unknown to the partition of vehicles A to E



(b) Inter-partition forwarding: vehicle F forwards the message to vehicle E



(c) Intra-partition forwarding: message is forwarded along the chain from vehicle E to vehicle A

Figure 4.8: Example of inter- and intra-partition forwarding

Figure 4.8 illustrates the arguments of the proof. Here, message transmissions are classified into inter-partition- and intra-partition-forwarding. As soon as vehicle F is in range of vehicle E, E overhears the advertisement, returns a broadcast instruction and receives the message. Immediately afterwards, it broadcasts the new message. This triggers further broadcasts by new receivers, causing the message to propagate up to vehicle A without delay.

The theorem makes an important claim regarding the optimality of Algorithm 4.1. The

algorithm will be reused for evaluation of the ideas presented in Chapter 5 and Chapter 6 because it performs near-optimal for low values of w_1, w_2, and b.

For practical use, of course $b > 0$. Likewise, $w_1, w_2 > 0$ to avoid collisions with neighbors that received the message or message advertisement. For our simulation study, we selected b randomly and uniformly distributed over $[0.375\,\mathrm{s}; 1.125\,\mathrm{s}]$, w_1 and w_2 randomly and uniformly distributed over $[0\,\mathrm{s}; 0.1\,\mathrm{s}]$ (the reason for the random jitter is to avoid repetitive collisions between synchronized vicinal nodes [CDA08]). We set $d = 500\,\mathrm{s}$. We simulated penetration rates of 5 %, 10 %, and 15 %. During each simulation run, we obtained 600 samples. For each penetration rate, we conducted three of these simulation runs.

The following graphs show for each penetration rate the EDF of those samples with $0 < \overline{V}_j < \infty$. In the same graph, there is the CDF of a log-normal distribution whose parameters μ and σ are determined through a maximum likelihood estimation. Given n samples $v_1, \ldots, v_n$ of information propagation speed, the estimators are [LSA01]:

$$\hat{\mu} = \frac{\sum_{k=1}^{n} ln(v_k)}{n}; \hat{\sigma}^2 = \frac{\sum_{k=1}^{n} (ln(v_k) - \hat{\mu})^2}{n} \tag{4.38}$$

The plotted log-normal distribution uses as μ and σ the average values over the maximum likelihood estimators of each seed. These graphs are followed by graphs of the computed CDF of search latency.

For calculation of the latency for DHT-based search, a quantification of the routing stretch in a VANET is required. Since we could not find any values for VANETs in the literature, we resort to published results for MANETs. Here, Cramer et al. presented evaluations regarding stretch [CF05], enhancing Chord [MKKB01] with proximity neighbor selection. They measured a stretch varying between 2.5 and 4.5 depending on the density of the network. We here assume a stretch of $s = 2.5$ as optimistic value.

4.2.2.2 Results, pure VANET

Looking at Figures 4.9, 4.10, and 4.11, it turns out that information propagates rather slowly through the VANET (with less than 25 m/s in more than 80 % of all cases at 5 % penetration rate, and with still less than 50 m/s in more than half of all cases at 15 % penetration rate).

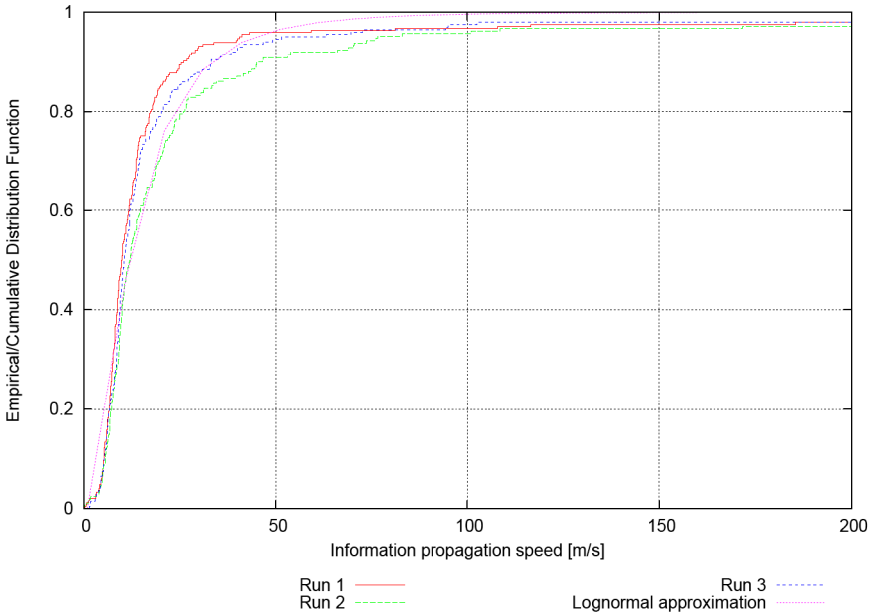


Figure 4.9: EDF: information propagation speed; log-normal approximation, 5 % penetration rate

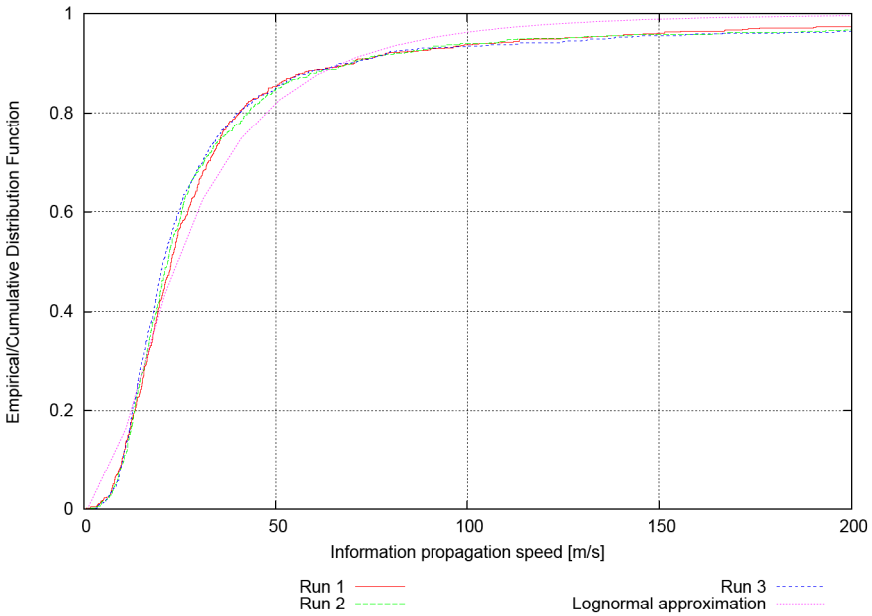


Figure 4.10: EDF information propagation speed; log-normal approximation, 10 % penetration rate

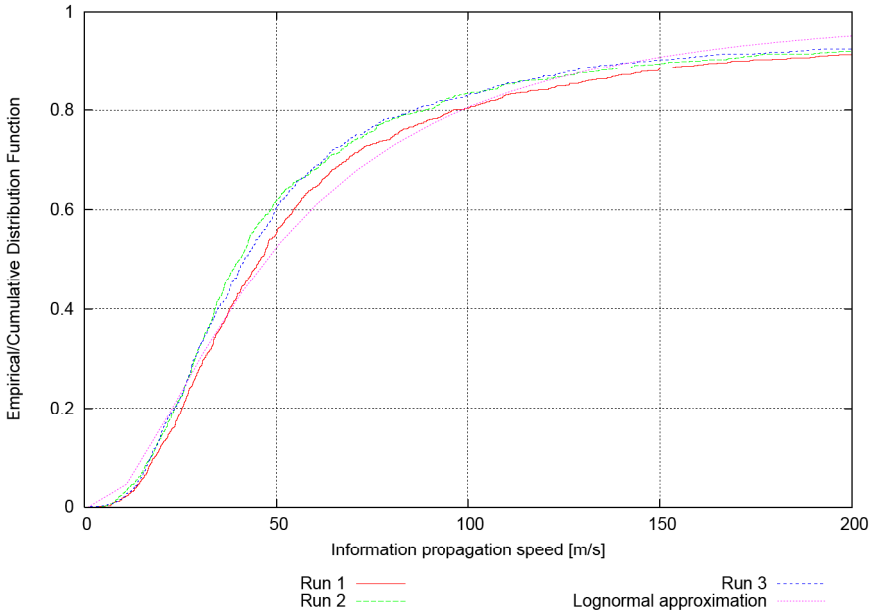


Figure 4.11: EDF: information propagation speed; log-normal approximation, 15 % penetration rate

Penetration rate	μ	σ	P_0	P_∞
5 %	2.474	0.644	58.794 %	4.266 %
10 %	3.179	0.638	33.736 %	5.533 %
15 %	3.857	0.751	19.989 %	9.315 %

Table 4.3: Log-normal parameters μ, σ; probability of delivery failure (P_0), probability of immediate delivery (P_∞)

Table 4.3 shows the values of μ and σ for each penetration rate, along with the probabilities P_0 and P_∞. The sparseness of the VANET leads to delivery failures in about 59 % of all cases at 5 % penetration rate and still about 20 % of all cases at 15 % penetration rate. The low values of P_∞ indicate that there are only few multi-hop deliveries.

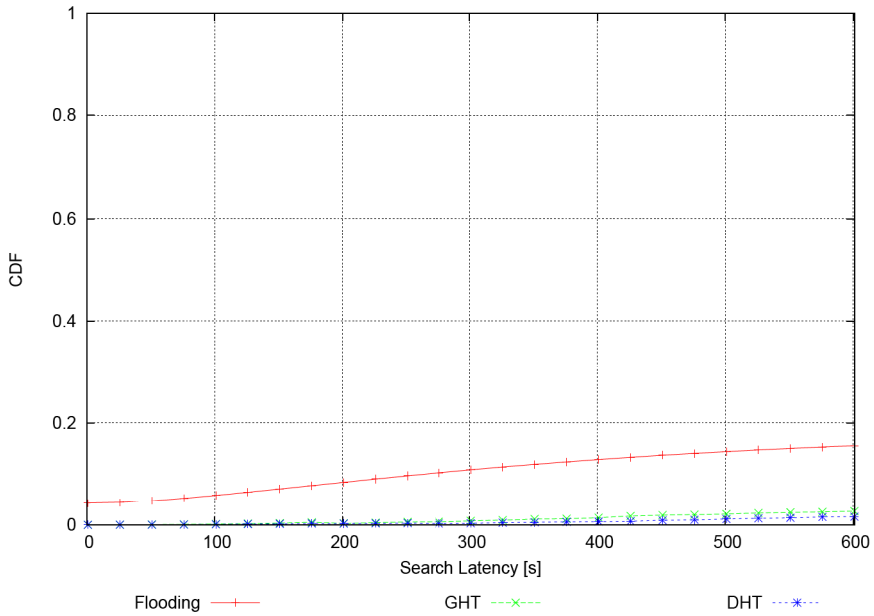


Figure 4.12: CDF: search latency, 5 % penetration rate

Figures 4.12, 4.13, and 4.14 show the calculated CDFs of search latency. Especially at 5 % and 10 % penetration rate, the communication latency is so high that most searches fail. Since the message delivery algorithm is near-optimal, these failures must result from the network topology. In the next section we present results of the same experiment in a slightly modified network: interconnected base stations are placed in the metropolitan area to improve communication.

4.2.2.3 Scenario with base stations

To improve information propagation speed, we placed interconnected base stations in the examined scenario and repeated the experiment. Regarding the number and placement of the base stations, our goal was to evaluate a realistic setting: placement of base stations provided by industry or public authorities underlies financial and geographical restrictions. There have been studies on the placement and effect of base stations, i.e. the work by Lochert et al. [LSCM07, LSW+08]. Following their finding that placing base

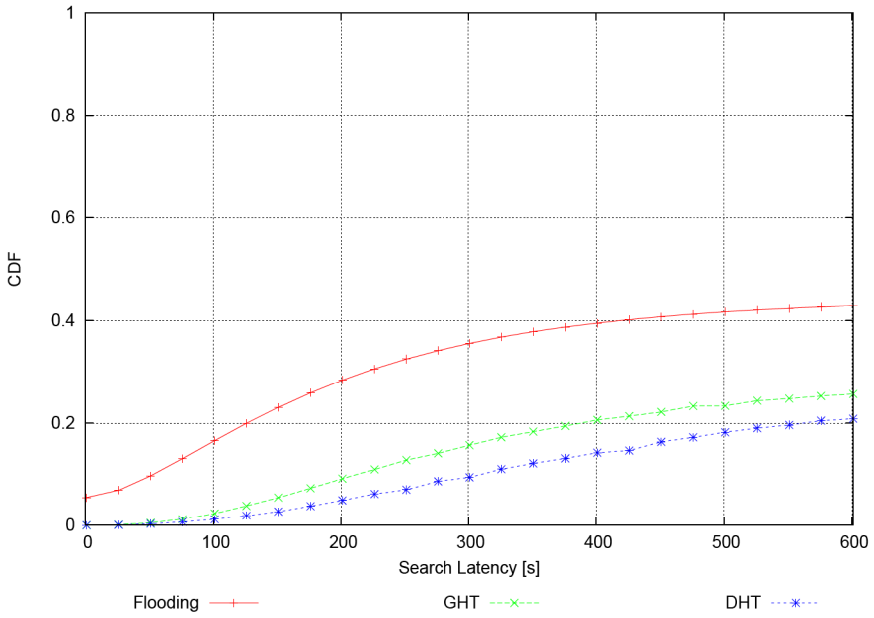


Figure 4.13: CDF: search latency, 10 % penetration rate

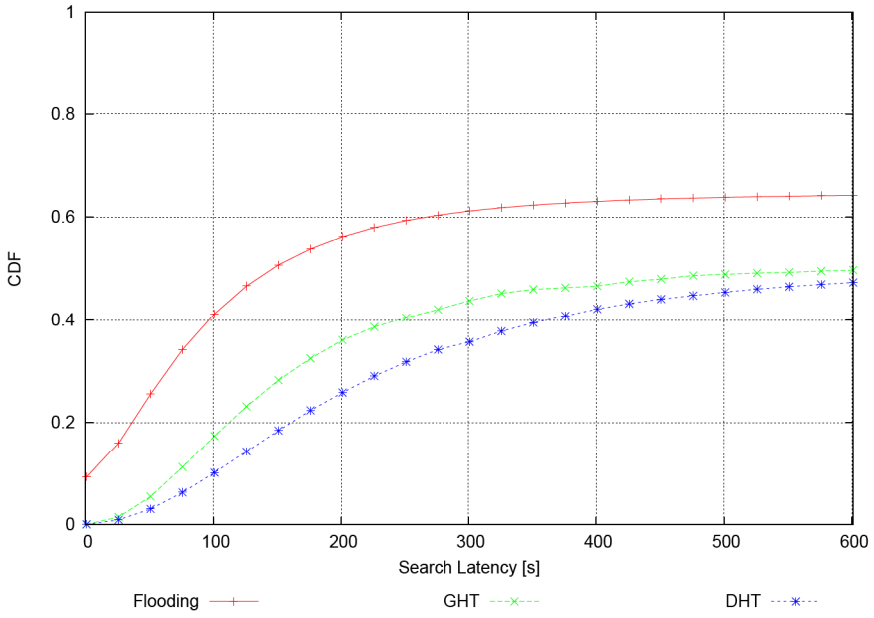


Figure 4.14: CDF: search latency, 15 % penetration rate

stations at intersections with high traffic density is a good heuristic, we placed nineteen base stations at the largest intersections of the metropolitan area (Figure 4.15).

It is generally not advisable to place only a small number of interconnected base stations in an area if they support communication between distant network nodes: network traffic flows towards these base stations, making them "hot spots" of traffic. With more base stations that are spatially distributed, network traffic is better spatially distributed as well.

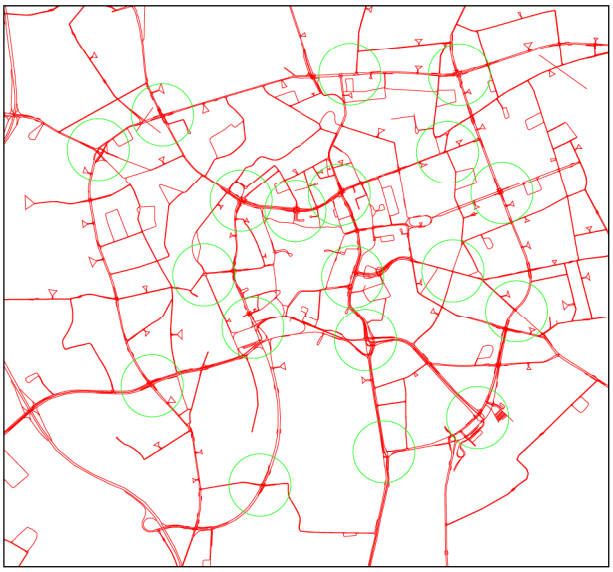


Figure 4.15: Scenario with base stations

To first give the reader a general idea how communication in the scenario is affected by the added base stations, we now introduce several metrics and their values when determined for the sample scenario.

4.2.2.3.1 Placement metrics The first metric is the EDF of *connection ratio*, which is defined as the fraction of presence time of a vehicle, during which it is in radio range of a base station. For our scenario, the EDF is given in Figure 4.16. It can be seen that 10 % of all vehicles are never in radio range of any base station. About 30 % of the vehicles are in range of a base station for more than half of their presence time. This shows that only a minor part of the area is covered by the base stations.

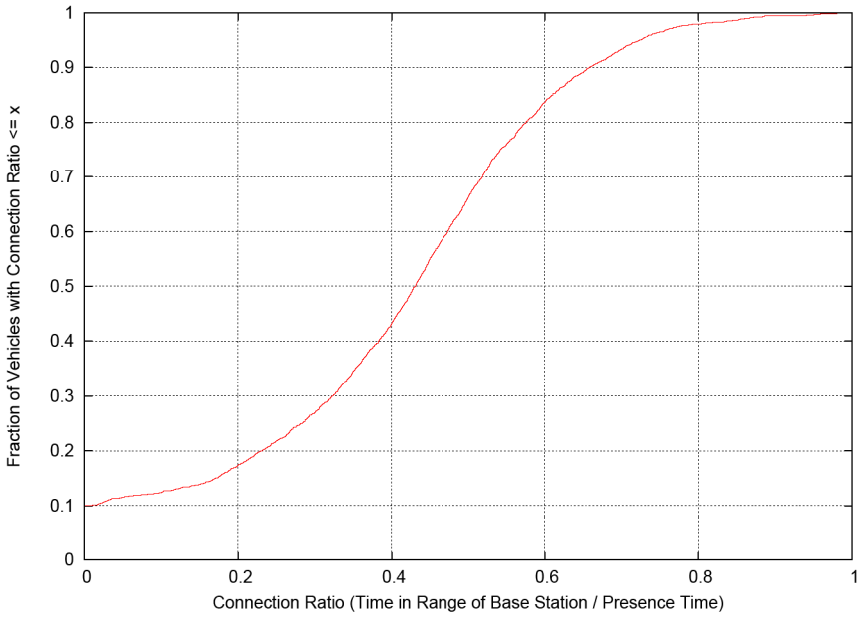


Figure 4.16: EDF: connection ratio

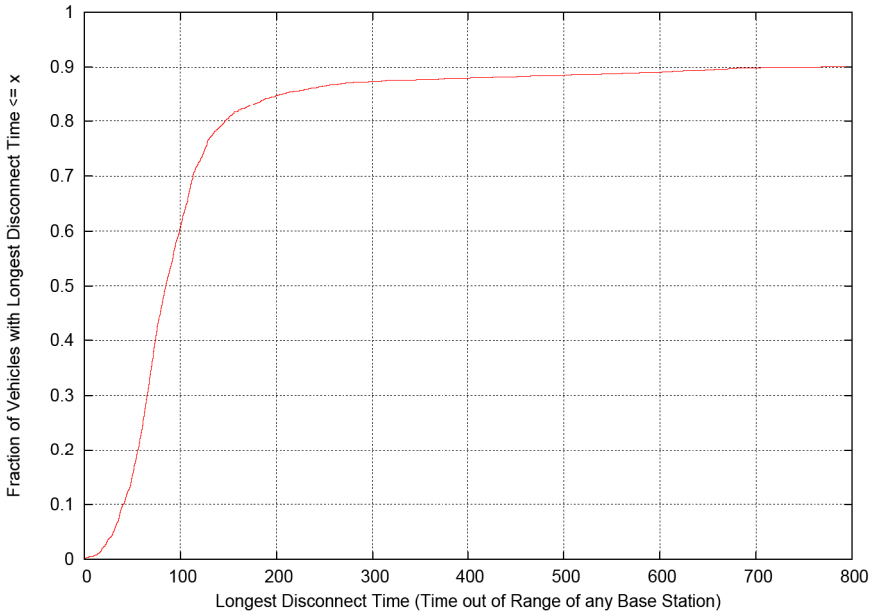


Figure 4.17: EDF: longest disconnect period

The second metric is the EDF of *the longest disconnect period*, the values being shown in Figure 4.17 for our scenario. 50 % of vehicles have a disconnect period of more than 90 s. Of course, 10 % of vehicles have no connection at all.

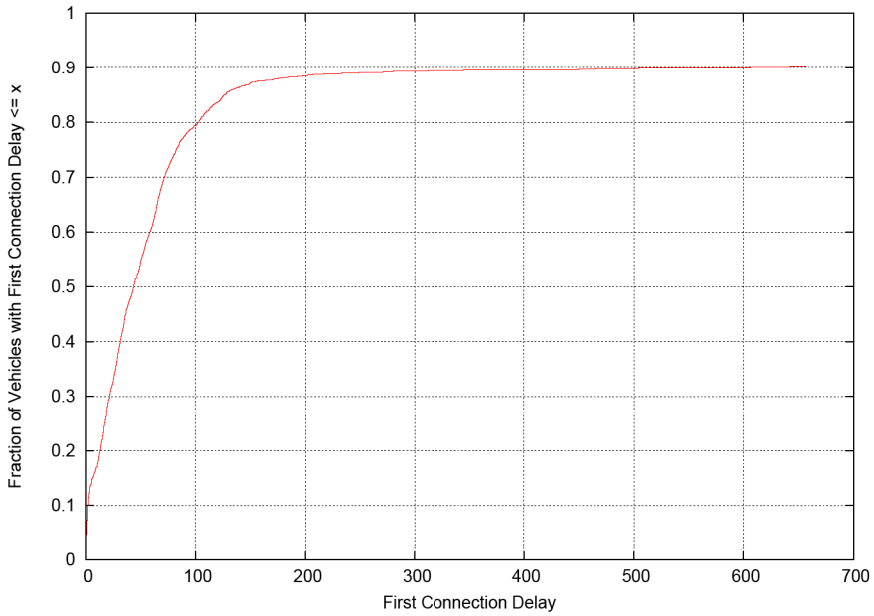


Figure 4.18: EDF: delay until first connection

The third metric is the EDF of *delay until the first connection to a base station*. Figure 4.18 shows 50 % of the vehicles are in range of the first base station in less than 50 s after joining the scenario.

The fourth metric is the EDF of the *total number of reached base stations*. In our scenario, about 55 % of vehicles are in contact with only two or less different base stations (Figure 4.19).

Penetration rate	μ	σ	P_0	P_∞
5 %	4.383	1.196	13.727 %	43.349 %
10 %	4.478	1.137	11.252 %	53.361 %
15 %	4.597	1.039	9.274 %	59.407 %

Table 4.4: Scenario with 19 base stations: log-normal parameters μ, σ; probability of delivery failure (P_0), probability of immediate delivery (P_∞)

4.2.2.3.2 Results Table 4.4 shows the determined parameters of the distributions as well as the probabilities of failed and multi-hop deliveries. Compared with Table 4.3,

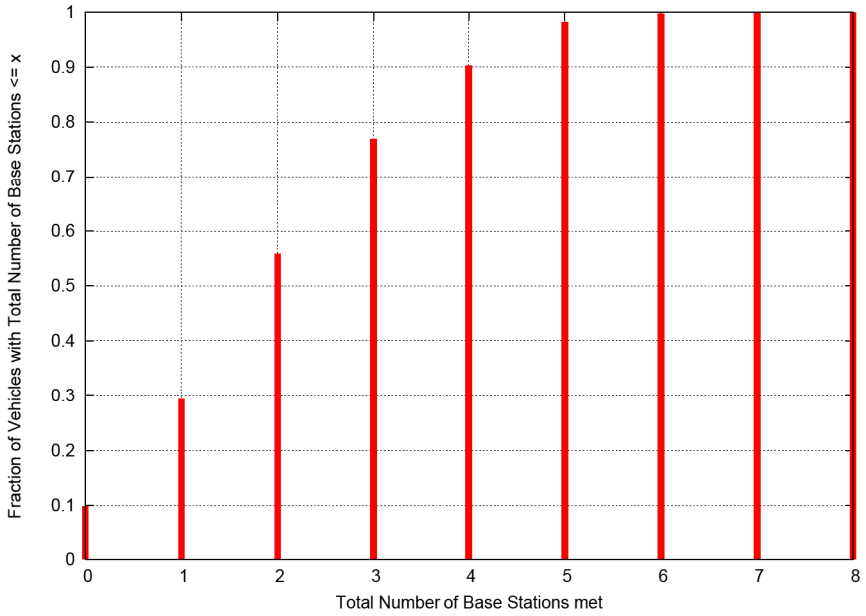


Figure 4.19: EDF: total number of reached base stations per vehicle

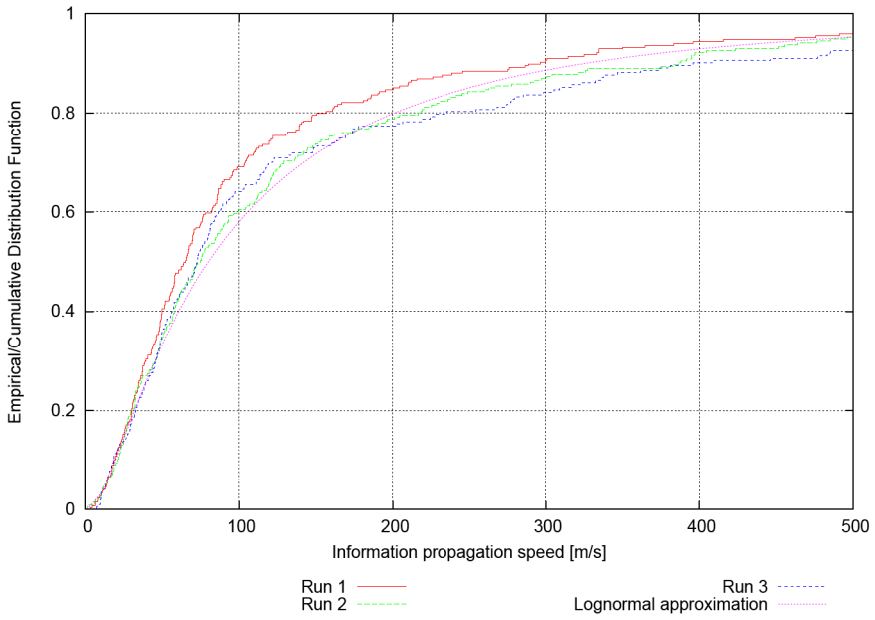


Figure 4.20: EDF: information propagation speed; log-normal approximation, 5 % penetration rate, 19 base stations

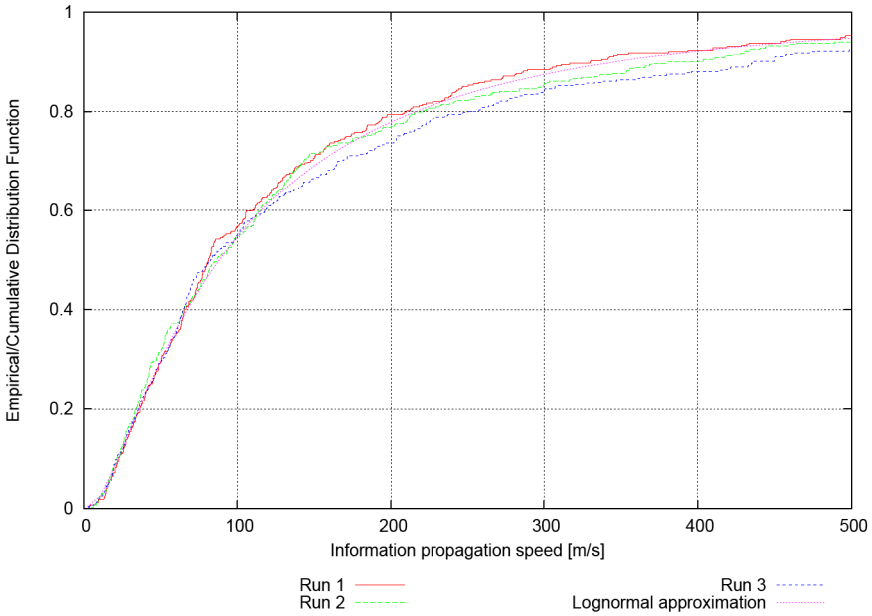


Figure 4.21: EDF: information propagation speed; log-normal approximation, 10 % penetration rate, 19 base stations

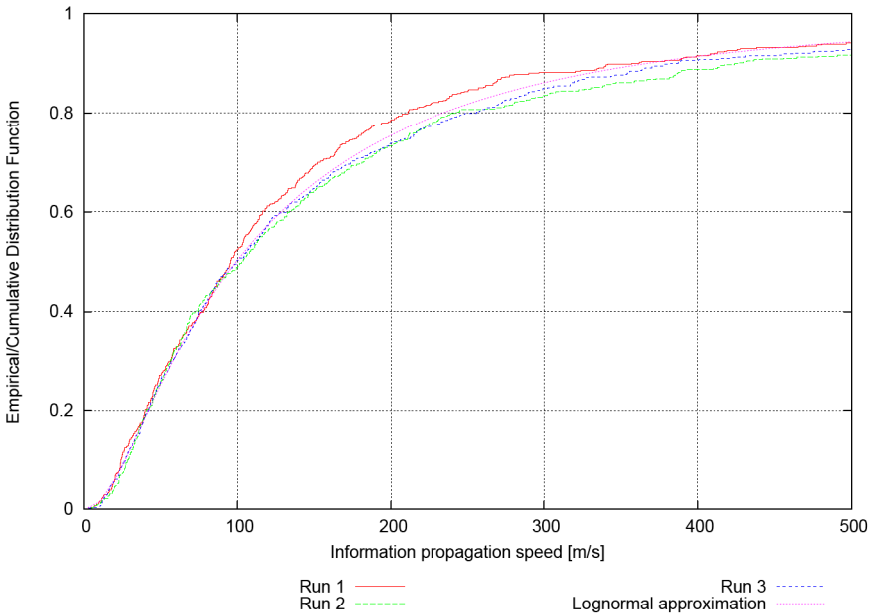


Figure 4.22: EDF: information propagation speed; its log-normal approximation, 15 % penetration rate, 19 base stations

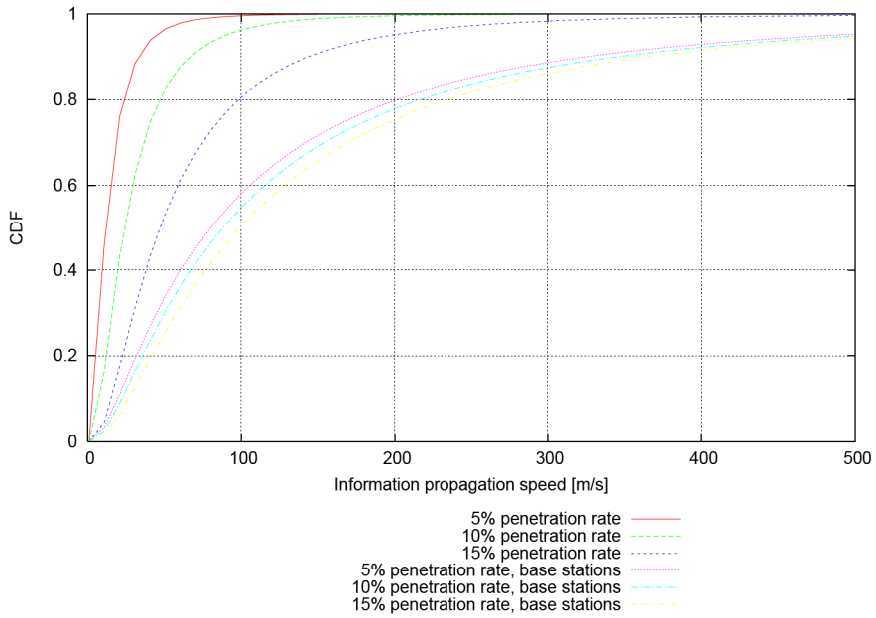


Figure 4.23: CDFs: log-normal approximations of 5%, 10%, and 15% penetration rate with and without base stations

failed deliveries are considerably less likely and the chance of multi-hop deliveries has strongly increased. The EDFs of information propagation speed are depicted in Figures 4.20, 4.21, and 4.22. Figure 4.23 shows the CDF of all log-normal approximations in one graph, with and without base stations. Clearly, the interlinked base stations strongly increase propagation speeds.

Figures 4.24, 4.25, and 4.26 show the calculated CDFs of search latency. The base stations have a very positive effect on search latency at low penetration rates: especially in the cases of 5 % and 10 % penetration rate, latency reduces drastically compared with the scenario without base stations. Because of the observed improvement, we will continue our work on a search method under the assumption that base stations support the communication in the VANET.

4.3 Discussion of Network Load

Network load of search should be minimized to enable coexistence with other applications in the VANET. In this section we qualitatively discuss the three search methods regarding their network load. A quantitative comparison of network load by flooding and index-based search is conducted in Section 6.4.2 of this thesis.

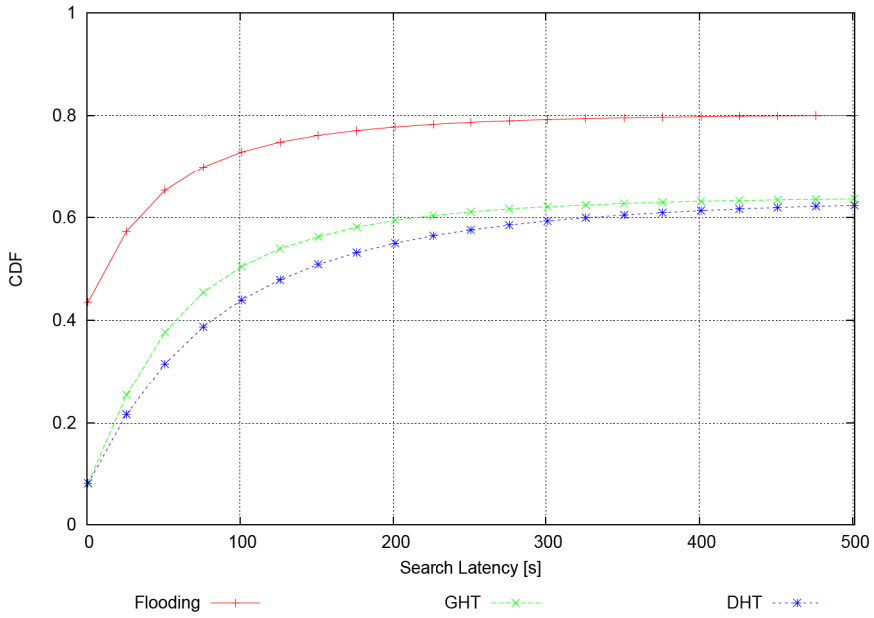


Figure 4.24: CDF: search latency, 5 % penetration rate, base stations

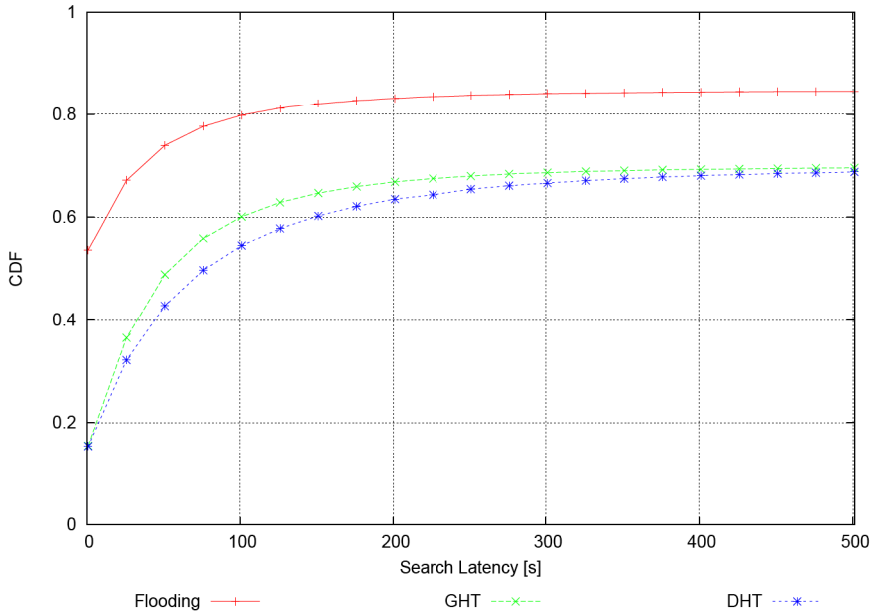


Figure 4.25: CDF: search latency, 10 % penetration rate, base stations

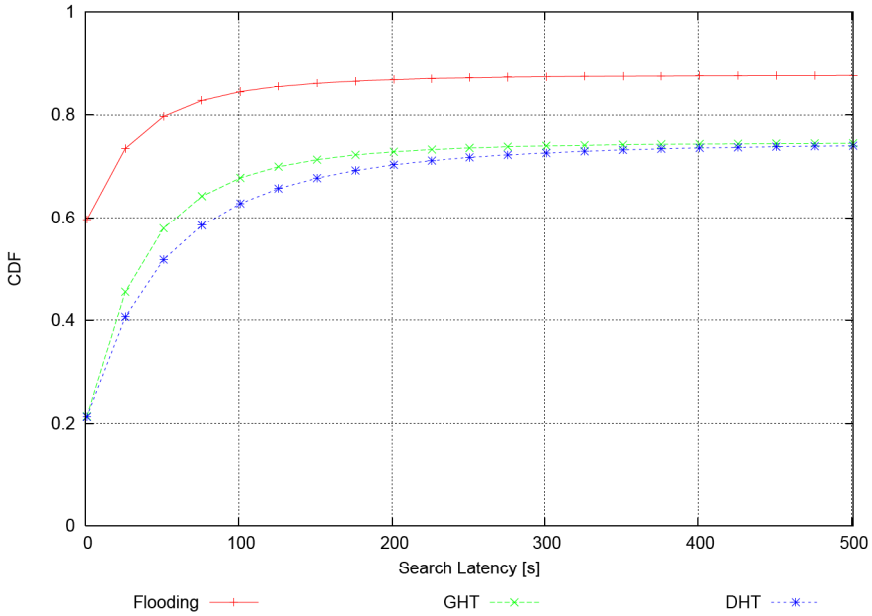


Figure 4.26: CDF: search latency, 15 % penetration rate, base stations

Flooding requires no organization of the VANET and therefore induces no network load for index maintenance. Nevertheless, vehicles flooding content requests can cause high area-wide network load because the request is likely to be repeated by vehicles in all regions of the metropolitan area. Therefore, among all search methods, flooding is expected to cause by far the most network load in sending the content request to an information source. Still, if the query rate in the network is low and there is a high possibility of finding an information source nearby, limited flooding (like expanding ring search [PR99]) is a reasonable search method.

In contrast to flooding, organizing the VANET by means of a GHT requires maintenance. Vehicles have to register their keys and positions at a certain area to make their information items available for search. Additionally, it is necessary to regularly send position updates as vehicles are moving. On the positive side, the key advantage of GHT-based search over flooding is that content requests are directed to a certain location in the network.

Organization by means of a DHT requires the highest amount of maintenance traffic as each vehicle has to maintain neighbor tables containing positions of its overlay neighbors. Search messages are overlay-routed to a vehicle knowing the location of the requested content, which implies that the request may zigzag the VANET. Therefore it is slower than a GHT and also causes higher network load.

We conclude that from a network load perspective a GHT-based approach is preferable if the search request rate justifies the maintenance overhead. Flooding should be avoided because of its high network load.

4.4 Conclusions

Our analysis of the computed CDFs of search latency has indicated that flooding-based search has a clear edge over the investigated hash-based search methods. But even at 10 % penetration rate without base stations, the expected success rate of flooding is only about 45 %. This result has motivated the second set of experiments where base stations were deployed in the scenario. This improves communication quality and consequently increases the expected success rate to more than 80 %, again considering flooding at 10 % penetration rate. However, the deployed base stations are not capable of closing the gap between the success rates of flooding and the hash-based search methods.

On the other hand, from a network load point of view, flooding-based search should be avoided whenever possible. Thus, none of the investigated search methods offers a ready-to-use solution in the investigated scenario. Nevertheless, as an important result, the presented feasibility study gives an indication of the properties and boundaries of an ideal VANET-based search method. It should have a required communication distance which is almost as low as with flooding. It should have a search network load as low as that of GHT-based search, with lowest possible maintenance effort. These findings guide our proposal of a VANET-based search method in Chapter 6.

4.5 Chapter Summary

In this chapter, the success rates of flooding-, GHT- and DHT-based search have been calculated. This calculation has combined a stochastic model of success rate and latency of the search schemes with a simulation study of information propagation speed. In this context, a message propagation algorithm with near-optimal delivery latency and delivery rate has been defined.

The index-based search schemes have a significantly lower expected success rate than flooding. Another main result is that without the support of base stations, high message delivery latency causes most searches to fail, no matter which search method is applied. Adding a number of base stations to the scenario, the failure rate of search is significantly

reduced. In practice, search success rate can be expected slightly lower than calculated, due to a slightly optimistic assumption in the analysis.

A further result is that either due to low success rate or due to high network load, none of the investigated search methods is a ready-to-use solution under the investigated conditions. Nevertheless, the observations of the behavior of the investigated search methods provide important groundwork for design of a better algorithm (Chapter 6).

Chapter 5

Routing in Infrastructure-supported VANETs

Content location and querying can be regarded as an application-supporting service or as an application itself and is therefore implemented in communication layers above the network layer. To enable the evaluation of the content location service, the functionality of the underlying layers has to be implemented as well. The implementations of physical layer and MAC layer have been presented in Chapter 3. In this chapter, we present the remaining piece: a routing-algorithm for infrastructure-supported VANETs in the network layer.

Routing problems in communication networks may be classified depending on the number of target nodes: unicast (one target node), multicast (more than one target node), broadcast (each network node is a target), and anycast (one out of many possible target nodes). In infrastructure-supported VANETs, the unicast routing problem occurs whenever the target of a message is a specific vehicle. The anycast routing problem occurs whenever the target of a message is a base station, since it is sufficient to reach any base station and use the backbone network to reach a specific one. In this chapter, solutions to the unicast and anycast routing problems are proposed. The quality of the solutions is assessed in a simulative comparison with the near-optimal message flooding scheme presented in Section 4.2.2.

Routing includes vehicles and base stations, both being referred to as *nodes*. We define the unicast routing problem as *delivering a message from a sender node to a target node in the fastest possible time, using the least possible number of intermediate nodes*[1]. We define the anycast routing problem as *delivering a message from a sender node to the*

[1]If the number of involved nodes was not an issue, one could simply obtain a solution by flooding messages to all vehicles.

node out of a set of possible target nodes, which is reachable in the fastest possible time, thereby involving the least possible number of intermediate nodes.

This chapter continues with a discussion of related work on unicast and anycast routing (Section 5.1), which is followed by a classification of routing situations that occur in infrastructure-supported VANETs (Section 5.2). Based on these observations, challenges for a routing algorithm are identified (Section 5.3). An algorithm tackling the challenges is then proposed (Section 5.4) and evaluated (Section 5.5). The chapter ends with conclusions (Section 5.6) and a summary (Section 5.7).

5.1 Related Work

This section starts with a review of research results on unicast routing in VANETs (Section 5.1.1). Thereafter, related work on anycast is surveyed (Section 5.1.2).

5.1.1 Unicast Routing

When nodes select a suitable next hop of a message, they consult their local knowledge base of the network topology. Fall et al. present a classification of delay-tolerant network (DTN) routing algorithms based on the respective knowledge base in [JFP04]. We here survey VANET routing algorithms and classify them in similar style (cf. Figure 5.1). There exist routing algorithms where nodes only need to know their position and the position of the target. Other routing algorithms additionaly presume knowledge of nodes' single-hop neighbors. Another class consists of those algorithms requiring supplemental knowledge of road topology. The most-knowledgeable algorithms additionally take into account the traffic situation. Before analyzing algorithms within the single classes, it is worth mentioning that routing in VANETs is a heavily studied field of research, which led to the publication of several survey papers [BM09, LW07, LH04, CVCAC06]. We here focus on unicast routing.

Contention-Based Forwarding (CBF) [FKM$^+$03, FHM$^+$04], is a beacon-less routing algorithm belonging to the class of algorithms with the least knowledge. The core idea of CBF is to introduce a contention period during which all receivers of a message compete to be the next forwarder. The contention period after which a receiver further forwards a message is inversely proportional to its distance to the target. Overhearing this forwarding, the other candidate forwarders suppress the message. CBF achieves good performance especially in open terrain, but the overhearing of forwarded messages

Figure 5.1: Knowledge-based classification of routing algorithms

is difficult in metropolitan areas with a high density of radio obstacles. If other receivers miss a forwarded message, duplicates occur in the VANET. To avoid these duplicates, the authors suggest active suppression and forwarder selection strategies, inducing additional control overhead. While the biggest advantage of CBF is the non-necessity of beacons, high-frequent periodic vehicle beaconing is the groundwork of many safety applications in VANETs and is regarded as a main application field of VANET technology.

Our routing algorithm relies on neighbor tables based on periodic vehicle beacons, and therefore belongs to the class of algorithms with next-higher knowledge level. Our work is closely related to Cached Greedy Geocast (CGGC) [MES04]. CGGC is a greedy, position-based unicast routing algorithm with the same knowledge base as our algorithm. In CGGC, when a node finds no forwarder closer to the destination (local minimum), it locally caches the message, as does our algorithm. CGGC is developed for infrastructure-less VANETs and consequently does not address the anycast problem occurring when a base station is targeted. In this case, our algorithm opportunistically alters the target base station. Besides CGGC, there exist several MANET routing algorithms in this class, but applying them without modification in VANETs leads to suboptimal results especially in city scenarios; Lochert et al. in [LHT+03] show this for Greedy Perimeter Stateless Routing (GPSR) [KK00], Ad-hoc On-demand Distance Vector Routing (AODV) [PR99] and Dynamic Source Routing (DSR) [JM96]. As improvement, Lochert et al. proposed Greedy Parameter Coordinator Routing (GPCR) [LMFH05], a position-based greedy routing protocol. In GPCR, routing decisions are made only by coordinator vehicles, which are the ones located on intersections. Vehicles determine if they are at an intersection with help of a detection algorithm that is based on their neighbors' beacon messages. We do not apply this algorithm since it seems unlikely that the intersection detection is reliable in sparse VANETs. Furthermore, in GPCR, a message is first routed to a vehicle on an intersection and then to the vehicle geographically

closest to the target. This may cause additional hops for the message.

Geographic Source Routing (GSR) [LHT+03] is a variant of GPCR including knowledge of the road topology. Based on this topology, the shortest path in terms of a list of intersections to be reached is calculated a priori and encoded into the packet. Like GPCR, GSR may induce additional message hops. To avoid these additional hops, Lee et al. proposed GPSRJ+ [LHLG07], where vehicles predict and directly address the vehicle to which the coordinator would forward the message. This requires that vehicles have knowledge of the street map.

None of the aforementioned approaches takes into account the traffic situation. Such knowledge would be helpful in situations as shown in Figure 5.6, where the geographically shortest path is not the fastest because there exists another path with a more dense communication network. These paths can be found by the routing protocol Vehicle-Assisted Data Delivery (VADD) [ZC06], which presumes that all vehicles have knowledge about road topology and the current traffic situation. In VADD, each vehicle evaluates this information to compute message delivery latency on each considered road segment, as well as to compute probabilities that messages can make required turns at intersections. This leads to a high delivery rate at low latency.

5.1.2 Anycast Routing

The term *anycast* originates from the Internet domain [PMM93]. Here, a single anycast address identifies a set of target nodes, out of which the best one according to a certain metric is chosen as target of a packet. Anycast is used to reliably provide services [AL06], since it is capable of providing automatic failover.

Robust services and therefore anycast are naturally important for MANETs, as the underlying communication network is highly dynamic. Instead of developing new anycast algorithms, Park and Macker propose techniques to make existing MANET routing algorithms anycast-capable [PM99a, PM99b]. The key idea of their enhancement is the insertion of a virtual node into the network graph. This virtual node is connected to all service-providing nodes, allowing link-state, distance-vector and link-reversal-based routing algorithms to compute the shortest path. Anycast extensions for AODV and DSR were defined by Wang et al. [WZJ03b, WZJ03a]. Because of the previously mentioned differences between MANETs and VANETs, it is unlikely that these approaches work efficiently in our target scenarios.

Lenders et al. point out that in highly dynamic networks, where routes to anycast servers change rapidly, it is promising to include the density of anycast servers into the—otherwise purely proximity-based—routing decision [LMP06]. While the density of servers seems an interesting criteria for routing, we think that the density of traffic in VANETs plays a more important role: routing towards an area containing multiple servers will fail if the communication network on the way is too sparse.

Since the knowledge base of our routing algorithm is limited to the locations of single-hop neighbors and base stations, we rely on a proximity-based opportunistic anycast: vehicles route anycast messages to their closest base station. This implies a change of the target when a vehicle physically carrying a message moves away from its previously closest base station and approaches a different one.

5.2 Classification of Routing Situations

In the following, a classification of *routing situations* in an infrastructure-supported VANET is presented. The reason for distinguishing between routing situations is that each of them has special properties that should be considered in an appropriate routing algorithm. The classification is based on originator and target type of a message. This way, four different routing situations are distinguished, namely *vehicle-to-base station*, *base station-to-vehicle*, *vehicle-to-vehicle*, and *base station-to-base station*.

The delivery of messages originating from a vehicle and being addressed to another vehicle may consist of a sequence of such routing situations: if routing the message through the backbone network reduces the communication distance in the wireless domain, the sequence vehicle-to-base station—base station-to-base station—base station-to-vehicle occurs.

Base station-to-base station routing uses the backbone network. In this thesis, we assume an idealized backbone network in the sense that all messages are delivered with zero latency. Thus, we do not further consider this routing situation.

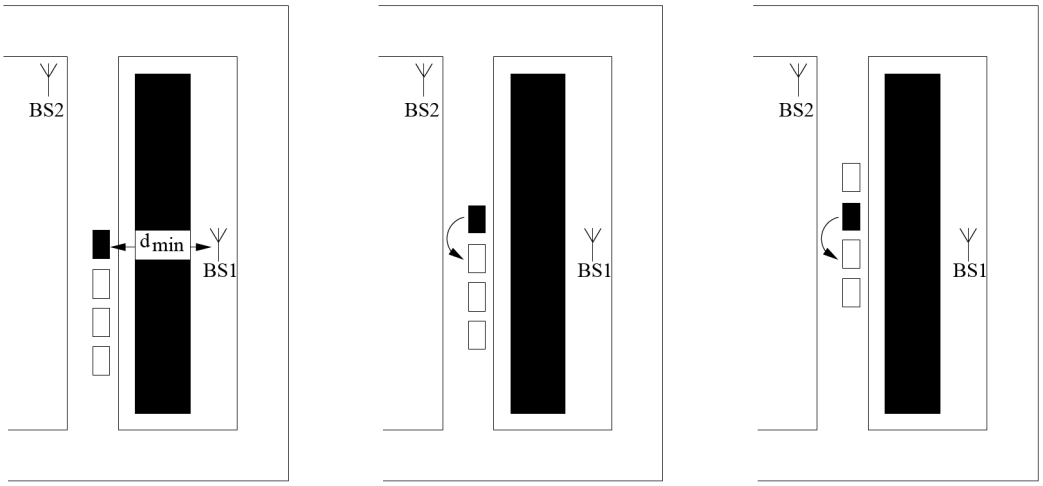
In case of vehicle-to-base station routing, the routing target is static and always online in the network. Furthermore, because of the properties of the backbone network, messages in this routing situation do not have to arrive at a certain predefined base station. They may reach *any* base station, from where they are forwarded to the intended base station immediately.

In the base station-to-vehicle routing situation, the routing target is mobile and may leave or already have left the VANET. Thus, we expect a slightly lower success rate than in the previous situation.

Vehicle-to-vehicle routing happens if routing a message through the backbone network does not reduce the communication distance in the wireless domain.

5.3 Challenges for Routing

We start our considerations having in mind a beeline distance-based greedy routing strategy. Each message contains the ID and position of the target node. When a node is to forward a message, it selects the neighbor that is closest to the target node in terms of beeline distance. When routing a message through a sparse VANET in this way, most likely there is no end-to-end multi-hop routing path between sender node and target node. Thus, at some intermediate node, the message cannot be forwarded closer to the destination. Therefore, nodes need to cache and physically carry it until a better node comes in range (store-and-forward).



(a) Black vehicle holds message targeted to base station BS1. Message is at local minimum distance d_{min} to destination.

(b) Vehicle notices that following vehicle is closer to destination than itself and forwards the message.

(c) Process repeats and message would be stuck if chain of vehicles continued.

Figure 5.2: Local minimum in beeline distance-based greedy routing

Routing messages in this manner, three important challenges have to be considered. Figure 5.2 illustrates the first of these: the possibility of local minima. The depicted

vehicle wants to send a message to BS1. Due to the radio obstacle, it cannot transmit the message. The problem for routing is that the message is currently at a minimum distance from BS1: in whichever direction the vehicle moves, it will move away from the target. When it moves from its current location and detects other vehicles near its old position that are closer to the target than itself, the message will be stuck.

The second challenge is the aging position information of target vehicles. Whenever the target node is a vehicle, its position changes during the time it takes to route the message through the VANET. Thus, the higher the latency of message routing, the more inaccurate becomes the contained position information of the vehicle. Without any knowledge about the new location of the vehicle, the message cannot be delivered.

The third challenge is that the high density of radio obstacles in VANETs increases the risk of faulty message transmissions even though the sender selects a seemingly valid neighbor from its neighbor table.

5.4 Proposed Routing Algorithm

In this section, an extended greedy algorithm is presented, seeking to minimize the beeline distance to the target node. In this context, it is explained how the algorithm combats the aforementioned challenges. At first, the knowledge base required by the algorithm is outlined in Section 5.4.1 before the algorithm itself is discussed in Section 5.4.2.

5.4.1 Knowledge Base

Our routing algorithm requires the following information to be available for each node:

- Current own position—determined e.g. by an onboard GPS device.
- ID and position of single-hop neighbors—extracted from periodic beacons. Each node sends a beacon randomly and uniformly distributed over the interval $[0.5 \cdot t_{BeaconInterval}, 1.5 \cdot t_{BeaconInterval}]$. A node removes a neighbor from its neighbor table if it has not received a new beacon after $1.5 \cdot t_{BeaconInterval}$. Alternatively, the node could wait for a longer period of time (e.g. another $1.5 \cdot t_{BeaconInterval}$) before removing the neighbor. We decided against this because otherwise neighbors that have just recently left the radio range of a node would still be considered valid and might erroneously be selected as best next hop of a message.

- Positions of base stations—information about the base station locations could be included in a static digital map, or, for a reasonable number of base stations, be distributed among vehicles proactively with negligibly small network load.

- ID and position of target node

5.4.2 Algorithm

receive($Message\ m, Node\ s$)

```
 1: if previouslyReceived(m) then
 2:    drop(m)
 3: else if ¬isBroadcast(m) then
 4:    if m.target.ID = me.ID then
 5:       handle(m) {Message handed to application layer}
 6:    else
 7:       m.dmin ← dist(me, m.target)
 8:       route(m)
 9:    end if
10: else if dist(me, m.target) ≤ broadcastRadius then
11:    broadcast(m)
12: else {Broadcast message received outside broadcastRadius}
13:    drop(m)
14: end if
```

route($Messagem$) {Called by *receive* and on neighbor table change}

```
 1: if dist(me, m.target) ≤ broadcastRadius then
 2:    broadcast(m)
 3: else
 4:    if isBaseStation(m.target) ∧ myClosestBaseStation ≠ m.target ∧
       dist(me, myClosestBaseStation) < m.dmin then
 5:       m.target ← myClosestBaseStation
 6:    end if
 7:    m.dmin ← min(m.dmin,dist(me, m.target))
 8:    bestneighbor ← closestNeighbor(m.target.position)
 9:    if dist(bestneighbor, m) < m.dmin then
10:       send(m, bestneighbor)
11:    else
12:       store(m) {Cache message and wait for suitable neighbor}
13:    end if
14: end if
```

Algorithm 5.1: Routing

We propose Algorithm 5.1, which is an extended greedy routing scheme. Each message contains a field d_{min}, indicating its current minimum distance to the target. When a node finds no neighbor closer to the target, it locally caches the message (line 12 of route function). Whenever there is a change in the routing table, the route function is called for each locally stored message. This method of resolving local minima is more flexible than the right-hand rule of GPSR, and has been shown to perform superior in highly mobile environments [LHT+03, MES04].

To overcome the challenge of reaching a moving target, we follow the recommendation of Zhao et al. [ZC06], who suggest a limited local broadcast in the target area. Thus, our routing algorithm checks whether the message is within a certain range of the target position (line 1 of route function). If this holds, the vehicle broadcasts the message. Note that this happens only once per vehicle in this area, because repeatedly received messages are discarded.

Finally, message transmissions in VANETs may fail due to collisions, obstacles or intended receivers having moved out of radio range. To ensure reliable communication in position-based routing, nodes apply a hop-by-hop acknowledgment scheme for all non-broadcast messages (cf. Section 3.1). If the sender does not receive an acknowledgment, it removes the unreachable neighbor from its neighbor table, selects a new neighbor and resends the message. It has to be noted that this approach may lead to duplicates of messages in the network: if the sender of a message does not receive an acknowledgment (e.g. due to a collision), it misleadingly sends a duplicate of the message to another neighbor. This may slightly increase network load.

In anycast routing situations, where the message target is a base station, we allow that this target may be altered (line 5 of route function). This improves routing in situations as depicted in Figure 5.2: The vehicle registering its information first sets base station BS1 as target of the register message. The message is currently at a local minimum distance d_{min} to BS1. Therefore, the vehicle caches the message until it approaches BS2 which becomes the new target of the message.

5.5 Evaluation

In this section the performance of the proposed routing scheme is evaluated. The underlying evaluation methodology is explained in Section 5.5.1. Results follow in Section 5.5.2.

5.5.1 Methodology

With help of simulation, the absolute performance of the proposed routing algorithm can easily be determined. Nevertheless, in a sparse VANET, message delivery in some cases may be totally impossible due to lack of communication partners, and thus the absolute performance of the algorithm is not a sufficient measure to rate its quality.

Therefore, in addition the relative performance of the algorithm is determined in comparison to that of the message flooding scheme (Algorithm 4.1), since the scheme approximates a routing algorithm with optimal delivery rate and latency (cf. Section 4.2.2). To allow for a direct comparison of the two algorithms in the same network topology, exactly the same vehicular movement patterns and message generation times were set for each algorithm. This allows to find possible paths to the target node, and to identify alternative routing strategies.

We set the same beacon interval in both algorithms: each node sent beacons randomly and uniformly distributed over the interval [0.375 s, 1.125 s]. The local broadcast radius was set to the radio range of 225 m. We evaluated the algorithms at 10 % and 20 % penetration rate. We conducted three simulation runs per penetration rate.

For each routing situation, about 1000 samples were collected per simulation run. To evaluate the "vehicle-to-base station" routing situation, we selected a vehicle at random that generated a message for a base station (anycast). To evaluate the "base station-to-vehicle" routing situation, we randomly selected one of the base stations to send a message to a vehicle randomly chosen among those that have the selected base station as nearest one. In the "vehicle-to-vehicle" routing situation, we selected a random (sender vehicle, target vehicle)-pair with a common nearest base station.

5.5.2 Results

We now compare the routing algorithms in the three evaluated routing situations. The evaluation includes the success rate (Section 5.5.2.1) and the delivery latency (Section 5.5.2.2).

5.5.2.1 Success rate

Figure 5.3 shows the success rates in the evaluated routing situations, averaged over all runs. A main finding here is that, although the proposed algorithm is fairly simple and

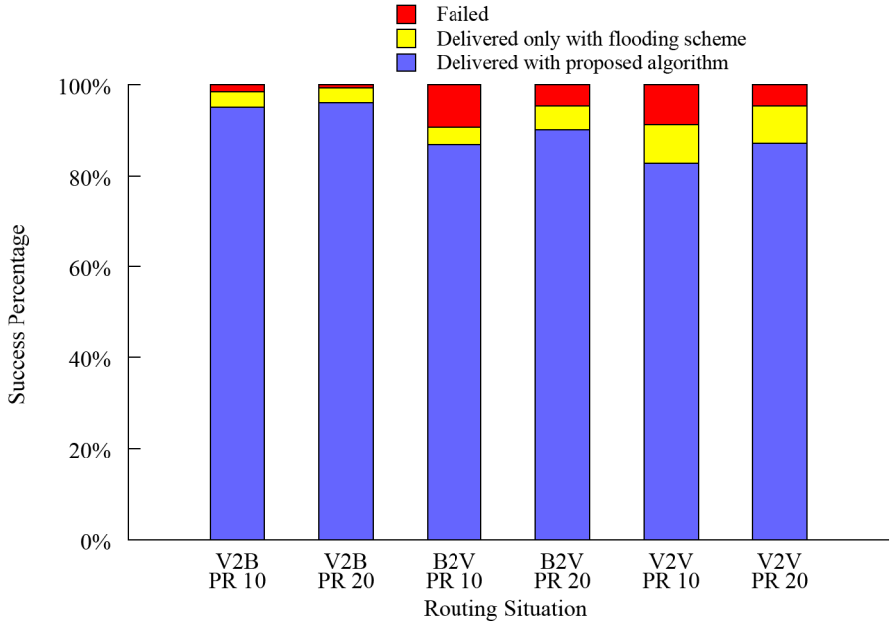


Figure 5.3: Routing performance in "vehicle-to-base station" (V2B), "base station-to-vehicle" (B2V), and "vehicle-to-vehicle" (V2V) situations. The proposed algorithm comes close to the flooding scheme. Success rate increases with penetration rate (PR) thanks to the more dense communication network. V2B has the highest success rate because of anycast.

relies on a small knowledge base, its performance is already close to that of the flooding scheme. The fairly high (relative and absolute) performance can be attributed to two issues. First, the algorithm includes measures combating the specific challenges of the routing situations. Second, with help of the network of base stations, the communication distances occurring between originator and target node are rather low; consequently, the routing algorithms are not faced with very complex network topologies. Examples of those cases where message delivery failed with the proposed algorithm but succeeded with the flooding scheme are studied below.

It can be seen that sometimes, even the flooding scheme fails to deliver a message. There are several possible reasons for this. One reason is that the target of the message has departed. Another reason especially at low penetration rate is that the network is sparse and there may not exist a route between sender and target of a message. Especially at 10 % penetration rate, it is difficult to reach a vehicle. The VANET becomes more dense at 20 % penetration rate, causing an increased delivery rate.

5.5.2.1.1 Vehicle-to-base station In this case, there is no specific target for the message but one out of many possible targets has to be reached. The success rate is the highest among the three investigated situations. The main reason is that the target node is at a fixed location and never leaves the VANET. Furthermore, there are multiple target nodes available, allowing for anycast. This leads to high success rates even at 10 % penetration rate.

Some messages are delivered by the flooding scheme but not by the routing algorithm. The reason is that one of the forwarders selected by the routing algorithm either departs from the scenario or moves into a direction where it does not encounter other vehicles for a longer period of time.

5.5.2.1.2 Base station-to-vehicle In this situation, delivery rates are lower than in the vehicle-to-base station situation, because the message has to arrive at a specific target.

It follows an analysis of samples of messages that the proposed routing algorithm cannot deliver, but the flooding scheme can. Many cases are similar to the one depicted in Figure 5.4. Here, the base station closest to the target forwards the message to a vehicle which does not meet a suitable receiver of the message. With flooding, the message is first forwarded by another base station from which forwarding to the target vehicle is possible. We encountered other situations similar to that depicted in Figure 5.5. Here,

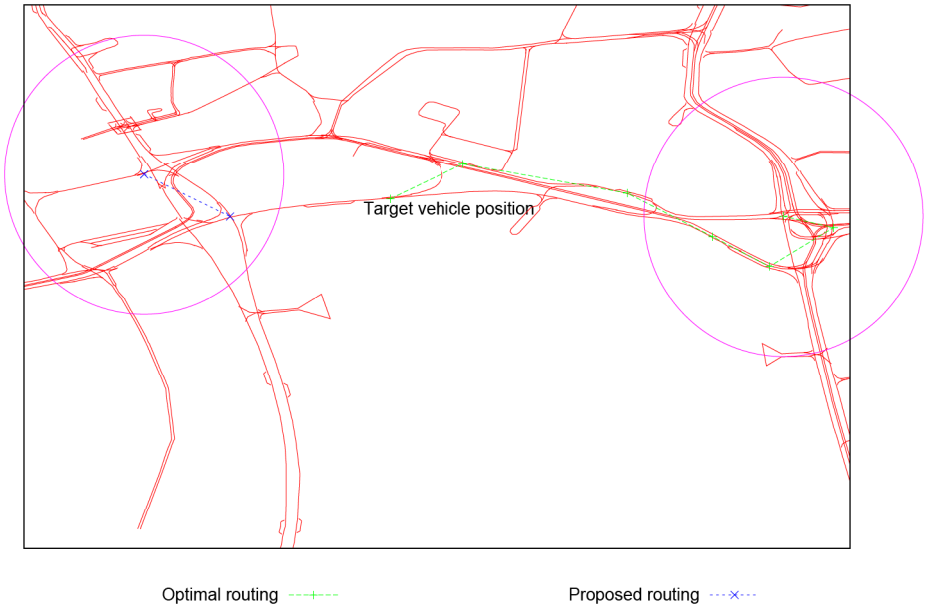


Figure 5.4: Delivery possible with distant base station as starting point

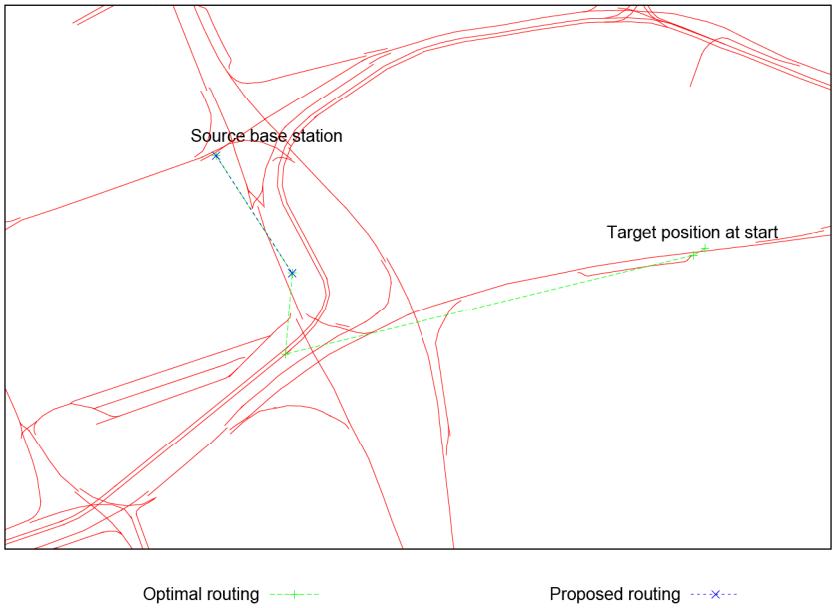


Figure 5.5: Optimal route discovered by flooding vs. route used by proposed algorithm

the flooding scheme reaches the target vehicle with help of a vehicle on the same cross street as the target vehicle, thereby increasing the remaining distance. Because of this increase in distance, the greedy routing algorithm does not consider this vehicle.

When this routing situation occurs during information search, its success rate can be expected to be higher than in the evaluated situation. The reason is that in search, an information source or an originator addressed by a base station has just recently successfully communicated with a base station (either sending the request or the registration). This is different from the vehicle-to-base station situation, where the message investigated may origin from a node which is not able to communicate with a base station.

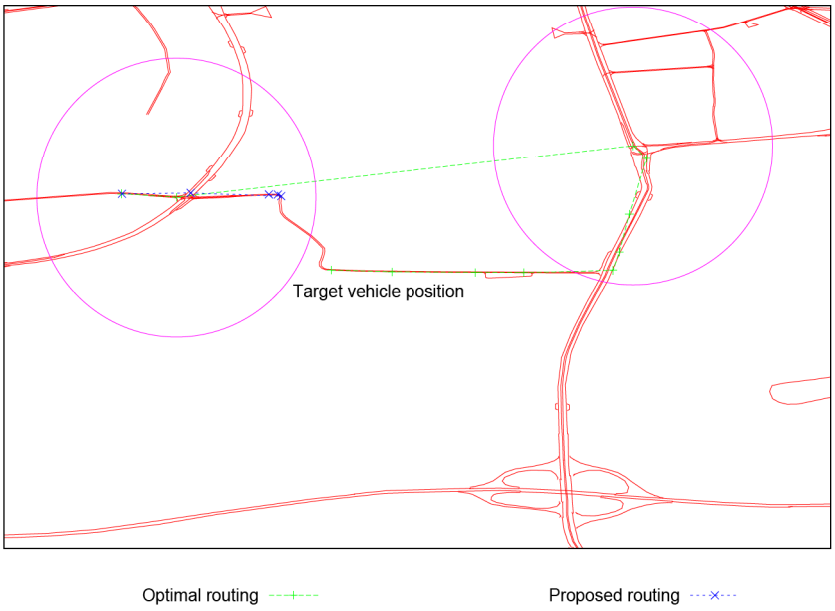


Figure 5.6: Example of failed vehicle-to-vehicle delivery: the delivery fails along the shortest path, but is possible on a geographically longer path.

5.5.2.1.3 Vehicle-to-vehicle The vehicle-to-vehicle routing situation has the lowest success rate among the three. The delivery rate is lower than in the base station-to-vehicle routing situation, because there, the base station was in the center of the area of which the target nodes were chosen from. In contrast, for the evaluation of the vehicle-to-vehicle routing situation, two vehicles are randomly selected from the area where a certain base station is the closest one; thus, the expected communication distance is larger on average.

Figure 5.6 shows a representative example where the proposed algorithm fails but the flooding scheme finds a solution. The proposed algorithm selects the shortest route which leads into a dead end because of missing communication partners. Instead, routing along a geographically longer path succeeds to reach the target vehicle, because that route contains many vehicles capable of forwarding the message. Of course, implementing an algorithm so that each vehicle is able to compute such a solution requires knowledge of current and future vehicle locations, which is infeasible in practice. Nevertheless, obtaining a rough knowledge about traffic density on nearby routes is feasible and could serve as a guideline.

Besides this type of failure, we observed cases where the sparseness of the VANET causes some messages to be underway for time periods of more than 30 s. In such cases, the target vehicle has moved too far away from the position used as target position, making it impossible for our algorithm to deliver the message. The optimal algorithm reached the vehicle at its distant position. To reproduce these solutions in practice, a very accurate position prediction is required, e.g. based on an intended route stored in the navigation system of the target vehicle.

5.5.2.2 Delivery latency

In this section the latency of the message deliveries is analyzed. Again, the different routing situations are distinguished. We observed qualitatively identical behavior in all simulation runs, and here only show the results of one run for each routing situation.

Figure 5.7 shows the EDF of delivery latency for the vehicle-to-base station routing situation. At this microscopic level (logarithmic x-axis), the curves exhibit several properties that are characteristic for all routing situations. The rightmost value of each curve is equivalent to the success rate of deliveries. This section continues with a comparison of the proposed algorithm and flooding at the same penetration rate. Afterwards, the behavior at increased penetration rate is analyzed.

In the time period between 0.0001 s and 0.1 s, the proposed algorithm has a much higher delivery rate than the flooding algorithm. This is because the jittered rebroadcasts and message requests of the flooding algorithm ($w_1, w_2 \in [0\,\text{s}; 0.1\,\text{s}]$) cause a small hop-by-hop delivery latency; the proposed algorithm has no such delay because only one node at a time is to forward the message, consequently no jitter is required for collision avoidance. We believe the influence of the short delay of flooding on its success rate is negligible because the possibility that a reachable node is unreachable after this short amount

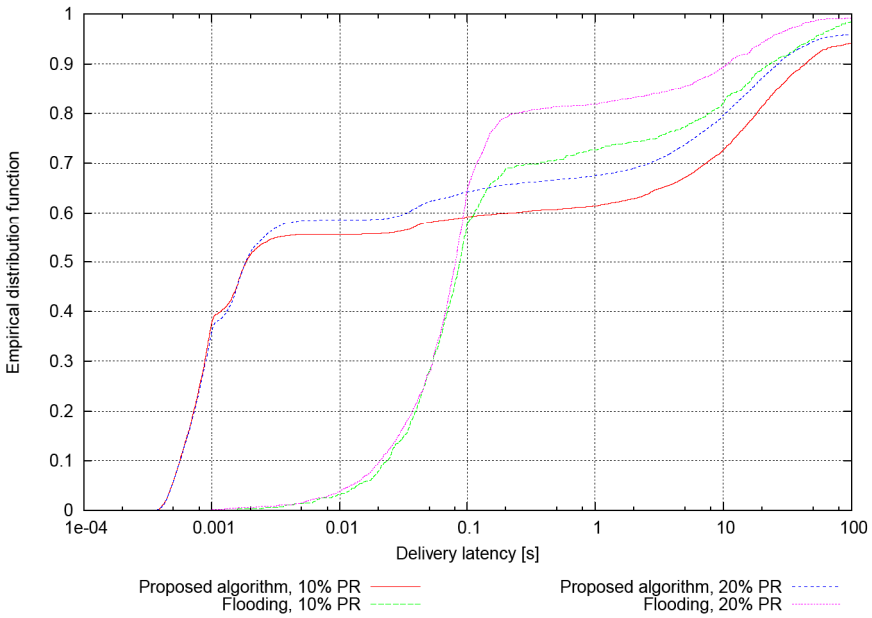


Figure 5.7: EDF: delivery latency in vehicle-to-base station routing

of time is almost zero. After 1 s, the flooding algorithm can be expected to reach all reachable nodes up to a distance of 10 hops away from the originator. At this point in time, the flooding algorithm already has a higher delivery rate because it always finds the fastest path to the target. After 10 s, both algorithms have delivered more than 75 % of all messages.

Considering the behavior at 10 % penetration rate, from the delivery rate of flooding at 1 s it can be concluded that an (up to about 10 hop-) multi-hop path to the target exists in about 75 % of all cases. The proposed routing algorithm at 1 s has a delivery rate of about 65 %, so it mostly finds a fast path. At 20 % penetration rate, the delivery percentage of flooding at 1 s is about 83 %, and that of the routing algorithm is roughly 70 %. This only minor increase in spite of the doubled penetration rate further underlines the importance of the base stations for the communication in the scenario.

In the base station-to-vehicle situation (Figure 5.8), the curve shapes are similar to those in the vehicle-to-base station case. A difference is that there are less multi-hop paths to the target available (about 70 % at 20 % penetration rate, considering the delivery rate of flooding at the 1 s mark). This is because the message is not targeted to one out of many nodes (anycast) as in the previous case, but only to one specific node (unicast).

Figure 5.9 depicts the EDF of delivery latency in the vehicle-to-vehicle routing situation.

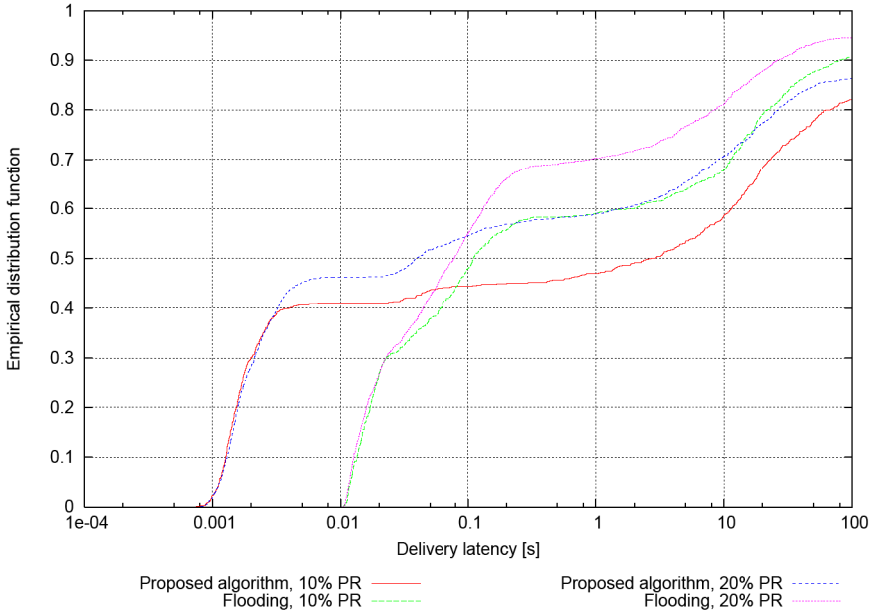


Figure 5.8: EDF: delivery latency in base station-to-vehicle routing

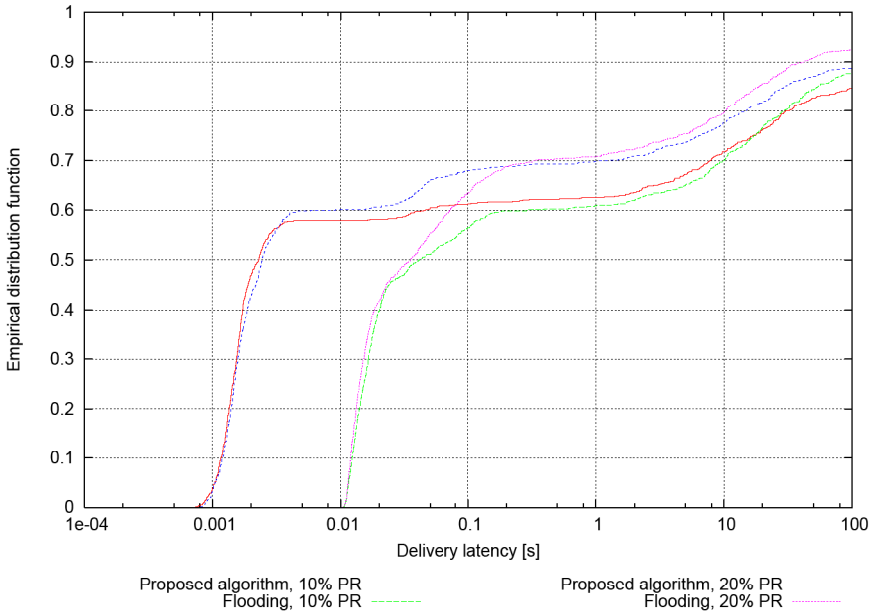


Figure 5.9: EDF: delivery latency in vehicle-to-vehicle routing

At 10 % penetration rate, the delivery rate of flooding exceeds that of the proposed algorithm only after more than 10 s. Flooding performs better in delivering messages heavily influenced by delays, because—unlike the proposed routing algorithm—it does not depend on the accuracy of the target position information, and is thus able to reach moving vehicles after a longer period of time.

5.6 Conclusions

In the evaluated scenario, the delivery rate of our proposed routing algorithm is within 5 %–10 % of the optimum, depending on the routing situation. Our evaluations show that the delivery latency of the proposed algorithm is close to the respectively achievable minimum. The good performance of the proposed algorithm can be attributed to two issues. First, the algorithm includes mechanisms that combat the specific challenges of routing in the specific routing situations. Second, the network of base stations reduces the communication distances within the wireless domain, and thus the complexity of the occurring network topologies is rather low.

It is important to note that the routing situations were evaluated independently. In contrast, in search, base station-to-vehicle routing is used only for vehicles that have just recently communicated with a base station (either for reaching an information source that has recently registered, or for reaching the originator who recently sent a search request to a base station). The same applies for vehicle-to-vehicle routing. Thus, slightly higher delivery rates can expected in search.

In future work, the routing algorithm may be improved in several ways. To increase the chance of predicting routes evolving in near future, vehicles may be provided with knowledge of road topology and traffic density [WER+03, NDLI04, LSW+08], as this gives vehicles an indication of the density of the communication network along alternate paths. Besides that, replication may increase the delivery rate at the cost of an increased network load. For example, vehicles may send search requests to multiple base stations; if an information source is found in the search index, a content request may be issued by multiple base stations.

5.7 Chapter Summary

This chapter has defined and addressed the problem of unicast and anycast routing in infrastructure-supported VANETs. Different routing situations occuring in such networks have been classified and their specific challenges have been identified. Thereon, a routing algorithm based on the idea of greedily minimizing the beeline distance to the target has been presented. It employs local caching to overcome local minima, limited local broadcast to reach moving vehicles, and message acknowledgments to increase reliability in communication. In anycast routing, the algorithm allows for an opportunistic distance-based change of the anycast target.

To determine the quality of the proposed algorithm, the algorithm has been compared to the message flooding scheme (Algorithm 4.1) by means of simulation. Both algorithms perform comparably in the anycast routing situation of vehicle-to-base station routing. Flooding has a slightly higher success rate in unicast routing, mostly due to the routing algorithm sometimes forwarding messages to nodes that never meet a suitable next hop. In such cases, the flooding scheme often finds possible routes on geographically longer paths. Most of these routes can only be found when vehicles have precise knowledge about future vehicle movements, which seems unlikely in practice.

Chapter 6

Index-based Search vs. Flooding-based Search—Implementation and Evaluation

In this chapter, an index-based search method for infrastructure-supported sparse VANETs is defined. Its success rate, latency, and network load are compared to that of a second search method which is an extension of the near-optimal flooding scheme (Algorithm 4.1). In Chapter 4, we have shown that the success rate of search based on the flooding scheme is an upper bound, as long it is not impacted by delays due to high network load. A main goal of this chapter is to determine how close the implemented schemes come to the theoretical bound.

The search methods are evaluated under conditions where the number of information items and queries in the VANET is low, so that the network load of registrations and queries does not lead to congestion on the wireless channel. Efficient registration is not addressed in this chapter; this is the topic of Chapter 7.

After reviewing related work (Section 6.1), this chapter continues with the definition of an index-based and a flooding-based search scheme (Section 6.2). Subsequently, details concerning the implementation of the search schemes are presented (Section 6.3), whereafter the schemes are evaluated in the sample scenario (Section 6.4). Conclusions from evaluation results are drawn in Section 6.5. The chapter is summarized in Section 6.6.

6.1 Related Work

This section focuses on related work on bandwidth-efficient flooding. Other search methods and search index types have been surveyed in Section 4.1.

One bandwidth-efficient flooding scheme, called Urban Multi-Hop Broadcast (UMB) Protocol has been proposed by Korkmaz et al. [KEOUO04]. In their scheme, only the most distant forwarder is selected to rebroadcast a message. This forwarder is determined through an RTS/CTS-like scheme for broadcasting, where nodes jam the channel by sending a black-burst signal after overhearing an RTS message. The length of this signal is proportional to the nodes' distance to the RTS sender. Immediately after sending the signal, each node overhears whether the channel is free, which is the case only for the most distant node from the RTS sender. That node sends a CTS message, thereby indicating it is responsible for the next forwarding step. The RTS sender then broadcasts the message to all nodes, whereafter the former CTS sender again starts the forwarding process with an RTS. This process only reaches all nodes along a single street. At intersections, a base station (assumed to be located on the intersection) initiates multiple of these processes, one for each outgoing road. The evaluations in the paper show that the bandwidth saving comes at the cost of an increased delivery latency. The authors have further extended their work for fully ad-hoc vehicular networks (Ad-Hoc Multi-Hop Broadcast, AMB), the key idea being that at an intersection the vehicles themselves determine the best repeater [KEO06]. Similar to UMB and AMB, Chiasserini et al. also propose a distance-based rebroadcast prioritization [CGG+06].

Alshaer and Horlait suggest that broadcast receivers only probabilistically rebroadcast a message [AH05]. The likeliness of a rebroadcast is inversely proportional to the network density, which nodes determine by the number of their one- and two-hop neighbors. Additionally, a rebroadcast delay is introduced, its duration computed based on a combination of the receiver's distance from the last sender and the network density.

Wisitpongphan et al. also define an efficient flooding scheme [WTP+06]. Three alternative rebroadcast rules for a received message are proposed. With the first rule, nodes immediately rebroadcast a message with a probability proportional to their distance from the last sender. The second rule advises all vehicles to backoff for a random time before rebroadcasting a message. When nodes overhear another node broadcasting the same message, they suppress their own rebroadcast. This suppression strategy is similar to the suppression scheme presented in this chapter for the case that multiple nodes are about to request the same advertised message ID. The third rule is a combination of rules one and two.

In the controlled flooding scheme by Harras et al., several parameters are introduced to limit the network load of flooding [HABR05]. Among these, a message lifetime limits the total age of messages. A times-to-send counter indicates how often a node may

send a message in total. A retransmission wait time delays the advertisement (and thus forwarding) of a message after it has just been forwarded. Like in the scheme of Harras et al., nodes in our bandwidth-efficient flooding scheme advertise message IDs in their beacons and use these and other parameters to reduce network load.

The original paper on the broadcast storm problem in MANETs also contains five bandwidth-reduction schemes [TNCS02]. The probabilistic scheme is the same as the first rule of [WTP+06]. With the counter-based scheme, a node defers message retransmission for a random period of time. If the node overhears c rebroadcasts of the message within this time, it discards its own scheduled rebroadcast. In the distance-based scheme, a node also delays the rebroadcast for a random period of time and listens for rebroadcasts of the same message; if it overhears a rebroadcast by another node closer than a threshold distance d, it cancels its own rebroadcast. The location-based scheme is the same as the distance-based scheme, except that nodes—instead of the distance—calculate whether their own broadcast will cover a minimum additional area. The last proposed scheme is a cluster-based scheme where a hierarchy in terms of clusters is formed in the network and only the clusterheads rebroadcast the message. Our bandwidth-efficient flooding scheme includes the counter-based scheme for suppression of broadcast instructions and the distance-based scheme for suppression of content requests and message advertisements.

If the VANET was fully connected and static, a flooding scheme could be determined that reaches all nodes with minimum latency at minimum bandwidth cost [ZPM04]: one could calculate the minimum connected dominating set (MCDS), defined as the subset of nodes with minimum cardinality that each network node is connected to. Only MCDS members may rebroadcast a message.

6.2 Search Methods

Let us briefly review the results of Chapter 4: neglecting congestion on the wireless channel, flooding is the fastest and most successful search method because the required communication distance is the lowest possible. Unfortunately, flooding is inefficient regarding network load; index-based search does not suffer from this problem—its network load is low. On the other hand, existing index-based schemes involve a third vehicle for checking the index and issuing a content request, therefore their communication distances and failure rates are higher than that of flooding. An ideal VANET search

method should unify the advantages of both approaches: low network load and communication distance. Conceptuation, implementation, and evaluation of such a scheme is our goal in this section.

The study conducted in Chapter 4 showed that search mostly fails in sparse VANETs without infrastructure-support because messages usually can not be delivered fast enough. Therefore, base stations are required to support search. The search approach presented in this section is based on the apparent idea of leveraging the base stations to store the search index. The rationale behind this is that the "detour" taken for a request to query the search index at a base station is statistically small if base stations are involved in most communication anyway. Furthermore, base stations as immobile, always-online VANET nodes are ideally suited to store the search index. Performance of such an index is therefore likely to represent the "best case" of structured search.

Alternatively, if hosting the index at the base stations is no option, the search index may also be stored in vehicles near the base stations; we expect this approach to perform comparably, as long as vehicles are available at these locations. Such a vehicle-hosted index requires additional efforts to take into account the mobility of the vehicular hosts, e.g. through local broadcast of the available index information. We here focus on the study of the "best case", storing the index at base stations (Section 6.2.1). To have a near-optimal reference search method, we extend the flooding scheme, Algorithm 4.1 (Section 6.2.2).

In both search schemes, information sources that have received a request return the reply via unicast (according to the algorithm introduced in Section 5.4).

6.2.1 Index-based Search

In our studies, we focus on the wireless domain and idealize the base stations and their backbone network. Consequently, we do not impose a certain distribution of the index among the base stations; it may for example be distributed in a DHT-like way, or completely replicated at each base station. In the following, we explain our implemented concept where vehicles register in a DHT that is distributed among base stations.

To register their information items, nodes register the respective hash keys (see Section 4.1.3) at the nearest base station. This is repeated periodically with interval $t_{repeatRegister}$, since the position information stored in the register message naturally becomes inaccurate over time because of node movement.

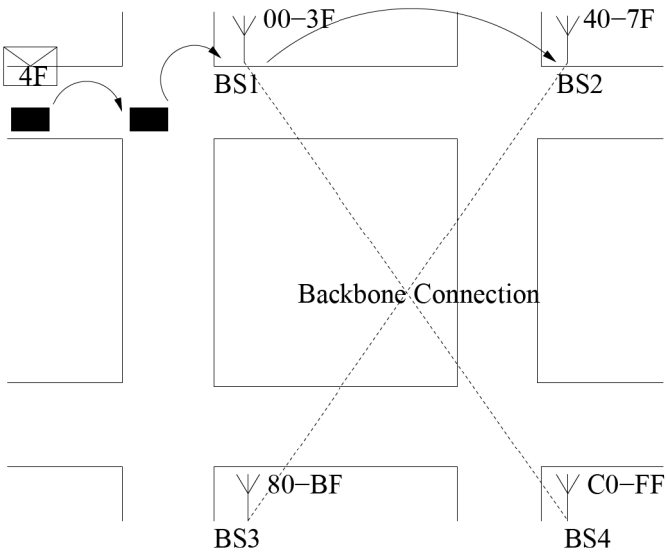


Figure 6.1: Vehicle sending registration for information item I with $h(D_I) = 4F$

Instead of registering in a fixed time interval, it is also conceivable that vehicles register after having moved a certain distance away from the origin of their last registration (fixed-distance interval). The rationale behind this alternative is that repetitive registration is not required as long as a vehicle is reachable at or very close to its last reported position. The advantage of this is that vehicles not moving for a longer period of time (e.g. because they have to wait at an intersection) do not generate additional network load. The drawback of this alternative is that based on the index, there no clear distinction between vehicles (and thus information items) that have left the scenario and vehicles that remain at a fixed position for a longer period of time. When the index is queried, the request might be redirected to an information source that has already departed. While this may also occur if registration is based on a fixed time interval, we believe that the distinction is easier when a fixed time interval is used.

Once a registration message has arrived at a base station, it is forwarded through the backbone network to the base station responsible for the included hash value, where it is finally stored for lookups. The search index is a soft-state index: vehicles do not actively unregister. To maintain the index, base stations delete all registrations older than $t_{RegisterAge_{max}}$, because chances are high that the information source which sent the registration has departed or moved far away and therefore contacting it at its reported position is likely to fail. Figure 6.1 illustrates information registration in an example where the search index of hash space 00-FF is distributed among four base stations.

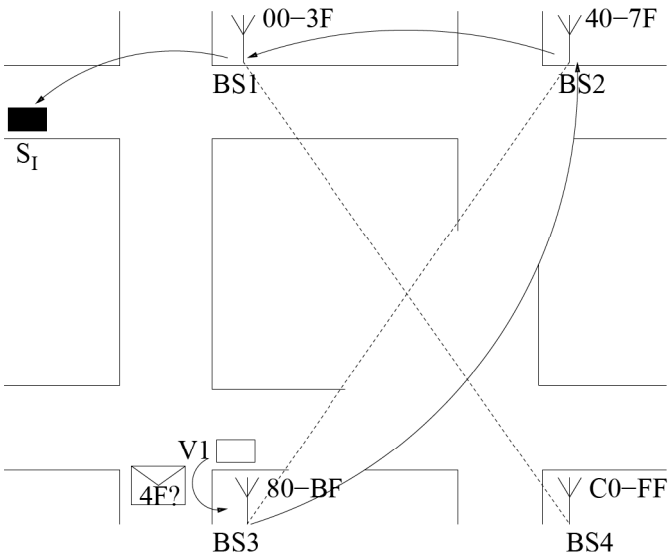


Figure 6.2: Vehicle searching for information item I with $h(D_I) = 4F$

A vehicle in need of information sends a search request to the nearest base station. The request contains the originator's ID, position, and the hash value of the information description of interest ($h(D_I)$). When the search request has arrived at the base station, it is forwarded through the backbone network to the base station responsible for the corresponding part of the search index. For example, in Figure 6.2, vehicle V1 requests information I with $h(D_I) = 4F$. This request arrives at base station BS3. From there, the request is forwarded to BS2 since BS2 is responsible for that part of the search index.

If the responsible base station has previously received a registration of an information source S_I, it sends a content request through the backbone network to the base station that is closest to S_I. Continuing the example, in Figure 6.2, BS2 forwards the content request to base station BS1. From there, the content request is sent to the information source through the VANET. When the content request arrives at the information source, the information source returns a reply message to the position of the originator, which is stored in the content request.

In case multiple information sources are registered for $h(D_I)$, the base station responsible for $h(D_I)$ forwards content requests to each information source. It is also conceivable that one may prefer to forward the message to only one information source, instead. This is a trade-off between robustness and network load. We decided to contact all information sources because this allows for a direct comparison between index-based

and flooding-based search in case multiple information sources exist.

If there is no registration of a requested information item in the index, the responsible base station creates a negative reply and forwards it to the base station closest to the originator. Starting from there, the reply is forwarded through the VANET to the originator.

6.2.2 Flooding-based Search

In this section, an extended version of the flooding scheme (Algorithm 4.1) is presented. If parameterized correctly, the network load of the original variant is reduced, while the levels of success rate and latency are maintained. The key idea behind flooding-based search is to forward a content request to every vehicle in the VANET as fast as possible, since each node could possibly be an information source. No expanding-ring-search—e.g. as done by AODV [PR99]—is performed, because that would further delay delivery of a request to distant nodes. Consequently, the presented scheme is designed to reach all information sources whenever possible, in the shortest amount of time.

To flood content requests in the VANET, we propose Algorithm 6.1. Compared with Algorithm 4.1, the new version has four extensions: limited message lifetime, limited number of transmissions per message per node, a retransmission delay, and message suppression strategies. The extensions are now outlined in more detail.

Message lifetime is infinite in Algorithm 4.1. Limiting this is crucial for the network load of the VANET. Naturally, the best time to stop flooding a request is when it has arrived at an information source. Nevertheless, in flooding-based search, VANET nodes have no indication whether a flooded request has arrived or has failed due to no information source being available. Thus the message lifetime has to be bounded artificially. Typical ideas of doing this in networks are limiting its allowed number of hops or limiting its validity time l [HABR05]. Limiting the hop count is reasonable if a message is to be distributed within a limited geographical area. But since we aim to reach all nodes in the VANET with a flooded request, we decide for the validity time (cf. line 1 of timeout function).

After some time, most of the VANET nodes will own a copy of the flooded request. Thus, the advertisements of these message IDs in vehicles' periodic beacons will remain unanswered. In this situation, limiting the number of total transmissions per message ID per node by a number k reduces the beacon load [HABR05] (cf. line 5 of receive-BroadcastInstruction function).

receiveAdvertise($ID\ m.id, Node\ s$)

```
1: if (unknown(m.id)) then
2:    send(m.id, s, w_1)
3: else
4:    if (dist(s, me)< minDist) then
5:       suppressAdvertise[m.id] ← true
6:    end if
7: end if
```

receiveBroadcastInstruction($ID\ m.id, Node\ s$)

```
1: if (currentTime − lastSent[m] > d) then
2:    send(m, s)
3:    timesSent[m] ← timesSent[m] + 1
4:    lastSent[m] ← currentTime
5:    if (timesSent[m] > k) then
6:       discard(m)
7:    end if
8: end if
```

receiveMessage($Message\ m, Node\ s$)

```
1: if (unknown(m)) then
2:    timesSent[m] ← 0
3:    broadcast(m, w_2)
4: else
5:    if (currentlyDeferringMessage(m) ∧ dist(s, me)< minDist) then
6:       cancelBroadcast(m)
7:    end if
8: end if
```

timeout($Timer\ b$)

```
1: if (m.age < l) then
2:    if (currentTime − lastSent[m] > d ∧ ¬suppressAdvertise[m.id]) then
3:       advertise(m.id)
4:    end if
5:    suppressAdvertise[m.id] ← false
6: else
7:    discard(m)
8: end if
```

overhearBroadcastInstruction($ID\ m.id$)

```
1: if (currentlyDeferringBroadcastInstruction(m.id)) then
2:    discardBroadcastInstruction(m.id)
3: end if
```

Algorithm 6.1: Extended advertisement-based message flooding scheme

b: beacon interval
l: message lifetime
d: retransmission delay
k: maximum times to send per node per message
$minDist$: threshold for distance-based message suppression
unknown($Message\ m$): message received for the first time
send($Message\ m, Node\ t, Delay\ w$): defer for w, then send m to t
send($ID\ m.id, Node\ t, Delay\ w$): defer for w, then send broadcast instruction for m to t
broadcast($Message\ m, Delay\ w$): defer for w, then broadcast m
advertise($ID\ m.id$): append $m.id$ to periodic beacon message
overhearBroadcastInstruction($ID\ m.id$): node overhears broadcast instruction by another node
currentlyDeferringBroadcastInstruction($ID\ m.id$): node is currently deferring broadcast instruction for $m.id$
currentlyDeferringMessage($Message\ m$): node is currently deferring broadcast of m
suppressAdvertise($ID\ m.id$): node does not advertise $m.id$ in next beacon
cancelBroadcast($Message\ m$): node does not rebroadcast m

Algorithm 6.2: Functions and variables of extended advertisement-based message flooding scheme

Similarly, after a node has broadcast a message, all nodes in its vicinity own the message. An advertisement in the next beacon will most likely not find new receivers. Thus, we introduce a mandatory retransmission delay d for recently broadcast messages [HABR05] (cf. line 1 of receiveBroadcastInstruction function and line 2 of timeout function).

Furthermore, in the original scheme, three types of messages are sent after deferring for a random period of time, not taking into account that the same message might just recently have been sent by other nearby nodes. First, nodes send beacons in interval b, containing advertised message IDs. Second, nodes return broadcast instructions for unknown messages after deferring for w_1. Third, nodes rebroadcast newly received messages after deferring for w_2.

The extended scheme includes message suppression strategies for these three message types. The key idea is that—while deferring a message transmission—if a node overhears a neighbor sending the same message, it suppresses its own transmission. For deferred advertisements and rebroadcasts, the neighbor has to be closer than a certain threshold distance $minDist$ to trigger the message suppression (cf. line 4 of receiveAdvertise function and line 5 of receiveMessage function). This strategy is the distance-based scheme of [TNCS02]. For deferred broadcast instructions, any neighbor in radio range sending a broadcast instruction for the same message triggers the suppression (cf. line 1 of over-

hearBroadcastInstruction function). In terms of the schemes proposed in [TNCS02], this strategy can be interpreted as distance-based scheme with $minDist$ equivalent to the radio range, or as counter-based scheme with $c = 1$.

It follows that more broadcast instructions are suppressed than advertisements and rebroadcasts. The rationale behind this is that broadcast instructions have a different target scope than the other two. It is sufficient that only one node returns a broadcast instruction on an advertisement to trigger the sending of the advertised message. In contrast, message advertisements and the advertised messages themselves need to propagate through the whole network as fast as possible. For example, if a node on an intersection suppresses a rebroadcast because it overhears that of another node in one direction, this might prevent or delay the propagation of the suppressed message in another direction.

When a base station receives a request, it broadcasts the request in the backbone network. With their next beacon, all base stations start advertising the message.

6.3 Implementation

In this section we present two implementation details concerning the search schemes. First, we outline the formats of the transmitted application-layer messages (Section 6.3.1). Second, we detail the global knowledge instance that allows for an evaluation of the schemes under specific conditions (Section 6.3.2).

6.3.1 Message Formats

Table 6.1 shows the formats of the implemented messages. In flooding-based search, beacons include the advertisement list of requests which are locally available and may be forwarded if requested. Messages of type BroadcastInstruction occur only in flooding-based search. Register messages occur only in index-based search.

Each information item of a node is mapped to a key that is a 64-bit hash value. Register, Request, and Reply messages contain a list of these keys. Note that there is no real content in Reply messages; these messages are transmitted only to check whether any content can be transferred at all.

Message type	Message field	Size [bytes]
All messages		
	Timestamp	8
	OriginatorID	4
	OriginatorPosition	8
	SenderID	4
Beacon		
	Flooding: List of Message IDs	$n \cdot 16$
BroadcastInstruction (Flooding only)		
	TargetID	4
	List of Message IDs	$n \cdot 16$
Register (Index only), Request, Reply		
	MessageID	16
	TargetID	4
	TargetPosition	8
	NextHopID	4
	MinDist	8
	Keys	$n \cdot 8$

Table 6.1: Message formats

6.3.2 Global Knowledge

To ease the evaluation of the search methods, we implement a global knowledge that provides two functionalities: a real-time list of keys in the VANET and a real-time position awareness.

When we evaluate the search engine, we are primarily interested in the results of queries that can be answered, i.e. the cases where an information source for a requested key is present in the VANET. For this purpose, we implement an artificial global knowledge that contains a list of all currently present keys. This is not the search index itself, but a real-time list to which joining vehicles add their keys and from which departing vehicles remove their keys. Thus, the real-time list is hard-state and may include keys yet unregistered in the index. When a vehicle is scheduled to issue a new search request, it randomly selects a key from the real-time list.

Furthermore, we use the global knowledge to evaluate the effect of position prediction. The idea is that vehicles include their planned route for the order of the next tens of seconds in registrations and requests (this apparent idea has already been mentioned e.g.

by Zhao and Cao [ZC06]). The underlying information may originate from a navigation system or may be an educated guess based on historical data. The benefit of such a prediction is that it allows limiting the region in which the target vehicle can be expected, and thus leads to a smaller required local broadcast area of the message. We implement this prediction as follows: starting from their last time of reporting (i.e. position information in registration or request), the exact position of each vehicle is known for a given period of time (parameter value *PredictionPeriod*). After this period of time, the last predicted position is assumed as current location of the vehicle.

6.4 Evaluation

In this section, we evaluate the performance of index- and flooding-based search. We first outline the underlying methodology (Section 6.4.1) and then present the parameter studies (Section 6.4.2, Section 6.4.3).

6.4.1 Methodology

The key question we investigate in this section is how close the success rate of index-based search comes to that of flooding and how these two schemes relate in terms of network load. Furthermore, we are interested to compare the performance of the implemented versions to our theoretical results.

The performance of both search methods is impacted by the choice of several parameters. We first study the performance of each search method individually with several parameter sets, identifying parameters that give high success rate and low network load. Thereafter, we investigate the relative performance of the search schemes at different request rates and different degrees of information availability.

Both search methods are studied under the condition that each vehicle carries exactly one information item. Thus, the network load caused by content registration of vehicles in index-based search is practically negligible.

6.4.2 Parameter Study: Optimize Search Success Rate

Finding a parameter set that results in high search success rate and low network load is an optimization problem. The number of parameters and their domains make it

impractical to explore the whole parameter space. Heuristic approaches (e.g. genetic algorithms, simulated annealing) come to mind, but require a large number of simulation runs to converge. Instead, we took a more pragmatic approach, selecting candidate values for evaluation of each parameter. We optimized both search methods for the same scenario under a penetration rate of 10 %. We set a request interval of 200 s for each vehicle, so that the impact of congestion on search performance is low. We evaluated each parameter set in a simulation of 1000 s of vehicular traffic, using three different random seeds of the traffic simulator. In the following, each parameter and the implications of its variation are discussed. This is followed by presentation and examination of the simulation results.

6.4.2.1 Discussion of parameter selection

Method	Parameter	Default value	Studied values
Both			
	Position prediction	60 s	0 s, 30 s, 60 s
	Local broadcast radius	0 m	0 m, 300 m, 600 m
	Message lifetime l	250 s	
	Beacon interval b	$\in$ [0.375 s;1.125 s]	
Index			
	Registration interval ($RegInt$)	60 s	30 s, 60 s, 90 s
	Maximum registration age	$1.5 \cdot RegInt$	$RegInt$, $1.5 \cdot RegInt$, $2 \cdot RegInt$
Flooding			
	Retransmission delay d	10 s	5 s, 10 s, 20 s, 60 s
	Times-to-send k	8	8, 4, 2, 1
	Suppression distance $minDist$	100 m	50 m, 100 m, 225 m
	Message request delay w_1	$\in$ [0 s;0.1 s]	
	Rebroadcast delay w_2	$\in$ [0 s;0.1 s]	

Table 6.2: Search parameters, their default and candidate values

The parameters of both search methods are listed in Table 6.2. Except for the local broadcast radius, all parameters were evaluated independently, setting the default values for the other parameters. We evaluated each candidate value of the local broadcast radius in combination with each candidate value of the position prediction period, as these two are closely related.

6.4.2.1.1 Common Parameters The position prediction period describes how well the future movements of vehicles can be anticipated. It influences the delivery of messages by means of position-based routing. Note that the required accuracy of the prediction for routing purposes is very low in comparison with positioning accuracy requirements of safety applications: vehicles near the target vehicle should have the target in their neighbor list; it suffices if a message reaches any vehicle in radio range of the target vehicle. By default, we assume a prediction period of 60 s. We additionally study prediction periods of 30 s and 0 s. All three values are evaluated in combination with all candidate values of the local broadcast radius.

The local broadcast radius is a parameter of position-based routing, thus it influences the delivery of replies in both search methods and of requests in index-based search. If it is selected too small and the target vehicle has moved too far away from the last predicted position, the message will not be delivered. By default, we turn off local broadcasting, since there is a rather long position prediction period of 60 s. We furthermore study values of 300 m and 600 m.

Vehicles delete messages with expired lifetime from their queue. Choosing a too small lifetime value leads to the dropping of messages that could still reach their intended target; a large value increases the storage space requirements in vehicles, but most importantly leads to advertisements of old message IDs in flooding. We set a default value of 250 s, supported by our theoretical results of search latency.

The beacon interval determines the accuracy of vehicles' neighbor tables and the inter-partition forwarding delay of flooded requests. The selected interval is the same as in Section 4.2.2 and Section 5.5.

6.4.2.1.2 Parameters of Index-based Search The registration interval is the interval in which each vehicle registers its information items in the index. This parameter allows for a trade-off between index accuracy and network load: the lower the value, the more accurate the index (i.e. the key- and location data of available information items); the higher the parameter value, the lower is the network load of index maintenance. We here set a default value of 60 s, being equal to the position prediction period.

The maximum registration age is the time period after which a registration is removed from the index. Selecting a high value for this parameter leads to the index containing many registrations of vehicles that have left the scenario—content requests forwarded to these vehicles are never answered. Removing outdated registrations is also necessary if the size of the index is important. This is the case if storage space is a scarce resource,

or if the lookup accuracy of the index decreases with an increasing number of stored information items (see Chapter 7). For these reasons, in the parameter study we seek to minimize this value without negatively impacting search performance. Naturally, as long as a vehicle periodically and successfully registers an information item, the item should not be removed from the index; consequently, the lowest studied maximum registration age is equivalent to the registration interval.

6.4.2.1.3 Parameters of Flooding During the retransmission delay period d, a vehicle does neither advertise nor send a content request it recently transmitted. The reason for introduction of this parameter is that if the message has previously been broadcast, all current neighbors are likely to have it available. Thus, the sender may wait before advertising it in his beacon again, thereby reducing network load. The larger this wait time is selected, the more bandwidth can be saved, but the higher is the probability of another vehicle passing through the radio range without receiving an advertisement of the message. Nevertheless, even if this should happen, there is still a high possibility the vehicle will receive the message anyway because all other neighbors that have just received the message advertise it. We set a default value of 10 s and additionally study values of 5 s, 20 s, and 60 s.

The times-to-send parameter has a similar effect as the retransmission wait time. After a certain time, all vehicles within a certain area are likely to own a flooded message. Vehicles advertising the message within such an area consume network bandwidth but are unlikely to find interested receivers. For this reason, the times-to-send parameter limits the number of broadcasts (and therefore advertisements) of a message per vehicle. When a vehicle has broadcast a message a number of times given by the times-to-send parameter, it deletes the message from its sending queue but still remembers the message ID to not again request it from advertising vehicles.

Message suppression reduces the network load of flooding by having nodes cancel their own rebroadcasts and advertisements in case they overhear another node within distance $minDist$ broadcasting the same message. The value allows for a tradeoff between aggressive ($minDist$ of zero) and bandwidth-efficient ($minDist$ equal to radio range) message dissemination.

The message request and rebroadcast delays w_1, w_2 help to avoid collisions caused by simultaneous transmissions of nearby nodes. Each node selects a value randomly and uniformly distributed over the given interval. The obtained values are low so that there is no unnecessary delay in message dissemination. The intervals are the same as those

of the simulation study of information propagation speed (Section 4.2.2) and of the evaluation of the routing algorithm (Section 5.5).

6.4.2.2 Results

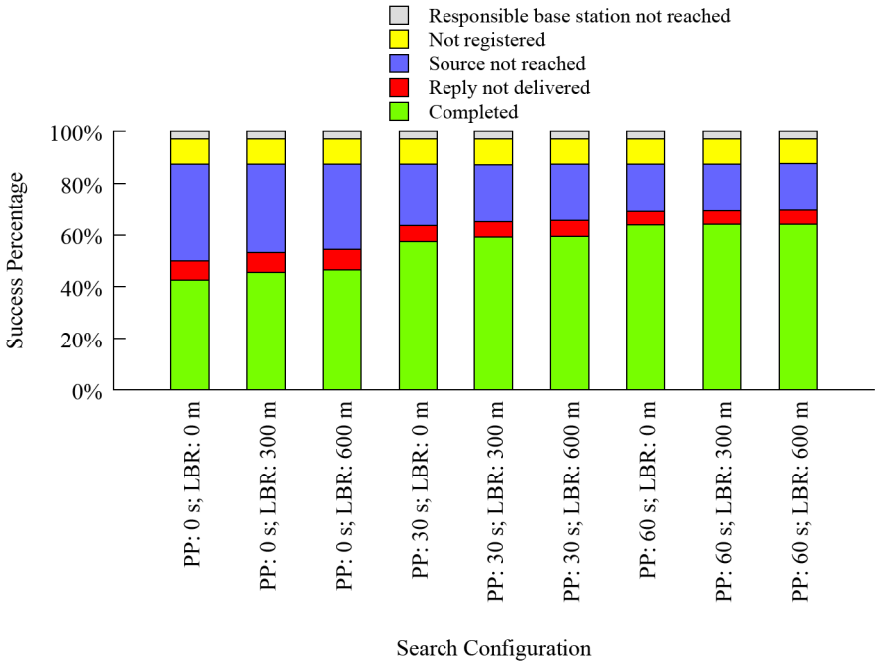


Figure 6.3: Success rate of index-based search depending on local broadcast radius (LBR) and position prediction period (PP), 10 % penetration rate

6.4.2.2.1 Local Broadcast Radius (Index) Figure 6.3 shows the success rate of index-based search depending on the investigated combinations of position prediction period and local broadcast radius. It is about 40 % without position prediction and increases to about 60 % with position prediction of 60 s. Those searches that fail do so mostly because requests redirected from base stations can not be delivered. This has three reasons. First, the respective information source may have departed. Second, the information source may be present, but located in an area where it is unreachable due to missing communication partners. Third, the information source may have moved away from its predicted position. The number of requests failing due to inaccurate position information is reduced when the prediction period is increased. Increasing the local broadcast radius only has a marginal effect on success rate. We suspected that since the VANET is sparse, there are no or only few vehicles that receive and rebroadcast the request in the target area. Thus, we investigated this effect at 20 % penetration rate (Figure 6.4), and found

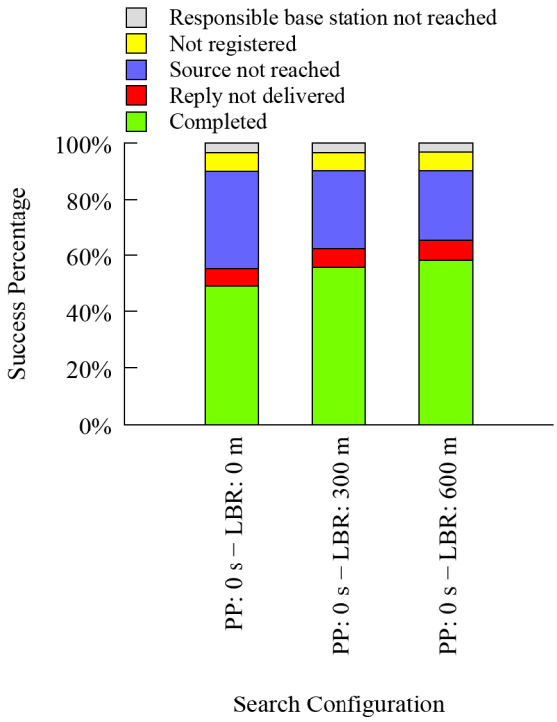


Figure 6.4: Success rate of index-based search depending on local broadcast radius (LBR), position prediction (PP) disabled, 20 % penetration rate

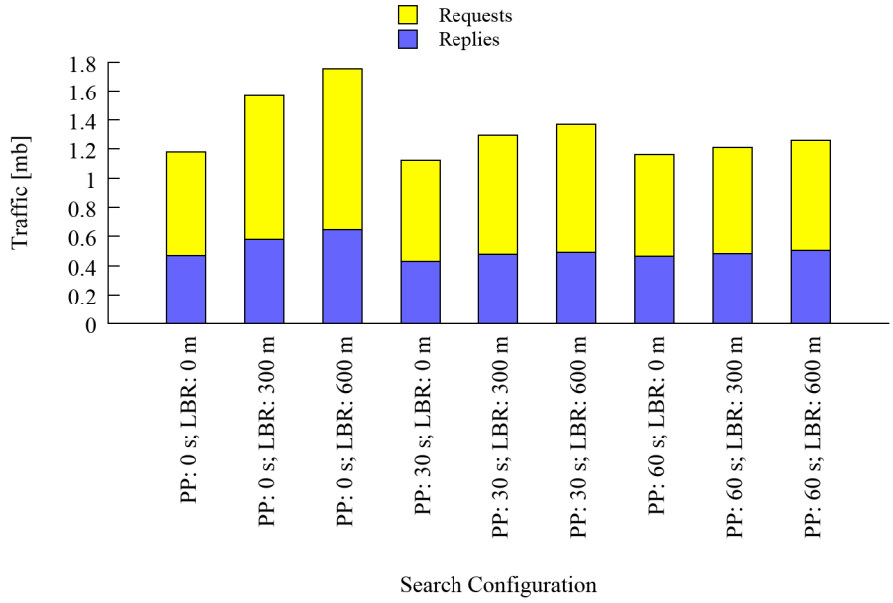


Figure 6.5: Network load of index-based search depending on local broadcast radius (LBR) and position prediction period (PP)

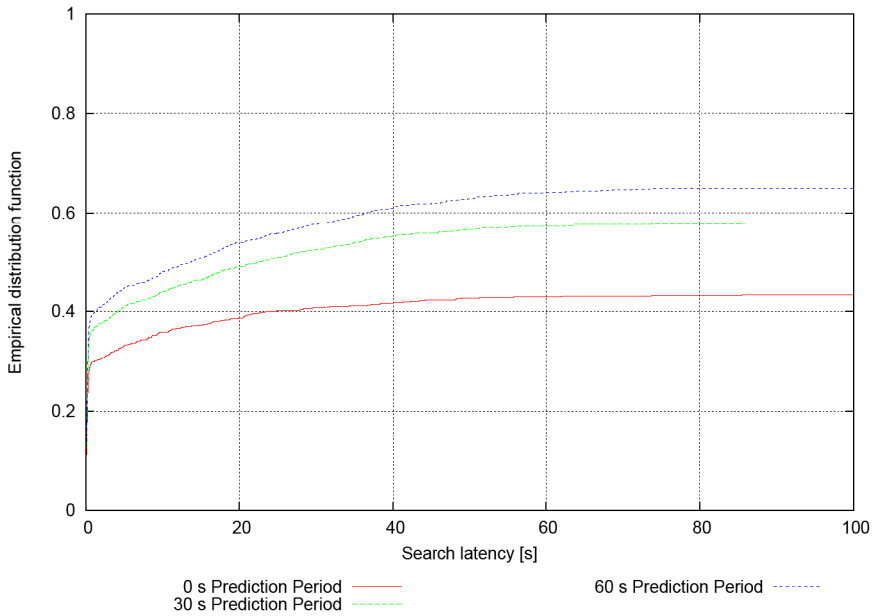


Figure 6.6: EDF: search latency of index-based search for 0 m local broadcast radius

that in this case an increase of the local broadcast radius leads to a slightly higher success rate. Still, the local broadcasting is not sufficient to compensate the lack of position accuracy.

The network load caused by search requests and replies is shown in Figure 6.5. As expected, network load increases with increasing local broadcast radius. Furthermore, there is always more traffic for requests than for replies. This is because of two reasons. First, only part of the requests are answered at all. Second, the position information used to route requests on average is older and thus less accurate than that used to route replies; more accurate position information avoids local broadcasting since messages are not locally broadcast if a vehicle carrying a message towards the target position discovers the target in its neighbor list. The second observation also explains why the highest network load occurs without position prediction. Although the success rate and consequently the number of replies is smaller without position prediction, most of the replies are locally broadcast and thus cause more network load than the replies at higher prediction periods.

Figure 6.6 shows the EDFs of search latency for the three investigated position prediction periods at 0 m local broadcast radius, based on one random seed. There was no significant difference in the results of other seeds and local broadcast radii. The properties of the EDFs are characteristic for all following EDFs of search latency. About two

thirds of successful searches are completed within the first second, indicating that there was an end-to-end multi-hop connection between originator, base station and information source available. The EDF then rises until about 60 s, after which no more searches are completed. The one third of searches finished after the first second are those relying on vehicular caching and carrying of the search message. The success rate of search can be determined from the right hand part of the EDF: the EDF reaches the success rate shown in Figure 6.3. With higher prediction periods, the more accurate position information increases the success rate of both the fast end-to-end multi-hop searches and the delayed searches.

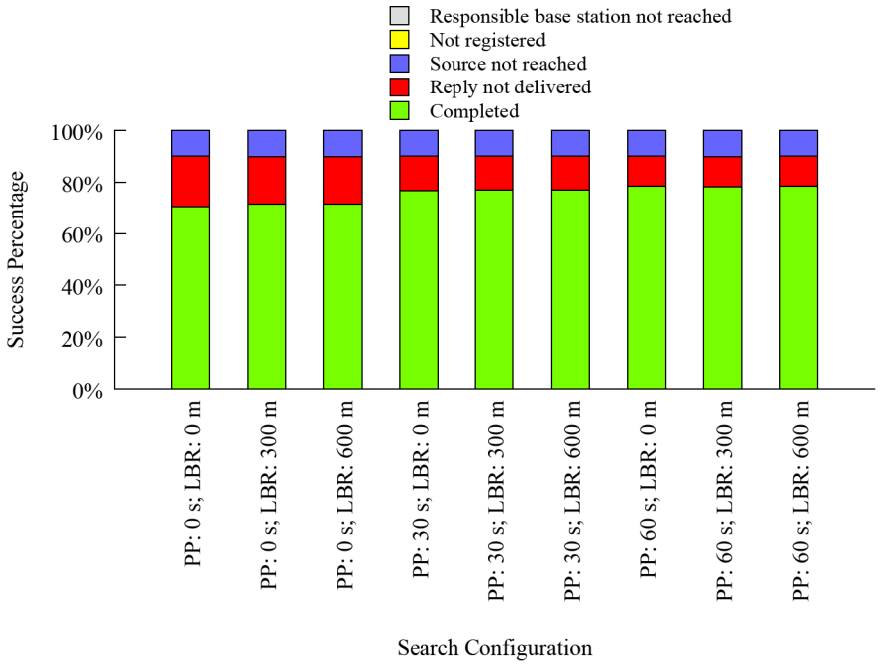


Figure 6.7: Success rate of flooding-based search depending on local broadcast radius (LBR) and position prediction period (PP), 10 % penetration rate

6.4.2.2.2 Local Broadcast Radius (Flooding) Figure 6.7 shows the success rate of flooding depending on position prediction period and local broadcast radius. Flooding has a high success rate even without any position prediction. The reason for this is that all requests are flooded, independently of prediction period and broadcast interval. The majority of searches fail because the reply cannot be returned, the reason being that these are returned via unicast, thereby relying on aging position information. Consequently, the success rate of the replies increases with increasing prediction period. Like

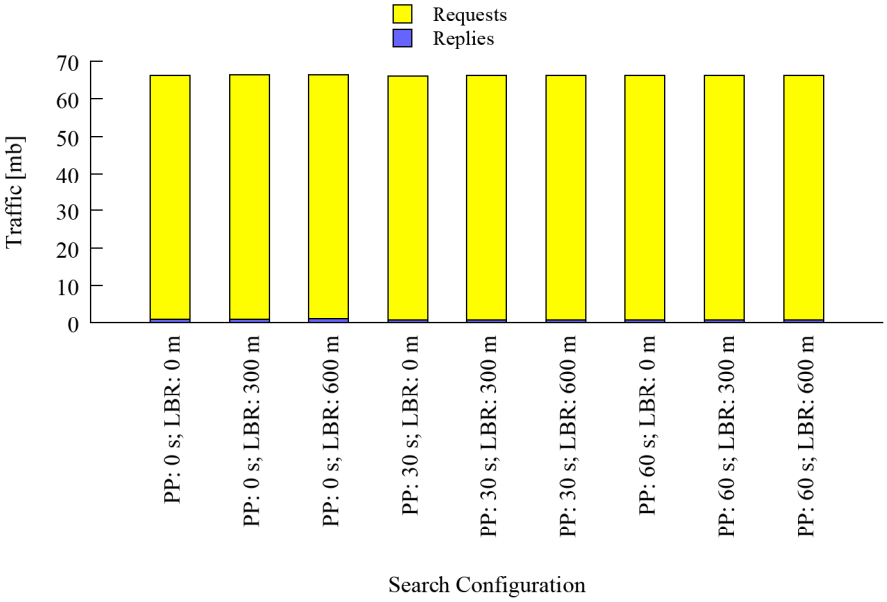


Figure 6.8: Network load of requests and replies in flooding-based search depending on local broadcast radius (LBR) and position prediction period (PP)

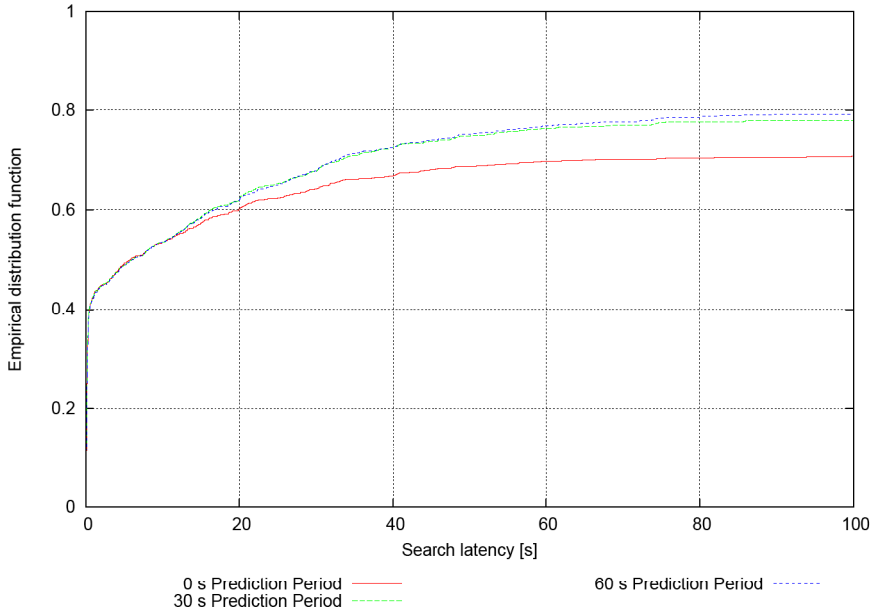
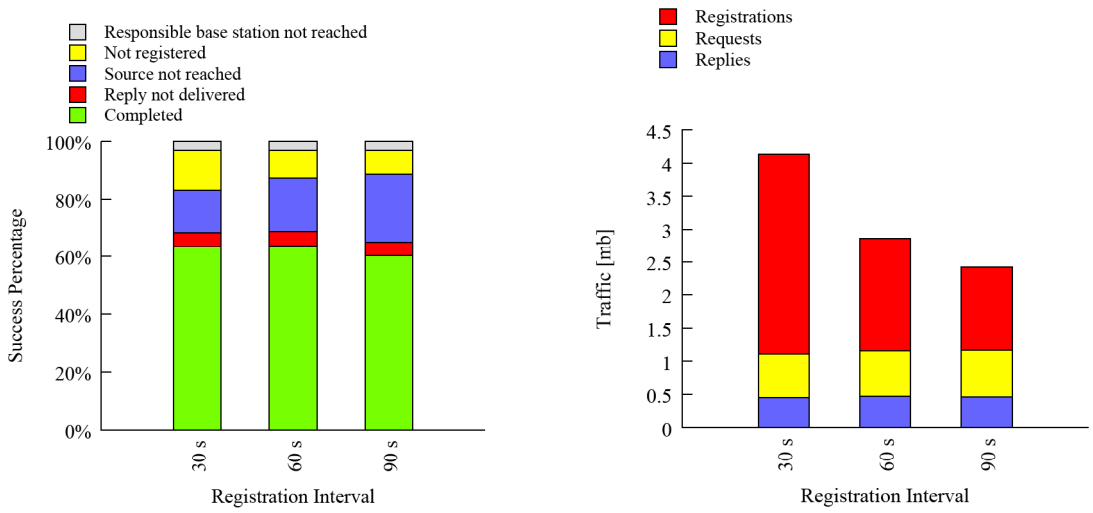


Figure 6.9: EDF: search latency of flooding-based search for 0 m local broadcast radius

in index-based search, local broadcasting does not improve search success rate at this low penetration rate.

Figure 6.8 depicts the network load of requests and replies in flooding with the studied parameters. Here it is observable that the high success rate of flooding comes at the cost of immense bandwidth requirements. The network load of replies is negligible compared with that of requests, so that the different values of prediction period and local broadcast radius do only marginally influence the total load.

The EDF of flooding-based search latency is shown in Figure 6.9. Here, it is interesting to note that the prediction period has no influence on the success rate of the fast end-to-end multi-hop searches. This is different to index-based search, for the following reason: flooding finds all end-to-end multi-hop paths to information sources, independent of the prediction period. Possible multi-hop paths are missed with less position prediction in index-based search, because requests are routed via unicast, where the underlying position information is often outdated.



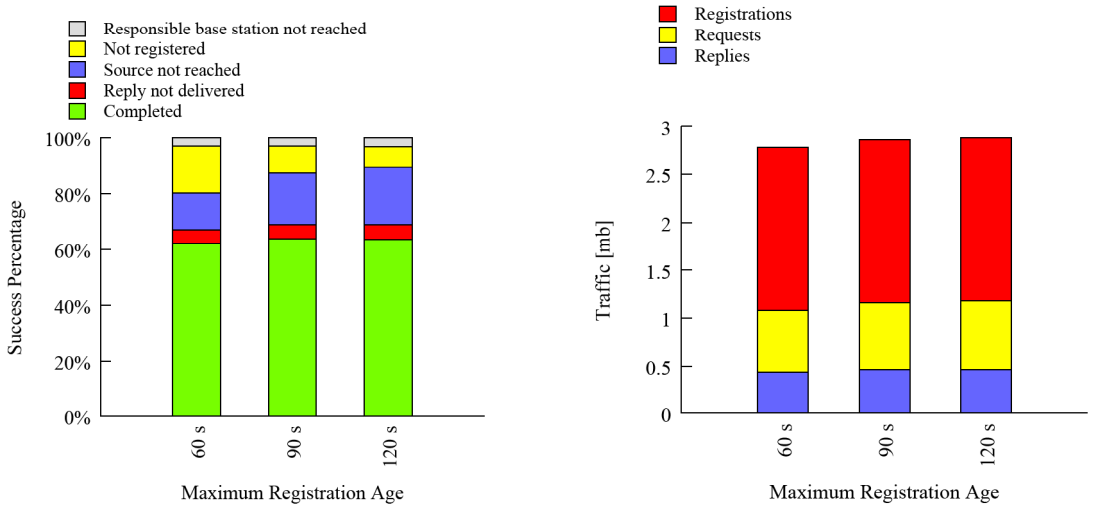
(a) Success rate

(b) Network load

Figure 6.10: Results of different registration intervals

6.4.2.2.3 Registration Interval (Index) Figure 6.10 shows success rate and network load of index-based search at registration intervals of 30 s, 60 s, and 90 s. From 90 s to 60 s, success rate increases because the higher position accuracy leads to more requests being delivered. From 60 s to 30 s network load increases, but success rate does not improve. The main reason is that the assumed position prediction interval is 60 s, which

means that two registrations of the same vehicle arriving in an interval of less than 60 s do not give better position accuracy than if they arrived exactly every 60 s. The graph of search latency is omitted as the EDFs behave analogously to those of Figure 6.6.

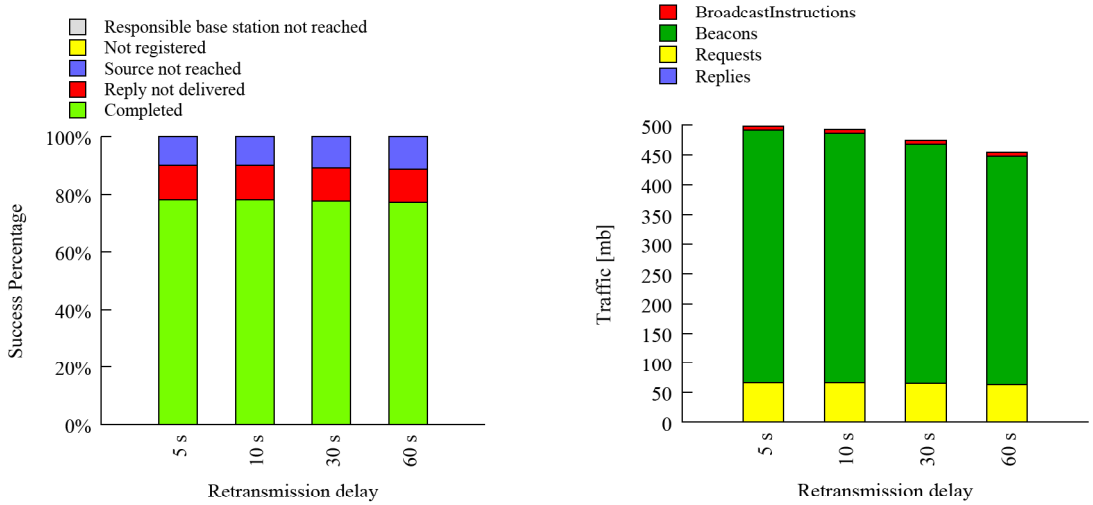


(a) Success rate (b) Network load

Figure 6.11: Results of different maximum registration age values in index. Registration interval is 60 s

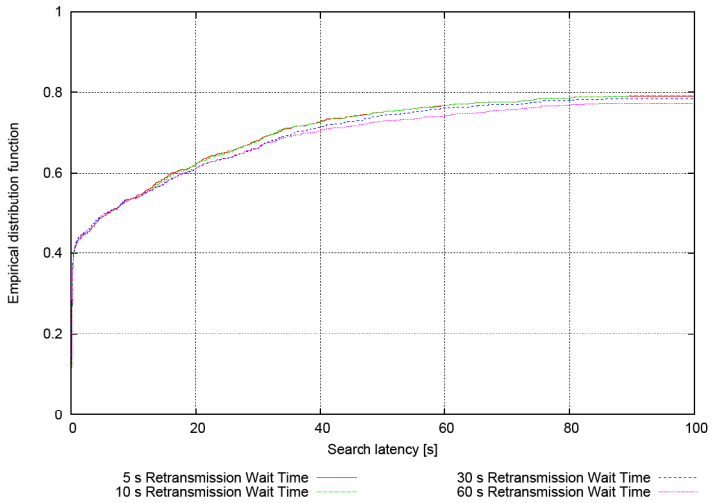
6.4.2.2.4 Maximum Registration Age (Index) Figure 6.11 shows success rate and network load of index-based search depending on the maximum registration age. The registration interval is fixed at 60 s. Success rate slightly increases from 60 s to 90 s. This is because some registrations by vehicles correctly registering every 60 s are considered outdated: while vehicles register every 60 s, the inter-arrival time of a vehicle's periodic registrations in the index varies. Note that, although fewer registrations are removed from the index because they are outdated, success rate only increases for about 1 %–2 %. Thus, the missing registration is a strong indication that the information source may be unreachable. At 120 s, requests redirected to sources based on outdated registration information do not increase the success rate. Network load is increased with higher maximum registration age, because base stations consider even older registrations to be valid and consequently redirect queries to the respective information sources.

6.4.2.2.5 Retransmission Wait Time (Flooding) Figure 6.12(a) shows the success rate of flooding at the studied retransmission delay times. With increasing parameter value, the success rate is only marginally decreased. Figure 6.12(b) shows the network



(a) Success rate

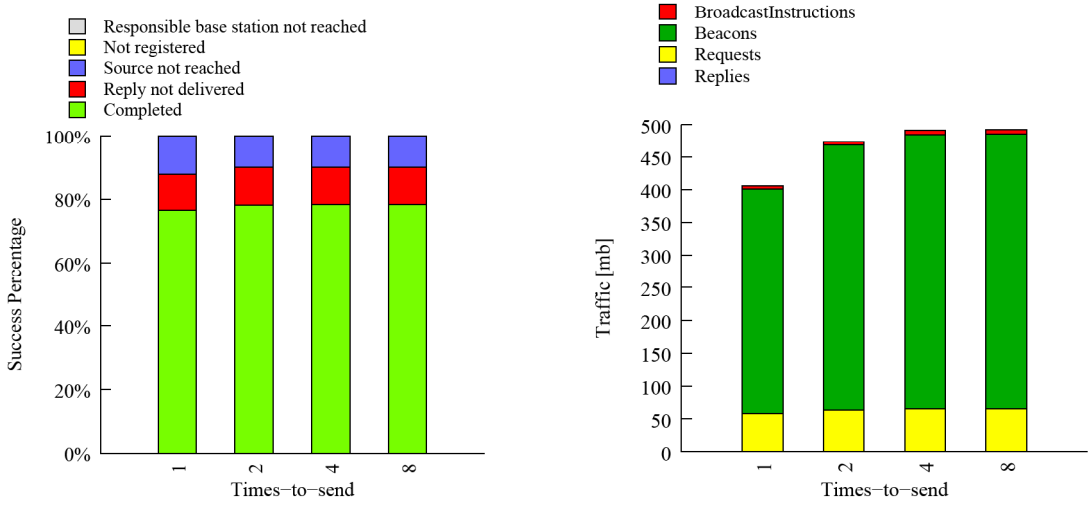
(b) Network load



(c) Latency

Figure 6.12: Results of different retransmission delay times (flooding)

load caused by broadcast instructions, beacons, requests and replies. Recall that beacons are used to advertise message IDs, and broadcast instructions are returned by nodes to whom the message is unknown. The beacons dominate the network traffic; their size reduces with higher retransmission delay (less messages are advertised). We conclude from these results that introducing a retransmission delay is an effective means of reducing the bandwidth required for flooding. The EDF of search latency is depicted in Figure 6.12(c). As expected, the latency of delayed searches—on average—is slightly increasing with higher retransmission delay.

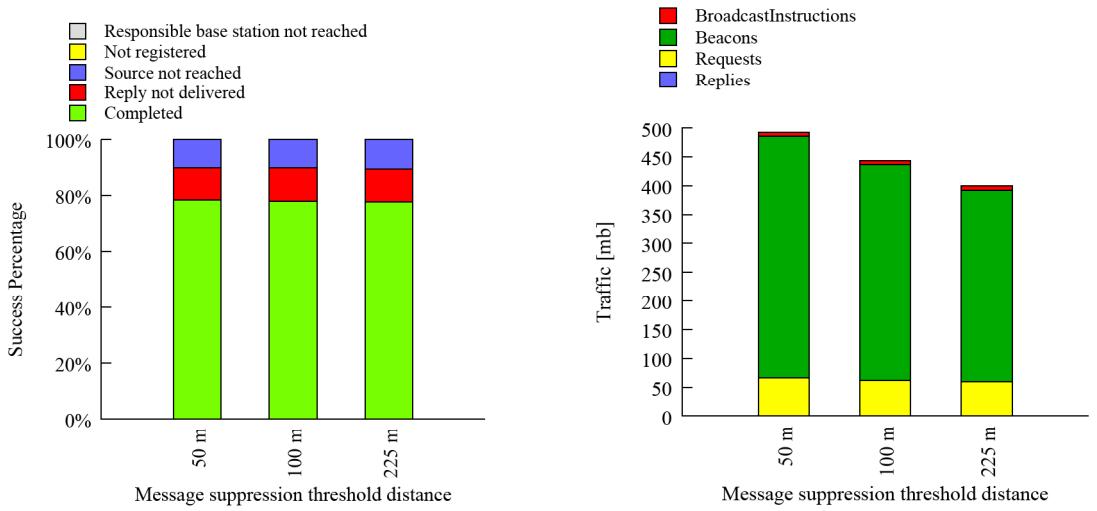


(a) Success rate (b) Network load

Figure 6.13: Results of different times-to-send values (flooding)

6.4.2.2.6 Times-to-send (Flooding) The success rate of flooding at the studied times-to-send parameter values is depicted in Figure 6.13(a). Except for a slight gain from a times-to-send value of 1 to 2, the success rate stays at a level of about 78 %. The success rate increases analogously to the network load (Figure 6.13(b)). Network load is unchanged between values of 4 and 8. This indicates that no vehicle is required to send a single message more than four times. At a value of less than 4, the saving in network load is a result of smaller beacons (messages are not advertised by some nodes any more) and of fewer forwarded requests. Recall that request messages are only forwarded if a node receiving the beacon with the corresponding message ID sends a broadcast instruction; thus, less traffic of request messages at a lower times-to-send value indicates an unsatisfied demand. Nevertheless, down to a value of 2, bandwidth-efficiency of flooding is improved without any noticeable impact on success rate. The

graph of search latency is omitted because the variation of the parameter did not result in any observable change in latency.



(a) Success rate

(b) Network load

Figure 6.14: Results of different message suppression threshold distances (flooding)

6.4.2.2.7 Message suppression threshold distance (Flooding) Figure 6.14 depicts success rate and network load at message suppression distance thresholds of 50 m, 100 m, and 225 m. While network load reduces from about 500 MB at 50 m to 400 MB at 225 m, the success rate is almost constant (there is a drop of 1.5 % between 50 m and 225 m). Thus, distance-based message suppression is another effective means for reduction of network load. The network load reduces with increasing suppression distance because beacon messages become smaller (cf. Section 6.4.2.2.6). Network load of requests decreases only marginally. The graph of search latency is omitted because the variation of the parameter did not result in any observable change in latency.

6.4.3 Parameter Study: Relative Performance

After having determined good parameter sets for each search scheme in the sample scenario, we now compare the two search schemes head-to-head, again investigating different parameters. These parameters are introduced in Section 6.4.3.1.

We evaluated each parameter set in a simulation of 1000 s of vehicular traffic, using three different random seeds of the traffic simulator. The simulation results are presented in Section 6.4.3.2.

6.4.3.1 Discussion of parameter values

Parameter	Default value	Studied values
Penetration rate	-	10 %, 20 %
Replication factor	1	1,2,4
Request rate	200 s	10 s, 100 s, 200 s

Table 6.3: Indirect search parameters and their default values

The parameters listed in Table 6.3 do not directly influence the behavior of the search methods, but are studied to determine the relative performance of the search methods under varying application demands. This way, we intend to give application developers an impression about the suitability of the search schemes under these circumstances.

Let us briefly discuss the indirect parameters. A higher penetration rate allows for a faster communication in the VANET and can be expected to increase the success rate. The replication factor is introduced to study search performance in case multiple replica of an information item are available: a replication factor of r means r subsequent vehicles joining the scenario carry the same information item. These vehicles do not appear at the same location but join the scenario at positions determined by the traffic simulator; thus, the replica are likely to be scattered within the simulated area. We measured the success rate and latency of the search schemes when there are 1,2, or 4 replica of an information item available. To determine the relative network load of both search methods, we studied request rates of 50 s, 100 s, and 200 s. A value of k s means each vehicle issues a new search request every k seconds.

For relative evaluation, the parameters of the search methods were set to the default values shown in Table 6.2, with one exception: we reduced message lifetime to 100 s at 20 % penetration rate, because already at 10 % penetration rate we could not observe any successful searches with a longer delay.

6.4.3.2 Results

6.4.3.2.1 Replication Factor Figure 6.15 depicts the success rates of the two search schemes at penetration rates 10 % and 20 % for replication factors 1,2, and 4. At 10 % penetration rate, flooding achieves an about 15 % better success rate than the index-based scheme. With only one replica available, the success rate of flooding at 10 % penetration rate is about 75 % while that of index-based search is about 60 %. With a total of four replica available, success rate increases to 90 % and 75 %, respectively. At

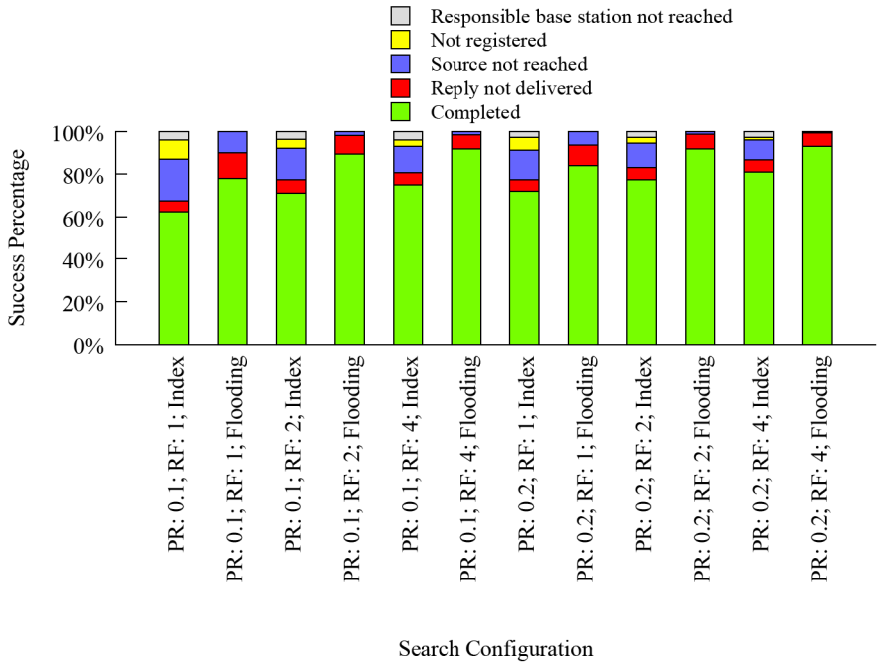


Figure 6.15: Success rate of both search methods depending on penetration rate (PR) and replication factor (RF)

20 % penetration rate, the gap between the success rates of the search methods shrinks to about 10 %. With 4 available replica, success rates are about 90 % for flooding, and 80 % for index-based search. Index-based search benefits from the better consistency of the search index at higher penetration rates: the more dense VANET leads to more successful registrations and allows for faster propagation of search and content requests to information sources.

Figure 6.16 displays the EDF of search latency at 10 % penetration rate. Comparing the leftmost and rightmost values of each curve, in general about two third of all successful searches are completed through an end-to-end multi-hop connection. Flooding finds about 5 % more multi-hop paths than index-based search (about 40 % vs. 35 %), for two main reasons. First, content requests are disseminated very aggressively in flooding. Second, with flooding there is no necessity for reaching a base station at all; if there is an information source nearby, it is addressed directly. With higher replication factor, the chance of having such a nearby information source increases and thus flooding has relatively more successful end-to-end multi-hop searches (about 65 % vs. 50 %). These observations are qualitatively identical at 20 % penetration rate (Figure 6.17), although at an about 10 % higher level of success rate.

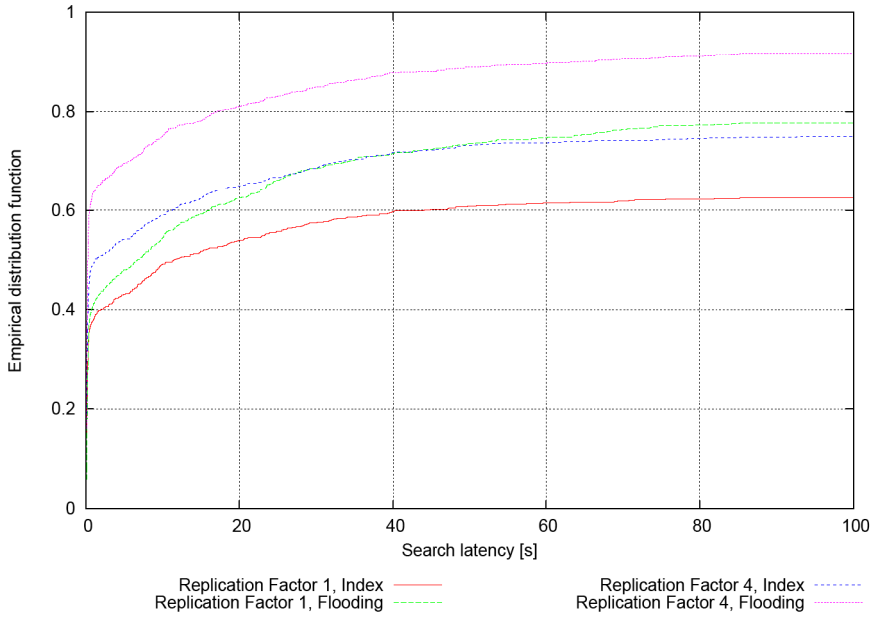


Figure 6.16: EDF: search latency of both search methods depending on replication factor, 10 % penetration rate

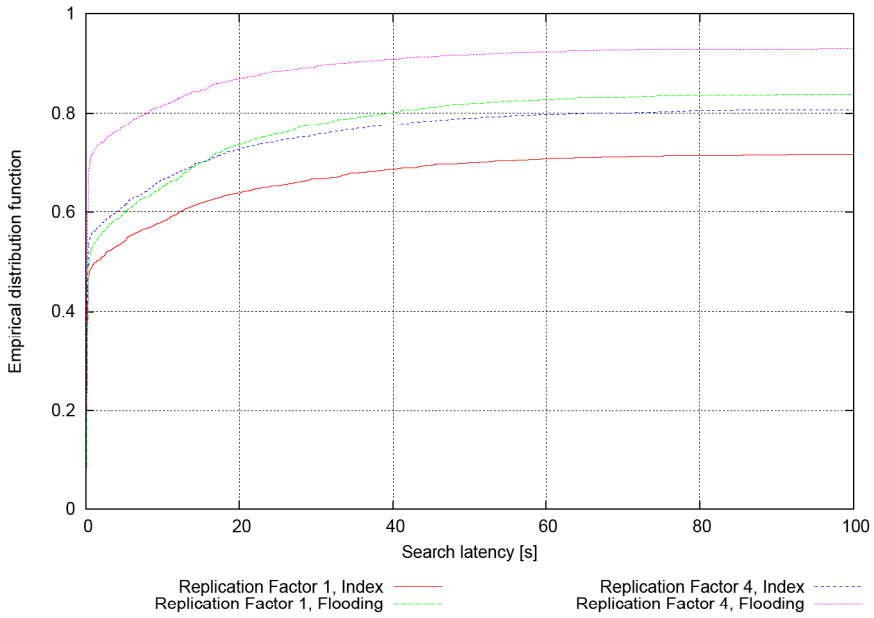


Figure 6.17: EDF: search latency of both search methods depending on replication factor, 20 % penetration rate

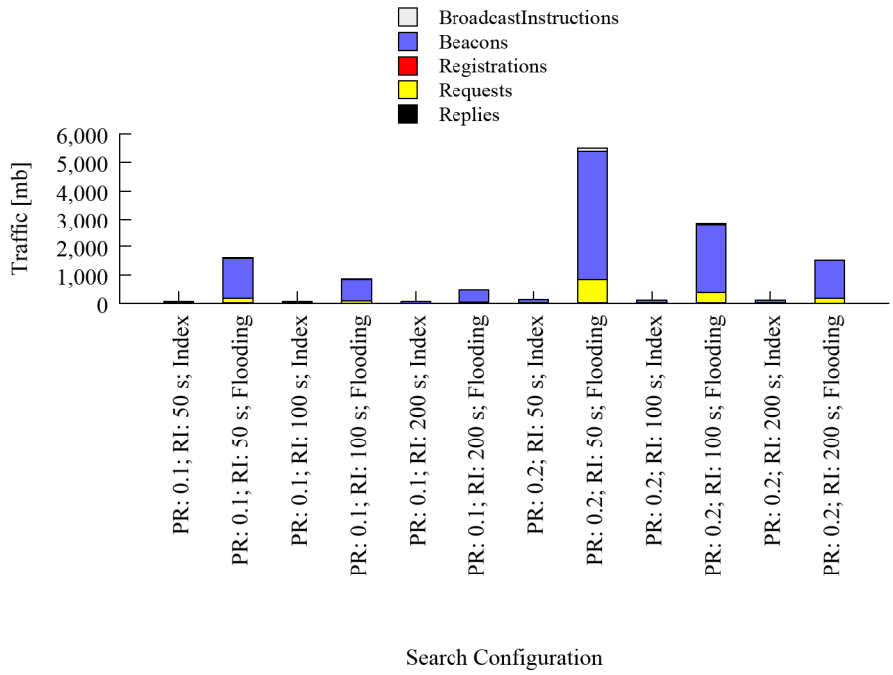


Figure 6.18: Network load of both search methods depending on penetration rate (PR) and request interval (RI)

6.4.3.2.2 Request Rate Figure 6.18 shows the network load of both search methods at the studied request rates. It can be seen that beacons and requests of flooding are the dominating factor in network traffic. The relative difference between the search methods varies between a factor of 4 at 10 % penetration rate at a request rate of 200 s and about 50 at 20 % penetration rate at a request rate of 50 s.

6.5 Conclusions

In the examined scenario, the index-based search scheme achieves an about 15 % worse success rate than flooding at 10 % penetration rate, and performs 10 % worse at 20 % penetration rate. On the other hand, the network load of index-based search is at least a factor of 4 lower than that of flooding, whose content requests and beacons cause immense traffic. The presented suppression schemes for flooded messages help to reduce network load, but not down to a level where it is comparable to index-based search. Under the studied conditions, neither search scheme achieves 100 % success rate; thus, the content-location service is a best-effort service that cannot guarantee acquisition of content at any time.

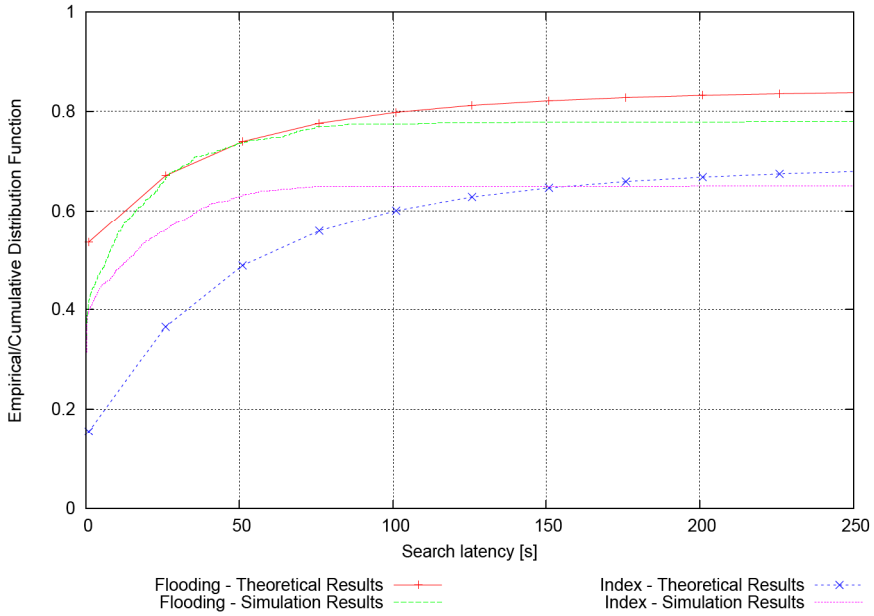


Figure 6.19: EDF/CDF: theoretical and simulation results of search latency of both search methods, 10 % penetration rate

Figure 6.19 depicts our simulation results as well as the curves of flooding and GHT-based search obtained in our feasibility study (Figure 4.25 of Chapter 4) at 10 % penetration rate. The curves of flooding are similar, especially in the rightmost part that reflects the success rate. The simulated flooding scheme achieves a slightly lower success rate and a smaller fraction of end-to-end multi-hop deliveries. To understand this behavior, recall the assumptions underlying our theoretical results: perfect routing, and equal success probabilities for delivery of search request, content request, and content retrieval. In our simulation, content is retrieved not with perfect routing but through Algorithm 5.1.

The curves of the index-based schemes differ because the underlying search indices are distributed in different ways. The implemented distribution of the search index among base stations leads to the positive effect that a significantly higher amount of searches are completed via end-to-end multi-hop connections than if the index was distributed among geographical regions, as assumed in our theoretical model. After 150 s, the success rate of the theoretical index-based scheme exceeds that of the simulated scheme. Here, the simulated scheme is based on realistic routing and position prediction capabilities, while the theoretical results imply a perfect routing scheme which allows for message delivery even after long periods of time.

There are several ideas for further work on the presented seach schemes. First, it would be interesting to see how the index-based scheme performs if the index was not distributed among the base stations but among vehicles near base stations. Second, the bandwidth usage of the flooding scheme could be further reduced. Instead of advertising message IDs, these IDs might more compactly be encoded into a Bloom filter [VB00]. Besides that, the number of message IDs a node may advertise in its beacon could be bounded and messages then could be ranked for advertising, e.g. according to their age.

6.6 Chapter Summary

In this chapter we have implemented and evaluated two search schemes for infrastructure-supported sparse VANETs: flooding-based search and search based on a distributed index. The flooding-based scheme includes message suppression strategies to improve its bandwidth-efficiency. The search index underlying the index-based variant is distributed among the base stations in the VANET, which is expected to result in low additional communication effort for querying the index since base stations are often involved in efficient communication between distant vehicles anyway.

The performance of both search methods depends on the choice of several parameters. We have conducted a parameter study, identifying the parameter sets that lead to high success rate and low network load for each search scheme. Besides that, the parameter study has underlined the effectiveness of the message suppression schemes in the flooding variant. Assessing the relative performance of the search methods, the proposed index-based scheme still has a slightly lower success rate than flooding, but its low network load makes index-based search a good practical choice.

The success rate and latency of the implemented flooding-based scheme is close to that of our theoretical model (Chapter 4). The index-based scheme improves the performance of the theoretical model of a GHT in the sense that the index distribution among base stations results in more searches being completed in the order of a few seconds. Because of our imperfect routing algorithm, the overall success rates of the search schemes are slightly lower than the theoretical best.

Chapter 7

Probabilistic Search Indices

The search index introduced in the previous chapter is maintained by vehicles periodically registering hash keys of their locally held information items. Thus, the size s of each vehicle's registration increases linearly with the number n of its information items: $s = c \cdot n$. If, like in the previous chapter, keys are calculated through a fixed-length hash function, the constant c of this formula corresponds to the length of the hash function.

In this chapter, we investigate Bloom filters as an alternative encoding of vehicles' information items. A Bloom filter is a data structure allowing for probabilistic membership tests. It is probabilistic, because checking the filter for membership of an information item may yield a false positive: the result is "true", although the item is not contained in the filter. The size of a Bloom filter may be reduced through compression. We examine both uncompressed and compressed Bloom filters.

The required bandwidth of registrations based on Bloom filters is compared to that of an apparent alternative: using shorter hash values as keys. Note that hash values in general also bear the possibility of false positives: the shorter the hash function, the higher the chance of two information items being mapped to the same hash value.

While our initial calculations in this chapter assume that each node registers individually, it is also conceivable that nodes register in a common Bloom filter. We first study theoretical aspects of such aggregated registrations before outlining how aggregates may effectively be constructed in a VANET. Finally, we verify the feasibility and efficiency of Bloom filter-based registrations and of the aggregation scheme in a simulation study.

This chapter continues with a discussion of related work (Section 7.1), consisting of two parts: an introduction of Bloom filters and other data structures allowing probabilistic set membership tests, as well as a review of in-network aggregation techniques. Indices consisting of Bloom filters and indices consisting of short hash values are defined and

analytically compared in Section 7.2. In Section 7.3 we calculate whether in-network aggregation of nodes' Bloom filters or hash keys reduces the false-positive rate of the index. Section 7.4 investigates the feasibility and bandwidth reduction when vehicles register using compressed Bloom filters instead of 64-bit hash values in the VANET scenario. We present our conclusions in Section 7.5 and summarize the chapter in Section 7.6.

7.1 Related Work

In the first part of this section (Section 7.1.1), we survey data structures allowing probabilistic set membership tests. The second part (Section 7.1.2) addresses related work on in-network data aggregation.

7.1.1 Probabilistic Membership Tests

The original Bloom filter [Blo70] has been developed as a space-efficient data structure capable of answering set membership queries either with yes or no, at the risk of false positives. Thereafter, several extensions have been proposed, providing Bloom filters with the ability of counting and deletion of members [FCAB00, CM03], or allowing them to store values (associative array) [CKR+04]. Boldi and Vigna generalized Bloom filters, obtaining a data structure called *compact approximator* [BV05]. With its help, functions can be approximated in the sense that the function value returned by the compact approximator is equal or greater than the real function value. Boldi and Vigna found this to be useful in text search based on the Boyer-Moore algorithm [BM77].

Because of their space-efficiency, Bloom filters have many applications in networks where bandwidth is costly. Here we focus on applications in MANETs; for a broader survey, we refer to [BM03]. In MANETs, Bloom filters have been suggested for discovery of services and resources [SI05, LH07]. Presence detection [TSM07] is another application, where nodes share an aging Bloom filter containing the IDs of active nodes in the MANET. The common element of the presented applications is that nodes proactively distribute the filter either in their close vicinity or in the whole MANET. In contrast, our approach is to send the filter solely to a single location.

An important question for our goal of minimizing network load is whether there exist even more space-efficient data structures than the original Bloom filter. In this context, the question arises whether it is possible to give an information theoretical lower bound

for the space m required by any data structure capable of representing sets of size n with a false-positive rate of R. In fact, this lower bound is

$$m \geq n \cdot \log_2(1/R). \tag{7.1}$$

A derivation of this bound can be found in [BM03]. It turns out that the Bloom filter uses a factor of approximately 1.44 more space than a space-optimal data structure with same functionality. Mitzenmacher [Mit01] has shown that this can be improved by compression, after which Bloom filters can theoretically achieve space-optimality. In this chapter we consider both uncompressed and compressed Bloom filters for the search index.

Broder and Mitzenmacher [BM03] also give an example of a space-optimal data structure. This data structure is a hash table of n entries with j bits each. To represent a set N of n information items, one employs a minimal perfect hash function [CHM97] h_1, assigning a unique hash value in the range of $\{0, \ldots, n-1\}$ to each information item $i \in N$, thereby obtaining its location in the hash table. Instead of storing i, one stores $h_2(i)$, where h_2 is a j-bit uniform hash function. The resulting $n \cdot j$-bit hash table has a false-positive rate of 2^{-j}. Note that the total amount of space required for transmitting this information is the size of the hash table plus the size of the perfect hash function itself (which is at least $O(n)$ [FK84]), because both need to be transmitted so that the receiver can decode the information. Besides this, perfect hashing has the drawback of high computational effort required to obtain the perfect hash function for a set. d-ary cuckoo hashing [FPSS05] is a computationally-efficient alternative to perfect hashing, providing almost the same space-efficiency of the hash table. It allows creation of a hash table with $(1+\epsilon) \cdot n$ entries when containing n information items. For approximate membership queries one can again store uniform hash values of length j in the hash table. Compared to the minimal perfect hashing scheme, cuckoo hashing has the advantage that transmission of the hash function is not required. On the other hand, queries require lookup of up to d entries, essentially leading to a false-positive rate higher than that of the perfect hashing scheme. Our list of short hash values can be viewed as the hash tables in absence of a hash function (perfect or cuckoo hashing), thus requiring the checking of each of the n list entries for the sought-after value. Consequently, constructing the list is more computationally-efficient, but checking the n list entries leads to a false-positive rate that increases with n.

Hash compaction is another hash table-based approach, first published by Wolper and Leroy [WL93] for the purpose of a more compact representation of visited states in

program verification, thereby tackling the state space explosion problem. It has later been improved by Stern and Dill [SD96], whose method we briefly describe here. To insert an information item in the hash table, a compact signature of the information item and a probe sequence of possible locations are computed based on multiple hash functions. For insertion and querying the hash table, Stern and Dill propose the use of ordered hashing, invented by Amble and Knuth [AK74], with the goal of overcoming the high worst case query time of the open addressing employed in the original hash compaction scheme. In ordered hashing, the hash table is probed according to the probe sequence and the signature of the information item is stored at the first location that is either empty or contains a higher signature value. In the latter case, a new location for the old information item has to be found, potentially leading to many subsequent information item shifts. To query for an information item, the hash table is checked for the signature of the sought-after information item according to the probe sequence. The query algorithm stops in three cases. First, if the checked slot is empty or contains a higher signature value, indicating that the hash table does not contain the information item. Second, if one of the probed slots contains the signature of the information item, indicating that it is part of the hash table, although—like a Bloom filter—with some remaining probability of a false positive. Third, if the complete list of slots given by the probe sequence has been checked without a result. Dillinger and Manolios evaluated a configuration of hash compaction against optimally configured uncompressed Bloom filters and found hash compaction to be more space-efficient [DM04]. It remains unclear how far the space required by the outlined approach is away from the minimum. Here we do not further investigate hash compaction because of its drawback of either inefficient insertion or querying of information items.

7.1.2 In-network Data Aggregation

In-network aggregation of data, like we propose for Bloom filters in Section 7.4.1, is an active field of research in the context of wireless sensor networks. The work of Fasolo et al. [FRWZ07] surveys research results in this area. The authors classify aggregation techniques into either lossy or lossless and into duplicate-sensitive or duplicate-insensitive.

Our proposed local opportunistic aggregation scheme for metropolitan VANETs is lossy in the sense that only the identity of the node aggregating the filters is preserved in the resulting message while the identities of the other nodes are lost. Our scheme is lossless in the sense that the aggregated Bloom filter contains the same information at the same false-positive rate as the atomic Bloom filters. Our scheme is duplicate-insensitive: the

same bits are set in a common Bloom filter, no matter how many nodes register a particular information item.

The authors of [FRWZ07] further mention that the theoretical limit of bandwidth reduction through aggregation techniques depends on the correlation of data that has to be aggregated. All our calculations in this chapter are based on the assumption that each node carries unique information items. Thus, the resulting savings in network load and false-positive rate of the search index represent the worst case.

Researchers have also proposed deterministic and probabilistic aggregation mechanisms for VANETs. Caliskan et al. define a deterministic hierarchical aggregation scheme for parking information [CGM06]: each vehicle disseminates detailed counts of free parking spots in its vicinity and only sums over parking spots in more distant regions. This way, vehicles have precise knowledge about parking spots in their surrounding and a coarse view of the parking situation in distant areas. Lochert et al. show that bandwidth can further be reduced if data such as parking lot occupancy is probabilistically aggregated [LSM07]. In their work, vehicles do not exchange total parking lot occupancies but count single available parking spots and store their observations in a soft-state Flajolet-Martin sketch. Aggregating these sketches is duplicate-insensitive.

7.2 Probabilistic Index Types

Assume there are t nodes $n_1, n_2, \ldots, n_t$ in the network, each registering its information items in the search index. Commonly, they would register standard hash keys of all their information items. In a probabilistic[1] search index as we propose it here, they transmit a data structure which probabilistically represents their whole set of information items—thereby reducing the amount of data to be transmitted, but potentially causing false positives in lookup operations. In this section, we analyze different candidate data structures for probabilistically registering sets of information items in an index.

Assume that node n_i uses some probabilistic data structure for representing the set of its information items. This representation will exhibit a certain false-positive rate (FPR) R_i, which depends on the type, size, and parameters of the probabilistic representation and on the number of information items hosted by the node. In the index, given that representations of the information items of all t nodes have arrived, all registrations can

[1] As pointed out in the introductory part of this chapter, even an index consisting of standard hash keys is non-deterministic; we here use the term *probabilistic* indicating that we think of much shorter representations than given by standard hash functions.

be checked whether they contain some given information item. Then, the total FPR $\overline{R}$ of the index is

$$\overline{R} = 1 - \prod_{i=1}^{t}(1 - R_i). \tag{7.2}$$

From now on, we consider the situation where each node adjusts size and parameters of its own information set representation to achieve a given, fixed FPR R. Then, the FPR of the index is

$$\overline{R} = 1 - (1 - R)^t. \tag{7.3}$$

Note that in such a system it is perfectly admissible that each node individually selects the format and parameters used for representing its set of information items (as long as the respective format is also understood by the nodes at which the index is stored and queried). In order to save communication overhead, we want to minimize the size of this representation for a given FPR R. Therefore, in the remainder of this section, we derive formulas for the sizes of registration information based on Bloom filters (Section 7.2.1), compressed Bloom filters (Section 7.2.2), and short hash lists (Section 7.2.3), and compare them based on R and the number of information items n (Section 7.2.4).

7.2.1 Bloom Filter-based Registration

A Bloom filter is a probabilistic representation of a set I of information items $\{i_1, i_2, \ldots, i_n\}$. The filter allows membership queries, answered with *yes* or *no*. If the answer is *no*, the queried information item is not part of the filter. If the answer is *yes*, it can be assumed that the information item is part of the filter with some remaining probability of a false positive. The filter consists of a bit array $B = b_1, \ldots, b_m$. k independent hash functions $h_1(i), \ldots, h_k(i)$ each map an information item i to values in the range $\{1, \ldots, m\}$. When an element i is inserted, the corresponding bits $b_{h_1(i)}, \ldots, b_{h_k(i)}$ are set to 1. A membership query for i is answered with *yes* if and only if all the bits $b_{h_1(i)}, \ldots, b_{h_k(i)}$ are set to 1.

If a Bloom filter is to be adjusted for registering n information items with FPR R_{BF} in an index, a node calculates the required number of bits m of the filter, and subsequently constructs the filter by inserting the information items. Following Broder and

Mitzenmacher [BM03], for a given size m, number of hash functions k, and information count n, the FPR R_{BF} can be computed as[2]

$$R_{\mathrm{BF}} = (1 - e^{\frac{-kn}{m}})^k = (1 - e^{\frac{-k}{c}})^k. \tag{7.4}$$

Analysis of this function reveals that for given R, $\frac{m}{n}$ is minimal for $k = k_{\mathrm{opt}} = \ln(2)\frac{m}{n}$. Then:

$$R_{\mathrm{BF}} = 0.5^{\frac{\ln(2)m}{n}} \tag{7.5}$$

In practice, of course k must be an integer.

Therefore, for given FPR R_{BF} and information count n, the size of an optimally constructed Bloom filter is

$$L_{\mathrm{BF}}(n, R_{\mathrm{BF}}) = n \cdot \frac{\ln(R_{\mathrm{BF}})}{\ln(0.5^{\ln(2)})} = n \cdot \frac{\ln(R_{\mathrm{BF}})}{\ln(2) \cdot \ln(0.5)}. \tag{7.6}$$

Note that, for a fixed FPR R_{BF}, the size of the Bloom filter is linear in n. Let $c_{\mathrm{BF}} := \ln(R_{\mathrm{BF}})/(\ln(2) \cdot \ln(0.5))$. When using Bloom filters we need c_{BF} bits per information item for the transmission of the registration information to the index, where c_{BF} depends solely on the desired FPR.

7.2.2 Using Compressed Bloom Filters

A way to further reduce the network load is to compress the Bloom filter, as proposed by Mitzenmacher [Mit01]. Generally, the entropy limit is a lower bound for the achievable compressed size of information. When $k = k_{\mathrm{opt}}$ for a Bloom filter, each bit is set with probability 1/2. For all practical purposes, the bits can also be considered independent. Consequently, the entropy of each bit in the filter is 1, such that the filter can not be further compressed. Nevertheless, Mitzenmacher finds that a smaller size $z < m$ can be achieved by selecting k and m in a way that reduces the entropy of the resulting Bloom filter while maintaining the same FPR. In theory, the lower bound for the achievable FPR for a given compressed size z is $R_{\mathrm{CBF}} = 0.5^{z/n}$.

[2] This formula is an asymptotic approximation more convenient for analysis than the exact term. It holds for $kn < m$, which we imply here.

Consequently, an ideal compressed Bloom filter, depending on number of information items n and the FPR R_{CBF}, has a size of

$$L_{\mathrm{CBF}}(n, R_{\mathrm{CBF}}) = n \cdot \frac{\ln(R_{\mathrm{CBF}})}{\ln(0.5)} = n \cdot \log_2(1/R_{\mathrm{CBF}}). \tag{7.7}$$

This is again linear in n, but now with a factor of $c_{\mathrm{CBF}} = \ln(R_{\mathrm{CBF}})/\ln(0.5) = \ln(2) \cdot c_{\mathrm{BF}}$, i. e., compressed Bloom filters save up to 30 % space while achieving the same FPR. The required space meets the information theoretical lower bound (Equation 7.1), thus the compressed Bloom filter is space-optimal.

However, it is not possible to achieve this in a real system. The theoretical bound of $R_{\mathrm{CBF}} = 0.5^{z/n}$ holds when k goes to 0 or infinity and m goes to infinity. In practice, k has to be an integer greater or equal than 1. It also cannot be arbitrarily large because this increases the computational effort of insertion and lookups. Memory constraints additionally limit the size of m. Here, we point out that in practice, if we choose some fixed number of bits c to be used per information item in the registration, we are always better off using compression, that is, $0.5^c < R_{\mathrm{CBF}} \leq 0.5^{\ln(2)c}$.

For practical relevance, we investigate the size of compressed Bloom filters with $k = 1$ and $k = 2$. We proceed as follows: we calculate the size m of the uncompressed filters depending on desired FPR R and information count n. Assuming an ideal compressor, we calculate their compressed size using the entropy. (In practice, compression very close to the entropy limit is possible for Bloom filters, e. g. by using arithmetic coding [WNC87]. We verify this in Appendix A.)

For $k = 1$, transforming Equation 7.4 yields

$$m_{\mathrm{k=1}} = \frac{-n}{\ln(1 - R_{\mathrm{CBF}})}. \tag{7.8}$$

The probability p that any given bit in the uncompressed Bloom filter with $k = 1$ is 0 is $p = (1 - 1/m)^n$ [BM03]. The bits are practically independent. Thus, the entropy of each bit is

$$\begin{aligned} H_{\mathrm{k=1}} = & -p \log_2 p - (1-p) \log_2 (1-p) \\ = & -(1 - 1/m_{\mathrm{k=1}})^n \log_2((1 - 1/m_{\mathrm{k=1}})^n) \\ & -(1 - (1 - 1/m_{\mathrm{k=1}})^n) \log_2(1 - (1 - 1/m_{\mathrm{k=1}})^n)) \end{aligned}$$

and

$$L_{\text{CBF, k=1}}(n, R_{\text{CBF}}) = m_{\text{k=1}} \cdot H_{\text{k=1}}. \tag{7.9}$$

For $k = 2$,

$$m_{\text{k=2}} = \frac{-2n}{ln(1 - R_{\text{CBF}}^{1/2})} \tag{7.10}$$

In this case, $p = (1 - 1/m)^{2n}$, thus

$$\begin{aligned} H_{\text{k=2}} = & -(1 - (1/m_{\text{k=2}})^2)^{2n} \log_2((1 - (1/m_{\text{k=2}})^2)^{2n}) \\ & -(1 - (1 - (1/m_{\text{k=2}})^2)^{2n}) \log_2(1 - (1 - (1/m_{\text{k=2}})^2)^{2n})) \end{aligned}$$

and

$$L_{\text{CBF, k=2}}(n, R) = m_{\text{k=2}} \cdot H_{\text{k=2}}. \tag{7.11}$$

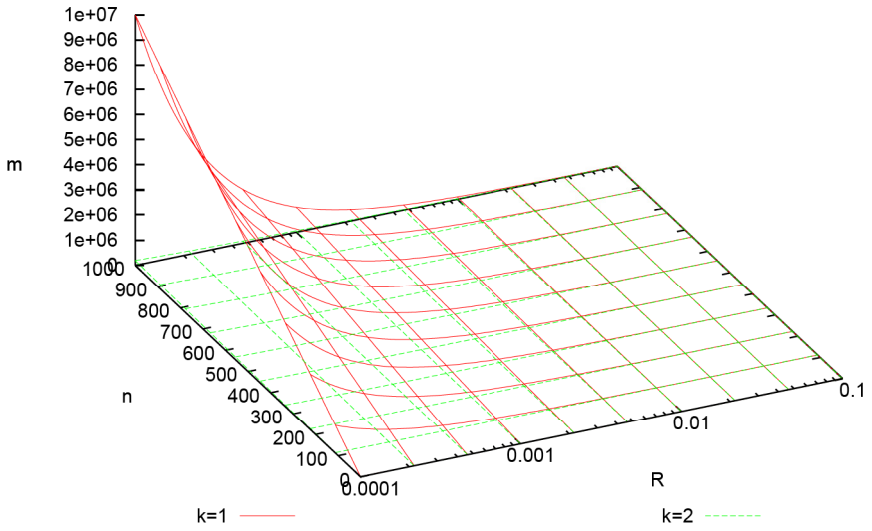


Figure 7.1: Size m of uncompressed Bloom filters with $k = 1$ and $k = 2$.

Figure 7.1 compares $m_{k=1}$ and $m_{k=2}$. Obviously, the Bloom filter with $k = 2$ has

advantages for practical use because the memory requirements of the uncompressed filter are much lower than in case of $k = 1$.

It has to be noted that compression and decompression increase the computational overhead. On the other hand, in practice good compression may be achieved by selecting a small number of hash functions k ($k < k_{\text{opt}}$) and a matching (larger) size m of the filter. This means that the computational overhead of insertion and query operations is reduced.

7.2.3 Truncated Hash List-based Registration

Alternatively to using Bloom filters for probabilistically representing the information items of a node, it is also conceivable to use a list of short hash keys. This as well bears the possibility of a false positive: the key in a lookup operation might have the same hash value as one of the keys in the list. The probability of such an event obviously depends on the number of elements in the list, again denoted by n, and the length of the hash values c. Assuming good hash functions, the FPR is given by

$$R_{\text{THL}}(n, c) = 1 - (1 - 2^{-c})^n. \tag{7.12}$$

Note that hash values of all reasonable sizes can easily be constructed by using a standard hash function (e. g., MD5 [Riv92] or SHA-1 [EJ01]) and truncating it after the desired number of bits. For a given FPR R_{THL} and n information items, the required total size of such a *truncated hash list* (THL) is therefore

$$L_{\text{THL}}(n, R_{\text{THL}}) = n \cdot c = n \cdot \frac{-\ln(1 - (1 - R_{\text{THL}})^{\frac{1}{n}})}{\ln(2)}. \tag{7.13}$$

Given that good hash functions are used, the entropy of each bit in the THL is 1. Consequently, the size of transmitted THLs can not be further reduced by compression.

7.2.4 Comparison

We now determine the conditions under which one of the index types described above—based on (compressed) Bloom filters, or THLs—is preferable. We say that one index type is *preferable*, if for a given FPR R, the registration of this index type is smaller

than in the other index types. For this purpose, we investigate L_{BF}, L_{CBF} and L_{THL} more closely.

Theorem 7.1. *The ideal compressed Bloom filter is smaller than the THL for all* $0 < R < 1$ *and* $n > 1$.

Proof For $0 < R < 1$ and $n > 1$ we have

$$\begin{aligned} & & (1-R)^{\frac{1}{n}} > & & (1-R) \\ \Leftrightarrow & & \ln(1-(1-R)^{\frac{1}{n}})/\ln(R) > & & 1 \\ & \Leftrightarrow & L_{\text{THL}}/L_{\text{CBF}} > & & 1 \\ & & \Leftrightarrow \quad L_{\text{THL}} > & & L_{\text{CBF}} \end{aligned}$$

□

However, as pointed out in Section 7.2.2, this result is of theoretical nature in the sense that ideal compression cannot be achieved in practice.

For practical relevance, we investigate the size of the compressed Bloom filters with $k = 1$ and $k = 2$ compared to a THL with the same FPR. For this it is required to compare Equation 7.9 and Equation 7.11 to Equation 7.13 under the condition that the FPR is equivalent. An analytical comparison seems challenging; here we present numerical results in Figure 7.2, which shows sizes of a THL and compressed Bloom filters with $k = 1$ and $k = 2$ depending on number of information items n and FPR R. It can be concluded that, starting from a very small number n of information items per registration, compressed Bloom filters are smaller. The compressed Bloom filter with $k = 2$ is marginally larger than with $k = 1$, but it has practical advantages because of its much smaller uncompressed size m.

In some situations, compressed Bloom filters cannot be used, for instance, due to the computational overhead or because of memory constraints that do not allow to store the uncompressed filter in the nodes. Therefore, we also investigate the tradeoff between uncompressed Bloom filters and THLs. As the following theorem shows, up to a certain number of information items n, the THL is smaller than the uncompressed Bloom filter. For larger n, the Bloom filter is smaller than the THL.

Theorem 7.2. *For all* $0 < R < 1$, $L_{THL} < L_{BF}$ *if and only if*

$$n < \frac{\ln(1-R)}{\ln(1 \quad R^{1/\ln(2)})}$$

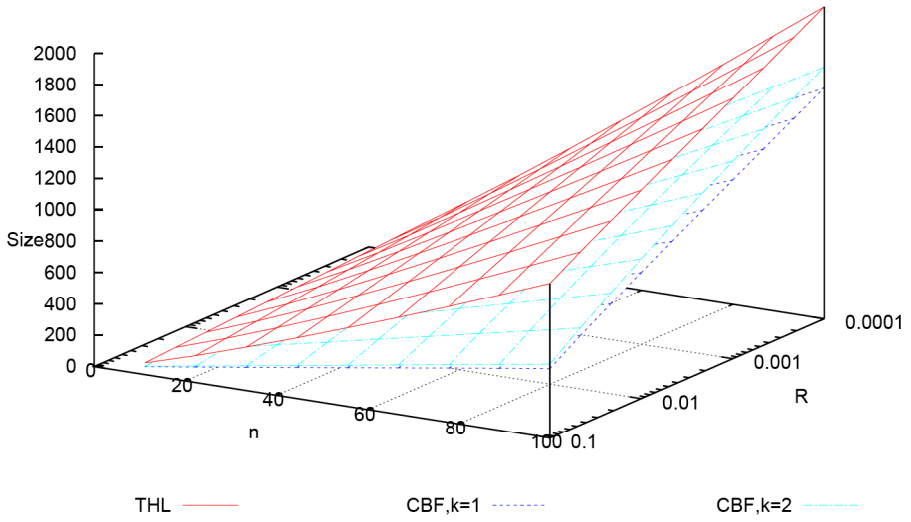


Figure 7.2: Size of compressed Bloom filter with $k = 1, k = 2$ and THL with equal FPR.

Proof

$$
\begin{aligned}
L_{\text{THL}} &< L_{\text{BF}} \\
\Leftrightarrow \quad L_{\text{THL}}/L_{\text{BF}} &< 1 \\
\Leftrightarrow \quad \frac{\ln(1-(1-R)^{\frac{1}{n}})}{\ln(R)} \cdot \ln(2) &< 1 \qquad (7.14) \\
\Leftrightarrow \quad (1-R)^{\frac{1}{n}} &< (1-R^{1/\ln(2)}) \\
\Leftrightarrow \quad n &< \frac{\ln(1-R)}{\ln(1-R^{1/ln(2)})}
\end{aligned}
$$

□

Figure 7.3 depicts the value of n for which the sizes of THL and Bloom filter are equal, depending on the desired FPR R (please note the logarithmic x-axis). For small R, the THL is more space-efficient up to a large number of entries (e. g., for $R = 10^{-7}$, $n \approx 1250$). For larger R, the Bloom filter is preferable already for a small number of registered information items (e. g., for $R = 0.01$, $n \approx 8$).

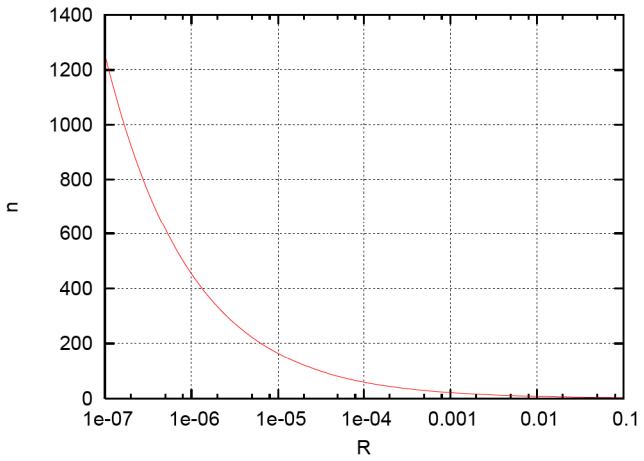


Figure 7.3: Number of registered information items n for which $L_{\text{THL}} = L_{\text{BF}}$.

7.2.5 Consequences on Design of a Probabilistic Index

The analysis conducted in the previous section indicates that in theory, compressed Bloom filters are most efficient independent of number of information items per node and desired FPR. Compressed Bloom filters with $k = 2$ are a very good practical choice for a probabilistic index.

If compression is no option due to memory constraints or computational requirements, the situation is not that clear: neither uncompressed Bloom filters nor THLs are generally more efficient. Instead, the achievable size depends on the number of information items in the node n and on the required FPR R. Thus, to minimize network load in networks where n cannot be predetermined to be above or below the break-even point at all times and for all nodes, it is an option to implement both index types and to choose one on demand. In this case, each node, given an FPR R, can then calculate the break-even point of Bloom filter and THL and check whether its individual number of information items is higher. If this holds, the node registers using a Bloom filter. Otherwise, it registers with a THL.

Regarding the value of R, the search index can be set up such that R is either static or dynamic: in the former case, one fixes R for all nodes in the network. Thus, if the number of nodes t changes, the overall FPR $\overline{R}$ of the search index varies. In the latter case, one fixes $\overline{R}$ instead and adjusts R dynamically, depending on the current number of nodes in the network. This number could for example be estimated on the basis of proactively distributed information.

Through Bloom filters, content can be registered more bandwidth-efficiently than through hash-keys, which is our primary goal in this chapter. Nevertheless, one should be aware of the accompanying trade-offs of a Bloom filter-based index. We have identified two such trade-offs and outline these in the following, briefly discussing possible solutions.

First, the network load of requests is higher with Bloom filters in the index, because sending a single hash value (as in a DHT query) is not sufficient to query them. Recall from Section 4.1.3 that nodes apply a well-defined description scheme, associating each local information item I with a description D_I, which is the input to the k hash functions giving the bit numbers in the Bloom filter. To query the filters in the search index for an item, the k hash keys of D_I have to be included in the request. As apparent alternative, D_I itself could be sent, allowing index nodes to compute the k hash functions, but potentially causing higher network load. As another idea, there is a way to query Bloom filters with k hash functions by only sending two hash keys.[3] This requires a slightly different construction of the filters: Kirsch and Mitzenmacher in [KM08] showed that the k hash values $h_0, \ldots, h_{k-1}$ for Bloom filters can be obtained through only two (independent and fully random) hash functions $g_1(x)$ and $g_2(x)$:

$$h_j(x) = (g_1(x) + jg_2(x)) \bmod m \text{ for } j = 0, \ldots, k-1 \tag{7.15}$$

A Bloom filter constructed in this way has the same asymptotic FPR as the standard Bloom filter. For checking the index, it thus suffices to transmit $g_1(x)$ and $g_2(x)$.

Second, if one is to find all registered sources for an information item, all filters in the index have to be checked, and thus all index nodes must be contacted. In a DHT, only the node responsible for the respective part of the key space has to check its index entries. There is an alternative Bloom filter construction, suggested by Putze et. al in [PSS07], that can be used in a distributed index so that only one node has to check its index entries, although at the cost of an increased FPR. The idea is to divide the Bloom filter bit array into d blocks: we compute a hash function modulo d to obtain a block ID, and then use the k hash functions to set bits only locally in that single block. In the index, the Bloom filter is split and the blocks are distributed among index nodes. When the index is queried for an item, the respective block ID is computed through the hash function and only the corresponding index node needs to check its blocks. The FPR of a blocked Bloom filter is higher than that of a standard Bloom filter, especially

[3] For compressed Bloom filters, we already have outlined that one or two hash functions are a good practical choice—thus, only these need to be sent in a query. The following approach addresses the generic case of Bloom filters with k hash functions

if the single blocks are small: the distribution of information items over blocks is not uniform—the block loads follow a binomial distribution. A trade-off between FPR and number of accessed blocks can be achieved through multi-blocking, where X blocks may be accessed and k/X bits are set in each block [PSS07].

7.3 In-Network Aggregation of Registrations

Our preceeding analysis assumes that the number of registrations in the search index is t when there are t nodes in the network. Then, the search index consists of t registrations, each of which is either a Bloom filter or a THL and contains a varying number of registered information items. (Note that uncompressed and compressed Bloom filters are practically equivalent, once they have arrived in the index.) The number of entries t can be reduced when the registration information of multiple nodes is combined. We call this *aggregated registration*. This is desirable for two reasons: first, the overhead incurred by packet headers can be reduced. Second, it decreases the FPR of the index in case of the Bloom filter, as the following analysis will show.

Information items from multiple nodes may be aggregated by these nodes before transmitting them to the index. If a group of nodes all use Bloom filters for registering their information items, one way of combining their registration information is to calculate the bitwise OR of their individually optimized Bloom filters. This results in a Bloom filter which represents the union of their information items. However, this does not reduce the FPR of the index; instead it has high chances of introducing additional false positives: when checking the individual filters BF_1 and BF_2, there is a set of information items S_1 for which the query of filter BF_1 returns a false positive. There is also such a set S_2 for filter BF_2. Because the OR-ed filter BF_{12} has all bits of filters BF_1 and BF_2 set to 1, querying it for information items of $S_1 \cup S_2$ will still return false positives. Additionally, more false positives may be introduced when $BF_1 \neq BF_2$.

What is more, in order to allow for an OR-ed filter to be formed, filters have to be of the same size m, and they need to use the same number of hash functions k. Both is not generally true, because nodes adjust the parameters of their filters based on their information count.

A better alternative is therefore the following: the idea is to cooperatively form one common optimally configured Bloom filter with a given FPR R_{BF} *within the network*.

One practical mechanism for accomplishing this kind of aggregation is detailed in Section 7.4.1; here, we first examine some theoretical aspects of such an approach. The following theorem claims that this approach is promising.

Theorem 7.3. *Assume that all Bloom filters for registration must have FPR R_{BF}. Further assume that through aggregation, the total number of Bloom filters created by s nodes in the network reduces to $t < s$. Then, the FPR of the search index consisting of the aggregated filters is lower than that of an index containing the individual filters.*

Proof The result follows directly from Equation 7.3 with the observation that the index of aggregated filters contains $t < s$ filters, each with an FPR R_{BF}. □

Note that except for saving header information, this does not reduce the network load because the combined filters are dimensioned based on a fixed R_{BF}.

The downside of aggregation of Bloom filters is that it leads to a certain anonymity of the nodes: if a Bloom filter contains information items of multiple nodes, querying it can only return whether *at least one* of these nodes hosts the respective information item. Nevertheless, if only Bloom filters of nodes which are geographically close together are merged within the network, the contained information can be associated with a geographical region. This provides at least a good idea about the region in which the information item may be found. We will come back to practical mechanisms and tradeoffs in this context in Section 7.4.1.

Aggregation of registration information is also conceivable for THLs, i.e. by concatenating those of individual nodes. The following theorem claims that this (in contrast to aggregation with Bloom filters) does not reduce the FPR of the search index.

Theorem 7.4. *A THL-based index containing s THLs $l_1, l_2, \ldots, l_s$, has the same FPR if l_1 and l_2 are instead concatenated into a single THL $l_{1,2}$.*

Proof The FPR of the concatenated THL $l_{1,2}$ is $R_{1,2} = 1 - (1 - 2^{-c_1})^{n_1}(1 - 2^{-c_2})^{n_2}$.

The FPR of the index with the concatenated THL (i. e., the index with $l_{1,2}, l_3, \ldots, l_s$) is

$$\begin{aligned}\overline{R} &= 1 - (1 - R_{1,2}) \prod_{i=3}^{s} (1 - R_{\mathrm{THL}}(n_i, c_i)) \\ &= 1 - (1 - 2^{-c_1})^{n_1} (1 - 2^{-c_2})^{n_2} \prod_{i=3}^{s} (1 - R_{\mathrm{THL}}(n_i, c_i)) \\ &= 1 - \prod_{i=1}^{s} (1 - R_{\mathrm{THL}}(n_i, c_i)),\end{aligned}$$

which is, according to Equation 7.2, equal to the FPR of the index with the two separate THLs l_1, l_2. □

7.4 Evaluation in a VANET Scenario

In the following, we first outline an aggregation scheme for Bloom filter-based registrations in metropolitan VANETs (Section 7.4.1). Then, we determine the configuration of the transmitted Bloom filters with respect to a given FPR (Section 7.4.2). In Section 7.4.3, we compare the Bloom filter-based index with and without compression to hash-based registration by means of simulative evaluation in the sample scenario.

7.4.1 Opportunistic Local Aggregation of Bloom Filters in Metropolitan VANETs

In a city, larger groups of vehicles wait at intersections because of cross-traffic or traffic lights. This observation is the basis of our opportunistic aggregation scheme. We define a time interval in which vehicles *may* register their information items, bounded by an upper limit after which vehicles *must* register their information items. Whenever a vehicle *must* register its information items, it becomes a coordinator for local aggregation. The coordinator and all vehicles in its radio range that *may* register their information items cooperatively create a common Bloom filter.

The coordination process works as follows: a new coordinator determines the total number of information items to be registered based on its own number and the number of information items that its neighbors *may* register (we assume the latter to be part of the periodic vehicle beacon message). Based on this number, the coordinator computes the size m of the combined Bloom filter. The coordinator broadcasts m, on which

all its neighbors that *may* register information items reply with a Bloom filter of this size, containing their registration information (*atomic Bloom filter*). The coordinator combines the atomic Bloom filters using a bit-wise OR operation, which yields the *combined Bloom filter*. It then registers the combined Bloom filter in the search index.

With aggregation, only the coordinator can be identified as originator of the Bloom filter; the original information sources cannot be directly identified when the index is queried for their information items. To solve this problem, we apply a two-step forwarding of requests from base stations: first, the incoming request is forwarded to the coordinator of the Bloom filter which contains the requested information item. Second, when the coordinator receives the request, it forwards it to the vehicle whose Bloom filter contained the requested information item. For this it is necessary that the coordinator caches all messages sent by its neighbors when creating a common Bloom filter. To avoid the coordinator becoming a single point of failure, all vehicles listen for atomic Bloom filters and forward incoming requests in case they overheard the registration of the information source.

7.4.2 Configuration of Bloom Filters

We first set a limit on the FPR of the search index. Second, we determine the expected number of registrations from simulation. Based on these two numbers, we calculate the FPR R required for a single registration.

It is required that the probabilistic VANET search engine scales well with the number of vehicles participating, i. e. that with large numbers of vehicles, the FPR of the index is still low. We set the goal that in worst case (100 % penetration rate), the FPR of the index may not exceed 5 %.

Figure 3.4 shows the number of vehicles present in the scenario during the time period we considered. From these numbers, we expect a maximum of 2000 vehicles registering their information items. Thus, the search index can be expected to contain 2000 registrations. Nevertheless, we are interested in the case that aggregation as outlined in Section 7.4.1 is employed. The real number of registrations in the index is determined by the effectiveness of the aggregation scheme. A priori, we here assume a reduction of factor 2 for our calculation, leading to an expected registration count of 1000 (the reasonability of this assumption will be verified in Section 7.4.3.3).

If 1000 registrations are expected in the search index, the FPR of a single registration (Bloom filter) needs to be

$$R_{BF} = 1 - 0.95^{\frac{1}{1000}} \approx 0.000051. \tag{7.16}$$

7.4.3 Evaluation of Bloom Filter-based Index

In this section, we evaluate the performance of the proposed Bloom filter-based index in the presented scenario. The respective simulation environment and scenario have been presented in Chapter 3. The underlying index-based search scheme is that described in Section 6.2.1, where the index is distributed among a set of base stations in the VANET area (their locations are defined in Section 4.2.2.3).

It follows an outline of the goals of our evaluation (Section 7.4.3.1). Thereafter, we present the simulation parameters (Section 7.4.3.2) and results (Section 7.4.3.3).

7.4.3.1 Goals

The main goal of our evaluation is to determine whether the applied Bloom filter-based registration saves a significant amount of network load while achieving search success rates comparable to the standard hash-based registration scheme. Additionally, we verify that the FPR of the Bloom filter-based index does not exceed the aforementioned upper bound. Finally, we evaluate the performance of our proposed aggregation scheme, pointing out how many vehicles combine their registrations. We do not investigate the feasibility of transferring large amounts of data. Instead we only assess whether it is possible to transfer any information at all: information sources answer requests by sending only a small packet with empty payload.

7.4.3.2 Simulation parameters

Table 7.1 contains the parameters for evaluation. Regarding the information count per vehicle, a realistic number can hardly be predicted because it depends on the applications using the search index. In [SGG02] and [WCZJ04] the numbers of files per host in Internet peer-to-peer file sharing systems has been investigated. Note that these systems are mainly used for sharing of large audio and video files, which can not be assumed as main content shared through a VANET search engine due to bandwidth restrictions. However, we used the results on Gnutella and Napster from [SGG02] as

Type	Name	Value
Simulation env.		
	Simulation duration	3600 s
	Information count per vehicle	100
	Request interval per vehicle	100 s
	Replication factor	1
Data structures		
	Size of hash values (hash-based index)	64 bits
	FPR of Bloom filter (R)	0.000051
	Hash functions for uncompressed Bloom filter (k)	8
	Hash functions for compressed Bloom filter (k)	2

Table 7.1: Simulation parameters

a rough guideline and decided for an information count of $n = 100$ in the following evaluation. Consequently, each node registered 100 information items.

For computational efficiency of our simulation, we fixed the number of hash functions of uncompressed Bloom filters to 8. As discussed in Section 7.4.2, we strived for an FPR of 5 % in case of 1000 registrations in the index. Thus, according to Equation 7.16, the FPR R_{BF} of a single filter was set to 0.000051. Therefore, according to Equation 7.4, $c = 24$ bits per information were allocated for the uncompressed Bloom filter with $k = 8$. The compressed Bloom filter requires only $c = 17$ bits per information (Equation 7.11). This means that it saves 29 % network load in comparison to the uncompressed Bloom filter. Compared to the standard registration with hash keys of length 64 bit, the compressed Bloom filter has an expected saving of roughly 73 % of network load.

We simulated transmission of both uncompressed and compressed Bloom filters. In case of compressed Bloom filters, vehicles in the simulation did not exchange real compressed filters. The high computational effort to do so is unnecessary, because the compressed size can quite accurately be determined from the entropy, as is verified in Appendix A. To evaluate the performance of compressed Bloom filter-based registration, vehicles sent the uncompressed filters with $k = 8$ and the simulated packet size in the network simulator was altered to reflect the size of a compressed Bloom filter with $k = 2$.

7.4.3.3 Results of Simulation

In this section we present our measured results. Whenever performance of compressed and uncompressed Bloom filters is identical, we omit the results of the compressed filters.

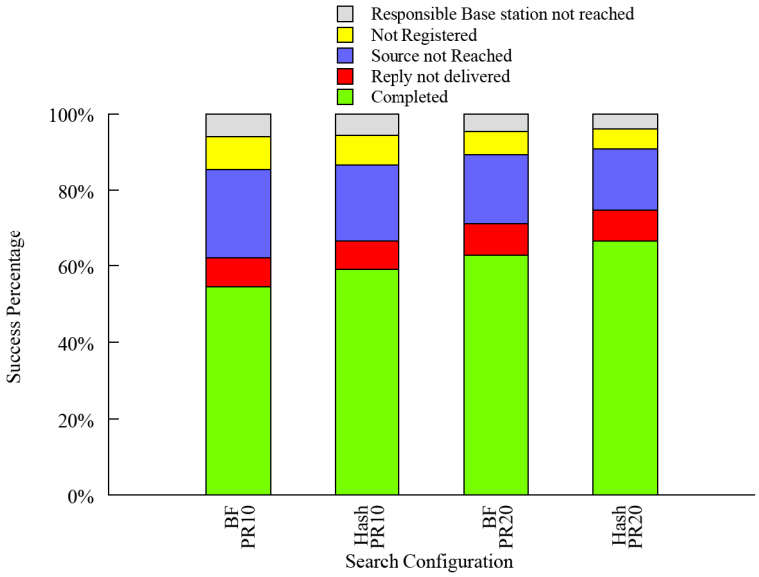


Figure 7.4: Search success rates using Bloom filter-based index (BF) and hash-based index (Hash) at penetration rates 10 % (PR10) and 20 % (PR20).

Figure 7.4 depicts the success rates of search requests based on the Bloom filter-based and on the hash-based index. The hash-based index has an about 2 % better success rate than the Bloom filter-based index. The slightly worse performance of the Bloom filter-based index is a result of the two-step forwarding process of requests that is applied because of aggregated registration: sometimes neither the coordinator nor a node that overheard the coordination process can be contacted and thus the information source cannot be identified. With increasing penetration rate, the consistency of both indices improves, thereby increasing the number of successfully forwarded and answered search requests.

Figure 7.5 shows the total network load under the simulated conditions. The saving of network load caused by index maintenance is clearly visible for the Bloom filter-based registration and matches our expectation of roughly 70 %. Note that for the simulated case of 100 information items per vehicle, network load of probabilistic index maintenance is below beacon load. Network load caused by requests and replies is low, due to the selected request interval of 100 s.

Figure 7.6 shows the effect of our proposed aggregation scheme. While most filters in the index contain the registration of only one information source, the average number of information sources per filter is 2.4 (3.6) at 10 % (20 %) penetration rate. Thus, without aggregation, the index would contain 2.4 (3.6) times more filters, having a negative

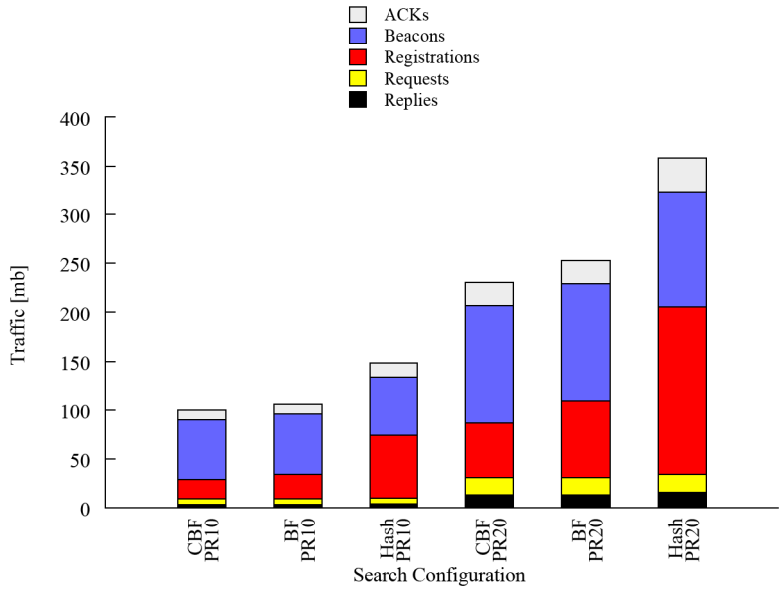


Figure 7.5: Network load using compressed Bloom filter-based index (CBF), uncompressed Bloom filter-based index (BF), and hash-based index (Hash) at penetration rates 10 % (PR10) and 20 % (PR20).

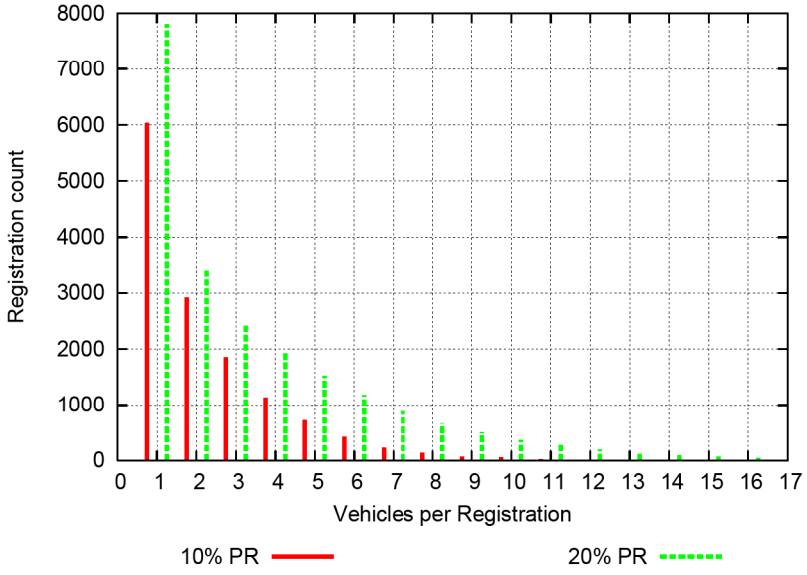


Figure 7.6: Registrations per filter.

impact on the FPR. Naturally, the effect of aggregation can be expected to become more and more significant with higher penetration rate since the number of close-by equipped vehicles increases.

Query result	10 % PR	20 % PR
1 source found in index	6906	15109
> 1 source found in index	55	196
FPR	0.8 %	1.3 %

Table 7.2: False-positive rates in simulation; penetration rates (PR) 10 % and 20 %

Table 7.2 shows the numbers of requests for which one or more than one information source has been discovered in the index, indicating a false positive. The resulting FPRs are far below the 5 % intended in the worst case, because the simulated penetration rates are much lower than assumed for the worst case scenario. Taking into account the increasing effect of aggregation on the number of filters in the index, the results support our confidence that even with full penetration, the FPR will be below the limit.

7.5 Conclusions

The Bloom filter—in its uncompressed and compressed form—has turned out to be a bandwidth-efficient alternative to using standard hash-keys for registration. Aggregation of Bloom filters is a viable means to reduce their total number in the index, and thereby the false-positive rate of index queries. We have confirmed both these findings analytically as well as in a simulation of a VANET scenario: the success rate of search with Bloom filter-based registration is only marginally lower than that with 64-bit hash value-based registration while the network load of registration is reduced by about 70 %. Our results also show that the proposed local opportunistic aggregation is effective in the metropolitan VANET already at low equipment densities. In a more dense VANET, aggregates are formed over more vehicles, thus the outlined scheme can be expected to be even more useful. With aggregation, only the coordinator of each aggregated filter can be identified; our proposed two-step forwarding of requests—first to the coordinator and then to the information source—has turned out to work well in simulation.

Our aggregation scheme may be improved in several ways. First, a common Bloom filter might not only be created among single-hop neighbors but within a larger area. Second, the current scheme assumes uncorrelated information and thus only reflects the worst-case savings: when a coordinator calculates the size of an aggregated Bloom filter

based on the number of information items of its neighbors, it implies that each vehicle owns unique information items. In practice, it is very likely that there are some common items. Thus, if the coordinator knows or estimates this number of common information items, it may set up a smaller filter that suffices for achieving the desired false-positive rate.

7.6 Chapter Summary

In this chapter we have investigated probabilistic data structures as a means for nodes to register their content in a search index, with the goal of causing lower network load with registrations than if nodes send lists of standard hash values. We have first surveyed candidate data structures and have decided for Bloom filters, as they stand out among the others because of their space-efficiency, which is why they have already found many applications in networks. In fact, with compression, Bloom filters can theoretically achieve space-optimality.

In a next step we have defined three types of registrations: uncompressed Bloom filters, compressed Bloom filters and lists of short hash values. We have analytically compared the sizes of registration messages under the condition that checking them for an information item yields the same false-positive rate. The analysis has confirmed the superiority of compressed Bloom filters, and has shown that for each false-positive rate, there is a break-even point in terms of a number of information items from which on the uncompressed Bloom filter is smaller than the list of hash values.

We have identified two additional challenges for queries of Bloom filter-based indices. First, all filters in the index must be checked if all available information sources have to be found, or if the index does not contain the sought-after information item. Second, a queried information item can not be encoded as single hash value in the search request, because Bloom filters applying (a variable number of) k hash functions have to be checked. In this context, we have sketched modified Bloom filter constructions that alleviate both these problems.

Next, we have calculated whether in-network aggregation of Bloom filters or of lists of short hash values reduces the false-positive rate of the search index. The result is that aggregation reduces the false-positive rate in case of Bloom filters, and—except for saving header information—makes no difference in case of hash lists.

Finally, we have proposed an opportunistic local aggregation scheme for metropolitan VANETs, based on the observation that larger groups of vehicles often wait at intersections or in traffic jams. We then have turned to the VANET scenario and have evaluated the feasibility and savings of registration with aggregated, compressed Bloom filters. Compared to 64-bit hash values, the new registration scheme saves 70 % bandwidth in the considered scenario, which is in line with our theoretical expectations.

Chapter 8

Thesis Summary and Future Work

In this thesis we have studied the problem of search in sparse VANETs. As a starting point, we have given an overview of the underlying wireless short-range communication technology and a discussion of expected equipment densities after market introduction of VANETs (Chapter 2). After this, we have introduced the simulation environment and scenario that serve for evaluation of our ideas in thesis (Chapter 3). The simulation environment is a unidirectional coupling of the traffic simulator VISSIM and the network simulator ns-2. The scenario is a four by four kilometer square around the city center of Braunschweig, with a vehicular traffic distribution calibrated to reflect realistic traffic conditions.

In Chapter 4, we have conducted a feasibility study, analytically determining the success rates of three candidate search methods that have emerged in related research areas: geographic hash table, distributed hash table, and flooding. Two elements have been combined to calculate the search latencies of the candidate methods: a stochastic model of their latency, expressed by combinations of random variables, assuming perfect routing, and a simulation study of information propagation speed. In the context of the simulation study, we have defined an aggressive message flooding scheme that achieves optimal latency and success rate as long as congestion on the wireless channel is negligible. Flooding naturally achieves the highest possible success rate among the investigated search methods, thus by calculating its success rate and latency under low network load, we have derived an upper bound. At 5 % and 10 % penetration rate, this upper bound yields a success rate of less than 45 % in the studied scenario, indicating that reliable search requires faster and more reliable communication. Following this finding, we have positioned 19 interlinked base stations at the most frequented intersections and repeated the study, finding that the upper bound of success rate increases to 85 % already at 10 % penetration rate. We have therefore limited our focus on infrastructure-supported

VANETs for the rest of this thesis. As another important result of our feasibility study, we have found that, compared to flooding, the additional communication step with the index-node degrades search success rate and latency (by almost 20 % at 10 % penetration rate in our scenario). Therefore, the existing index-based solutions can not be recommended in VANETs.

A content location service resides on top of the network layer. Therefore, moving from theoretical discussions to implementation and evaluation of a VANET search method, it is essential to define an underlying routing algorithm. We have defined such an algorithm for infrastructure-based VANETs in Chapter 5. As a starting point, we have identified challenges and classified the routing situations in such networks. Upon that, we have presented a position-based routing algorithm that greedily seeks to minimize the beeline distance to the target position, employing local caching in case of local minima, and opportunistically altering the target if in anycast mode. To determine the level of optimality of the algorithm, we have compared its success rate and latency to that of the message flooding scheme introduced in the feasibility study. The success rate of our proposed algorithm is only 2 %–7 % lower than that of the flooding scheme, depending on the routing situation.

Based on the results of the feasibility study, we have then defined an index-based search method in Chapter 6. To keep the cost of communication with the index node low, we have proposed to store the search index at the base stations, as most of the time, messages are routed between vehicles with help of base stations anyway. Consequently, the search performance of such an index is the best case of any distributed index-based search in infrastructure-supported VANETs. To make their local information available for search, vehicles periodically send registration messages containing hash-keys of their locally held information items to the search index. To have a reference method as comparison, we have extended the aggressive flooding scheme with message suppression mechanisms for better bandwidth-efficiency. For each search method, we have determined parameters that give close-to-optimal performance and have then evaluated the methods head-to-head. While the bandwidth-efficient flooding scheme achieves a success rate that is about 10 %–15 % higher than that of the index-based scheme, the network load caused by flooded requests is by far (at least factor 4) greater than that of the index-based method. With increasing availability of information items, and with increasing penetration rate, the success rate of the index-based scheme approaches that of the flooding scheme, making the index-based scheme the better solution. Compared to the theoretical results of Chapter 4, the implemented schemes perform slightly worse. This has two reasons: first, the assumption of independent deliveries causes a slight

overestimation of the success rate in the feasibility study; second, search and content request messages in index-based search as well as content retrieval messages in both studied schemes are not delivered with help of a perfect routing algorithm but through a realistic routing scheme.

When implementing a search index in a network as dynamic as a VANET, a non-negligible part of network traffic is caused by index maintenance. To reduce this traffic, we have proposed a probabilistic index in Chapter 7 of this thesis. The key idea is that vehicles periodically register their information items encoded into compressed Bloom filters instead of hash keys. Like with shorter hash keys, reducing the length of a compressed Bloom filter increases the risk of false positives in membership queries, but our analysis has shown that compressed Bloom filters are more space-efficient than lists of hash keys—in fact, they are space-optimal. We further have calculated that, from a certain number of contained information items that depends on the desired false-positive rate, even uncompressed Bloom filters are more efficient than lists of hash keys. Additionally, we have found that the false-positive rate of a search index consisting of Bloom filters can be further reduced when network nodes form a common Bloom filter instead of registering individually. Motivated by this theoretical finding, we have proposed an aggregation scheme for VANETs, where vicinal nodes opportunistically create a common filter. We then have conducted a simulation study where nodes commonly register compressed Bloom filters, using the proposed aggregation scheme. Compared to registration with 64-bit hash keys, the new scheme requires only 30 % of the bandwidth for index maintenance, at a false-positive rate of less than 5 %. Thereby, our aggregation scheme helps to reduce the filter count in the index by more than 50 %.

There is ample opportunity of further research around the topic of content location and querying. First of all, it would be interesting to evaluate our proposed ideas under different vehicular traffic conditions and different infrastructure configurations. Besides that, to our knowledge there are no specific answers to the question when content should be requested on-demand (pull-based communication) and when it should be distributed proactively (push-based communication). Related to this, the controlled proactive replication of content in VANETs could be investigated to improve search success rate and latency. Regarding the space-efficiency of different index types, registrations in Bloom filter-based indices can be encoded more compactly than in a DHT, but network load of queries is increased. The question remains whether there is another data structure offering both more space-efficient registration and queries. Furthermore, our aggregation scheme may be improved so that a common Bloom filter is formed over nodes in a larger area (e.g. over a whole VANET partition with end-to-end connectivity). Besides that,

coordinators of aggregation might take into account the correlation between information items in their vicinity, allowing for a smaller filter size and thus greater bandwidth savings than if presuming uncorrelated information.

Concluding, we have studied the search for content in sparse VANETs. The methodology used in our feasibility study is a means to estimate the influence of communication conditions to solutions of the search problem, and may of course be used for other applications with similar communication properties. The implementation and parameter study of the search schemes has shown the results achievable in practice and has underlined the conditions under which index-based search is the better alternative than flooding. Finally, for situations where index-based search is applicable, our proposal and analysis of probabilistic encodings of registration information has shown the potential of bandwidth savings in index creation and maintenance.

Appendix A

Compressing Bloom Filters Near Their Entropy Limit

For the example case of $k = 2$ hash functions, we verify here that compression of Bloom filters close to the entropy limit is possible through arithmetic coding. We set $n = 100$ information items and select an FPR of $R = 0.000051$ (as used in Section 7.4.3). This yields a size of $m = 27906$ bits for the uncompressed filter. For these values, (7.11) indicates a compressed size of approximately 1707 bits.

To verify this, we created 1000 filters, making use of an existing implementation of Bloom filters[1]. The filters were then compressed using a freely available implementation of arithmetic coding[2] to see how close we come to the entropy limit. We repeated the experiment ten times.

The minimum and maximum of the EDFs of compressed filter sizes over all runs are shown in Figure A.1. None of the Bloom filters had a compressed size of more than 1724 bits, the average size was 1718 bits. Note that the fact that there are some filters with compressed size of less than 1707 bits is normal; for an optimal compression the *average* size is equal to the entropy. The results observed here come very close to the theoretical optimum and confirm our theoretical expectations.

[1] http://www.partow.net/programming/hashfunctions/index.html
[2] http://www.db.toronto.edu/radford/ac.software.html

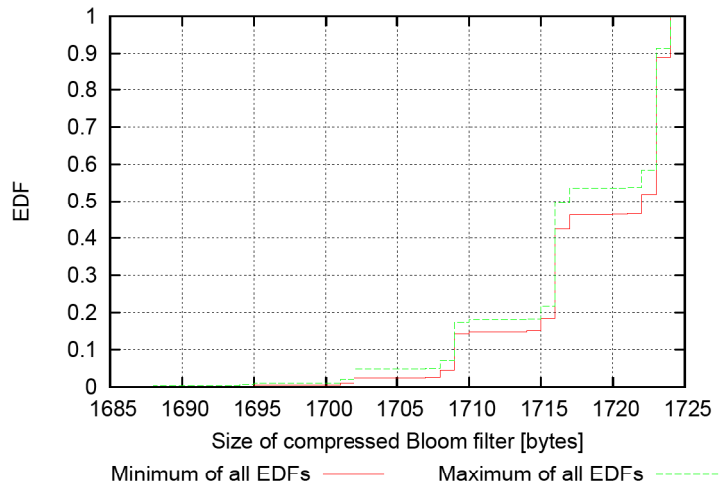


Figure A.1: EDF: compressed Bloom filter sizes (minimum and maximum over all ten experiment runs)

Bibliography

Own Publications

[LSW+08] Christian Lochert, Björn Scheuermann, Christian Wewetzer, Andreas Lübke, and Martin Mauve. Data Aggregation and Roadside Unit Placement for a VANET Traffic Information System. In *VANET 2008: Proceedings of the Fifth ACM International Workshop on Vehicular Ad-Hoc Networks*, September 2008.

[WCLM07] Christian Wewetzer, Murat Caliskan, Andreas Lübke, and Martin Mauve. The Feasibility of a Search Engine for Metropolitan Vehicular Ad-Hoc Networks. In *AutoNet 2007: Proceedings of the 2nd IEEE Workshop on Automotive Networking and Applications*, November 2007.

[WCLM08] Christian Wewetzer, Murat Caliskan, Andreas Lübke, and Martin Mauve. A Search Engine for Vehicular Ad-Hoc Networks. In *V2VCOM 2008: Proceedings of the 4th IEEE Workshop on Vehicle to Vehicle Communications*, June 2008.

[WCML07] Christian Wewetzer, Murat Caliskan, Klaus Meier, and Andreas Lübke. Experimental Evaluation of UMTS and Wireless LAN for Inter-Vehicle Communication. In *ITST 2007: Proceedings of the 7th International Conference on ITS Telecommunications*, pages 287–292, June 2007.

[WHWL08] Axel Wegener, Horst Hellbrück, Christian Wewetzer, and Andreas Lübke. VANET Simulation Environment with Feedback Loop and its Application to Traffic Light Assistance. In *AutoNet 2008: Proceedings of the 3rd IEEE Workshop on Automotive Networking and Applications*, December 2008.

[WSLM] Christian Wewetzer, Björn Scheuermann, Andreas Lübke, and Martin Mauve. Content Registration in VANETs - Saving Bandwidth through Node Cooperation. Under submission.

Other References

[AH05] Hamada Alshaer and Eric Horlait. An optimized adaptive broadcast scheme for inter-vehicle communication. In *Vehicular Technology Conference Spring 2005*, Stockholm, Sweden, May 2005.

[AK74] O. Amble and Donald E. Knuth. Ordered hash tables. *Computer Journal*, 17(2):135–142, 1974.

[AL06] J. Abley and K. Lindqvist. Operation of Anycast Services. RFC 4786 (Best Current Practice), December 2006.

[ALG+05] J. Anda, J. LeBrun, D. Ghosal, C-N. Chuah, and H. M. Zhang. Vgrid: Vehicular ad hoc networking and computing grid for intelligent traffic control. In *IEEE Vehicular Technology Conference*, May 2005.

[BA05] Nabhendra Bisnik and Alhussein Abouzeid. Modeling and analysis of random walk search algorithms in p2p networks. In *HOT-P2P '05: Proceedings of the Second International Workshop on Hot Topics in Peer-to-Peer Systems*, pages 95–103, Washington, DC, USA, 2005. IEEE Computer Society.

[BABM05] John Bicket, Daniel Aguayo, Sanjit Biswas, and Robert Morris. Architecture and evaluation of an unplanned 802.11b mesh network. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 31–42, New York, NY, USA, 2005. ACM.

[BHM07] Rainer Baumann, Simon Heimlicher, and Martin May. Towards realistic mobility models for vehicular ad-hoc networks. In *Proceedings of the 26th Annual IEEE Conference on Computer Communications*, Anchorage, AK, Alaska, May 2007.

[BL85] Chris Buckley and Alan F. Lewit. Optimization of inverted vector searches. In *SIGIR '85: Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 97–110, New York, NY, USA, 1985. ACM.

[Blo70] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970.

[BM77] Robert S. Boyer and J. Strother Moore. A fast string searching algorithm. *Communications of the ACM*, 20(10):62–72, 1977.

[BM03] Andrei Z. Broder and Michael Mitzenmacher. Survey: Network applications of bloom filters: A survey. *Internet Mathematics*, 1(4), 2003.

[BM09] James Bernsen and D. Manivannan. Review: Unicast routing protocols for vehicular ad hoc networks: A critical comparison and classification. *Pervasive and Mobile Computing*, 5(1):1–18, 2009.

[BV05] Paolo Boldi and Sebastiano Vigna. Mutable strings in java: design, implementation and lightweight text-search algorithms. *Science of Computer Programming*, 54(1):3–23, 2005.

[CBD02] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2:483–502, 2002.

[CDA08] T. Clausen, C. Dearlove, and B. Adamson. Jitter Considerations in Mobile Ad Hoc Networks (MANETs). RFC 5148 (Informational), February 2008.

[CDK05] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems: Concepts and Design (4th Edition) (International Computer Science)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[CF05] Curt Cramer and Thomas Fuhrmann. Proximity Neighbor Selection for a DHT in Wireless Multi-Hop Networks. In *P2P '05: Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing (P2P'05)*, pages 3–10, Washington, DC, USA, 2005. IEEE Computer Society.

[CGG+06] Carla-Fabiana Chiasserini, Rossano Gaeta, Michele Garetto, Marco Gribaudo, and Matteo Sereno. Efficient broadcasting of safety messages in multihop vehicular networks. In *Proceedings of 20th International Parallel and Distributed Processing Symposium (IPDPS 06)*. IEEE, 2006.

[CGM06] Murat Caliskan, Daniel Graupner, and Martin Mauve. Decentralized Discovery of Free Parking Places. In *Proceedings of the Third ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2006)*, pages 30–39, September 2006.

[CHM97] Zbigniew J. Czech, George Havas, and Bohdan S. Majewski. Perfect hashing. *Theoretical Computer Science*, 182(1-2):1–143, 1997.

[CKR+04] Bernard Chazelle, Joe Kilian, Ronitt Rubinfeld, Ayellet Tal, and Oh Boy. The bloomier filter: an efficient data structure for static support lookup tables. In *In Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 30–39, 2004.

[CM03] Saar Cohen and Yossi Matias. Spectral bloom filters. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 241–252, New York, NY, USA, 2003. ACM.

[Con07] Car-2-Car Communication Consortium. Car-2-Car Communication Consortium Manifesto V1.1: Overview of the C2C-CC system. Technical report, Car-2-Car Communication Consortium, August 2007.

[CSLSC97] IEEE Computer Society LAN/MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.* The Institute of Electrical and Electronic Engineers, New York, New York, 1997. IEEE Std. 802.11–1997.

[CSLSC98] IEEE Computer Society LAN/MAN Standards Committee. *Standard for Information Technology—Telecommunications and information exchange between systems—Local and Metropolitan Area Networks—Specific Requirements; Part 2: Logical Link Control.* The Institute of Electrical and Electronic Engineers, New York, New York, 1998. IEEE Std. 802.2.

[CSLSC99a] IEEE Computer Society LAN/MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High-speed Physical Layer in the 5 GHz Band.* The Institute of Electrical and Electronic Engineers, New York, New York, 1999. IEEE Std. 802.11a–1999.

[CSLSC99b] IEEE Computer Society LAN/MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band.* The Institute of Electrical and Electronic Engineers, New York, New York, 1999. IEEE Std. 802.11b–1999.

[CSLSC03] IEEE Computer Society LAN/MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Amendment 4: Further Higher Data Rate Extension in the 2.4 GHz Band.* The Institute of Electrical and Electronic Engineers, New York, New York, 2003. IEEE Std. 802.11g–2003.

[CSLSC05] IEEE Computer Society LAN/MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements.* The Institute of Electrical and Electronic Engineers, New York, New York, 2005. IEEE Std. 802.11e–2005.

[CSLSC06] IEEE Computer Society LAN/MAN Standards Committee. *Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Multichannel Operation.* The Institute of Electrical and Electronic Engineers, New York, New York, 2006. IEEE Std. P1609.4.

[CSLSC07a] IEEE Computer Society LAN/MAN Standards Committee. *Trial Use Standard for Wireless Access in Vehicular Environments (WAVE) - Architecture.* The Institute of Electrical and Electronic Engineers, New York, New York, 2007. IEEE Std. P1609.0.

[CSLSC07b] IEEE Computer Society LAN/MAN Standards Committee. *Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Networking Services*. The Institute of Electrical and Electronic Engineers, New York, New York, 2007. IEEE Std. P1609.3.

[CSLSC08] IEEE Computer Society LAN/MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Draft Amendment for Wireless Access in Vehicular Networks*. The Institute of Electrical and Electronic Engineers, New York, New York, 2008. IEEE Std. 802.11p–2008.

[CVCAC06] J. Chennikara-Varghese, Wai Chen, O. Altintas, and Shengwei Cai. Survey of routing protocols for inter-vehicle communications. *Annual International Conference on Mobile and Ubiquitous Systems*, 0:1–5, 2006.

[DM04] Peter C. Dillinger and Panagiotis Manolios. Bloom filters in probabilistic verification. In Alan J. Hu and Andrew K. Martin, editors, *Proceedings of 5th International Conference on Formal Methods in Computer-Aided Design (FMCAD 04)*, volume 3312 of *Lecture Notes in Computer Science*, pages 367–381. Springer, 2004.

[DSR07] DSRC committee of the SAE. Draft SAE J2735 Dedicated Short Range Communications (DSRC) Message Set Dictionary, August 2007.

[EGH+06] Tamer ElBatt, Siddhartha K. Goel, Gavin Holland, Hariharan Krishnan, and Jayendra Parikh. Cooperative collision warning using dedicated short range wireless communications. In *VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, pages 1–9, New York, NY, USA, 2006. ACM.

[EJ01] D. Eastlake 3rd and P. Jones. US Secure Hash Algorithm 1 (SHA1). RFC 3174 (Informational), September 2001.

[FCAB00] Li Fan, Pei Cao, Jussara M. Almeida, and Andrei Z. Broder. Summary cache: a scalable wide-area web cache sharing protocol. *IEEE/ACM Transactions on Networking*, 8(3):281–293, 2000.

[FCC05] Marco Fiore, Claudio Casetti, and Carla-Fabiana Chiasserini. On-demand content delivery in vehicular wireless networks. In *MSWiM '05: Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 87–94, New York, NY, USA, 2005. ACM.

[FHM+04] Holger Fuessler, Hannes Hartenstein, Martin Mauve, Wolfgang Effelsberg, and Joerg Widmer. Contention-based forwarding for street scenarios. In *1st International Workshop in Intelligent Transportation (WIT 2004)*, 2004.

[Fio06] Marco Fiore. Mobility models in inter-vehicle communications literature. Technical report, Politecnico di Torino, 2006.

[FK84] M. L. Fredman and J. Komlòs. On the size of separating systems and families of perfect hash functions. *SIAM Journal of Algebraic Discrete Methods*, 5(1):61–68, March 1984.

[FK06] Roy Friedman and Gabriel Kliot. Location services in wireless ad hoc and hybrid networks: A survey. Technical report, Department of Computer Science, Technion, Haifa, Israel, October 2006.

[FKM+03] Holger Fuessler, Michael Kaesemann, Martin Mauve, Hannes Hartenstein, and Joerg Widmer. Contention-based forwarding for mobile ad-hoc networks. *Elsevier's Ad Hoc Networks*, 1(4):351 – 369, 2003.

[FPSS05] Dimitris Fotakis, Rasmus Pagh, Peter Sanders, and Paul G. Spirakis. Space efficient hash tables with worst case constant access time. *Theory of Computing Systems*, 38(2):229–248, 2005.

[FRWZ07] Elena Fasolo, Michele Rossi, Jörg Widmer, and Michele Zorzi. In-network aggregation techniques for wireless sensor networks: A survey. *IEEE Communication Magazine*, April 2007.

[GZ06] Anandha Gopalan and Taieb Znati. V2V-Net: A Vehicle-to-Vehicle Overlay Structure for Service Deployment in Vehicular Ad Hoc Networks. Technical report, Dept. of Computer Science, University of Pittsburgh, 2006.

[HABR05] Khaled A. Harras, Kevin C. Almeroth, and Elizabeth M. Belding-Royer. Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding in Sparse Mobile Networks. In Raouf Boutaba, Kevin C. Almeroth, Ramón Puigjaner, Sherman X. Shen, and James P. Black, editors, *NETWORKING*, volume 3462 of *Lecture Notes in Computer Science*, pages 1180–1192. Springer, 2005.

[HBS01] *Handbuch für die Bemessung von Straßenverkehrsanlagen (HBS)*. Forschungsgesellschaft für Straßen- und Verkehrswesen, Cologne, Germany, 2001.

[HHCW05] Elgan Huang, Wenjun Hu, Jon Crowcroft, and Ian Wassell. Towards commercial mobile ad hoc network applications: a radio dispatch system. In *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 355–365, New York, NY, USA, 2005. ACM.

[ITU94] ITU-T. Recommendation X.200: Information technology — open systems interconnection — basic reference model: The basic model, 1994.

[IW07] Khaled Ibrahim and Michele C. Weigle. Accurate data aggregation for vanets. In *VANET '07: Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*, pages 71–72, New York, NY, USA, 2007. ACM.

[JFP04] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 145–158, New York, NY, USA, 2004. ACM.

[JHP+03] Jorjeta G. Jetcheva, Yih-Chun Hu, Santashil Palchaudhuri, Amit Kumar Saha, and David B. Johnson. Design and evaluation of a metropolitan area multitier wireless ad hoc network architecture. *IEEE Workshop on Mobile Computing Systems and Applications*, 0:32, 2003.

[JM96] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[KEO06] G. Korkmaz, E. Ekici, and F. Ozguner. An efficient fully ad-hoc multi-hop broadcast protocol for inter-vehicular communication systems. *Proceesings of IEEE International Conference on Communications (ICC '06)*, 1:423–428, June 2006.

[KEOUO04] Gökhan Korkmaz, Eylem Ekici, Füsun Özgüner, and Ümit Özgüner. Urban multi-hop broadcast protocol for inter-vehicle communication systems. In *VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 76–85, New York, NY, USA, 2004. ACM.

[KK00] Brad Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 243–254, New York, NY, USA, 2000. ACM Press.

[KM08] Adam Kirsch and Michael Mitzenmacher. Less hashing, same performance: Building a better bloom filter. *Random Structures & Algorithms*, 33(2):187–218, 2008.

[KT75] L. Kleinrock and F. Tobagi. Packet switching in radio channels: Part i–carrier sense multiple-access modes and their throughput-delay characteristics. *IEEE Transactions on Communications [legacy, pre - 1988]*, 23(12):1400–1416, December 1975.

[LCC+02] Qin Lv, Pei Cao, Edith Cohen, Kai Li, and Scott Shenker. Search and replication in unstructured peer-to-peer networks. In *SIGMETRICS '02: Proceedings of the 2002 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 258–259, New York, NY, USA, 2002. ACM.

[LCS+05] Christian Lochert, Murat Caliskan, Björn Scheuermann, Andreas Barthels, Alfonso Cervantes, and Martin Mauve. Multiple Simulator Interlinking Environment for Inter Vehicle Communication. In *VANET 2005: Proceedings of the Second ACM International Workshop on Vehicular Ad Hoc Networks*, pages 87–88, September 2005.

[LH04] Jun Luo and Jean-Pierre Hubaux. A Survey of Inter-Vehicle Communication. Technical report, EPFL Lausanne, 2004.

[LH07] F. Liu and G. J. Heijenk. Context discovery using attenuated bloom filters in ad-hoc networks. *Journal of Internet Engineering*, 1(1):49–58, 2007.

[LHLG07] Kevin C. Lee, Jerome Haerri, Uichin Lee, and Mario Gerla. Enhanced Perimeter Routing for Geographic Forwarding Protocols in Urban Vehicular Scenarios. In *AutoNet 2007: Proceedings of the 2nd IEEE Workshop on Automotive Networking and Application*, November 2007.

[LHT+03] Christian Lochert, Hannes Hartenstein, Jing Tian, Dagmar Herrmann, Holger Füßler, and Martin Mauve. A Routing Strategy for Vehicular Ad Hoc Networks in City Environments. In *IV 2003: Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 156–161, June 2003.

[LMFH05] Christian Lochert, Martin Mauve, Holger Füßler, and Hannes Hartenstein. Geographic Routing in City Scenarios. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 9(1):69–72, January 2005.

[LMP06] Vincent Lenders, Martin May, and Bernhard Plattner. Density-based vs. proximity-based anycast routing for mobile networks. In *IEEE INFOCOM 2006*, Barcelona, Spain, April 2006.

[LPAG06] Uichin Lee, Joon-Sang Park, Eyal Amir, and Mario Gerla. FleaNet: A Virtual Market Place on Vehicular Networks. In *V2VCOM'06*, San Jose, CA, USA, July 2006.

[LPY+06] Uichin Lee, Joon-Sang Park, Joseph Yeh, Giovanni Pau, and Mario Gerla. CodeTorrent: Content Distribution using Network Coding in VANETs. In *MobiShare'06*, Los Angeles, CA, USA, September 2006.

[LSA01] E. Limpert, W. A. Stahel, and M. Abbt. Log-normal distributions across the sciences: Keys and clues. *BioScience*, 51(5):341–352, May 2001.

[LSCM07] Christian Lochert, Björn Scheuermann, Murat Caliskan, and Martin Mauve. The Feasibility of Information Dissemination in Vehicular Ad-Hoc Networks. In *WONS 2007: Fourth Annual Conference on Wireless On demand Network Systems and Services*, pages 92–99, January 2007.

[LSM07] Christian Lochert, Björn Scheuermann, and Martin Mauve. Probabilistic aggregation for data dissemination in VANETs. In *VANET '07: Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*, pages 1–8, New York, NY, USA, 2007. ACM.

[LW04] Xiuqi Li and Jie Wu. Searching techniques in peer-to-peer networks. In *Handbook of Theoretical and Algorithmic Aspects of Ad Hoc, Sensor, and Peer-to-Peer Networks*. CRC Press, 2004.

[LW07] Fan Li and Yu Wang. Routing in vehicular ad hoc networks: A survey. In Imielinski and Korth, editors, *IEEE Vehicular Technology Magazine*, volume 2, pages 12–22. IEEE, June 2007.

[Mat05] Irina Matschke. The impact of dynamic navigation on the travel times in urban networks. In *Proceedings of the 10th EURO Working Group on Transportation*, Poznan, Poland, September 2005.

[MES04] Christian Maihöfer, Reinhold Eberhardt, and Elmar Schoch. CGGC: Cached greedy geocast. In Peter Langendörfer, Mingyan Liu, Ibrahim Matta, and Vassilios Tsaoussidis, editors, *Conference on Wired/Wireless Internet Communication (WWIC 2004)*, volume 2957 of *Lecture Notes in Computer Science*, Frankfurt (Oder), Germany, February 2004. Springer.

[Mit01] Michael Mitzenmacher. Compressed bloom filters. In *PODC '01: Proceedings of the twentieth annual ACM symposium on Principles of distributed computing*, pages 144–150, New York, NY, USA, 2001. ACM.

[MKKB01] Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In *ACM SIGCOMM 2001*, San Diego, CA, USA, September 2001.

[MMP+05] K. Matheus, R. Morich, I. Paulus, C. Menig, A. Lübke, B. Rech, and W. Specks. Car-to-Car Communication - Market Introduction and Success Factors. In *Proceedings of ITS 2005: 5th European Congress and Exhibition on Intelligent Transport Systems and Services*, Hannover, Germany, 2005.

[MPGW06] A. Mahajan, N. Potnis, K. Gopalan, and A. Wang. Urban mobility models for vanets. In *Proc. of 2nd Workshop on Next Generation Wireless Networks*, 2006.

[MWH01] Martin Mauve, Jörg Widmer, and Hannes Hartenstein. A Survey on Position-Based Routing in Mobile Ad-Hoc Networks. *IEEE Network Magazine*, 15(6):30–39, November 2001.

[NDLI04] Tamer Nadeem, Sasan Dashtinezhad, Chunyuan Liao, and Liviu Iftode. Trafficview: traffic data dissemination using car-to-car communication. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 8(3):6–19, 2004.

[NDP+05] Alok Nandan, Shirshanka Das, Giovanni Pau, Mario Gerla, and M. Y. Sanadidi. Co-operative downloading in vehicular ad-hoc wireless networks. In *WONS '05: Proceedings of the Second Annual Conference on Wireless On-demand Network Systems and Services*, pages 32–41, Washington, DC, USA, 2005. IEEE Computer Society.

[NDZ+05] Alok Nandan, Shirshanka Das, Biao Zhou, Giovanni Pau, and Mario Gerla. Adtorrent: Digital billboards for vehicular networks. In *Proceedings of IEEE/ACM International Workshop on Vehicle-to-Vehicle Communications (V2VCOM)*, 2005.

[PBMW99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Manuscript, 1999.

[PDH04] Himabindu Pucha, Saumitra M. Das, and Y. Charlie Hu. Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks. In *WMCSA '04: Proceedings of the Sixth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'04)*, pages 163–173, Washington, DC, USA, 2004. IEEE Computer Society.

[PM99a] V. Park and J. Macker. Anycast routing for mobile services. In *Proceedings of the Conference on Information Sciences and Systems, CISS '99*, January 1999.

[PM99b] V. D. Park and J. P. Macker. Anycast routing for mobile networking. In *Proceedings of the IEEE Military Communications Conference, MILCOM 1999*, volume 1, pages 1–5, October November 1999.

[PMM93] C. Partridge, T. Mendez, and W. Milliken. Host Anycasting Service. RFC 1546 (Informational), November 1993.

[PR99] Charles E. Perkins and Elizabeth M. Royer. Ad-hoc On-Demand Distance Vector Routing. *2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, 00:90, 1999.

[PSS07] Felix Putze, Peter Sanders, and Johannes Singler. Cache-, hash- and space-efficient bloom filters. In Camil Demetrescu, editor, *6th International Workshop on Experimental Algorithms (WEA 2007)*, volume 4525 of *Lecture Notes in Computer Science*, pages 108–121. Springer, 2007.

[RD01] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems". In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, November 2001.

[RFH+01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Schenker. A scalable content-addressable network. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, volume 31, pages 161–172. ACM Press, October 2001.

[Rit01] Jordan Ritter. Why Gnutella can't scale. No, really., 2001. `http://www.darkridge.com/~jpr5/doc/gnutella.html`.

[Riv92] R. Rivest. The MD5 Message-Digest Algorithm. RFC 1321 (Informational), April 1992.

[RKY+02] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: A Geographic Hash Table for Data-Centric Storage in SensorNets. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, September 2002.

[SD96] Ulrich Stern and David L. Dill. A new scheme for memory-efficient probabilistic verification. In *Proceedings of IFIP TC6/WG6.1 Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing, and Verification*, pages 333–348, 1996.

[SGG02] Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, January 2002.

[SI05] Francoise Sailhan and Valerie Issarny. Scalable service discovery for manet. In *PERCOM '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications*, pages 235–244, Washington, DC, USA, 2005. IEEE Computer Society.

[SJ04] Amit Kumar Saha and David B. Johnson. Modeling mobility for vehicular ad-hoc networks. In *VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 91–92, New York, NY, USA, 2004. ACM.

[SSS00] Murray R. Spiegel, John J. Schiller, and Alu R. Srinivasan. *Schaum's Outline of Probability and Statistics*. McGraw-Hill, March 2000.

[TM07] Marc Torrent-Moreno. *Inter-Vehicle Communications: Achieving Safety in a Distributed Wireless Environment: Challenges, Systems and Protocols*. PhD thesis, University of Karlsruhe, Germany, 2007.

[TNCS02] Yu-Chee Tseng, Sze-Yao Ni, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. *Wireless Networks*, 8(2/3):153–167, 2002.

[TR03] Dimitrios Tsoumakos and Nick Roussopoulos. A comparison of peer-to-peer search methods. In Vassilis Christophides and Juliana Freire, editors, *WebDB*, pages 61–66, 2003.

[TSM07] Thi Minh Chau Tran, Björn Scheuermann, and Martin Mauve. Detecting the presence of nodes in manets. In *CHANTS '07: Proceedings of the second workshop on Challenged networks CHANTS*, pages 43–50, New York, NY, USA, 2007. ACM.

[VB00] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report, 2000.

[WBM+07] Nawaporn Wisitpongphan, Fan Bai, Priyantha Mudalige, Varsha K. Sadekar, and Ozan K. Tonguz. Routing in sparse vehicular ad hoc wireless networks. *IEEE Journal on Selected Areas in Communications*, 25(8):1538–1556, 2007.

[WCZJ04] Wenjie Wang, Hyunseok Chang, Amgad Zeitoun, and Sugih Jamin. Characterizing guarded hosts in peer-to-peer file sharing systems. In *In Proceedings of IEEE Global Communications Conference*, Global Internet and Next Generation Networks, 2004.

[Wei] E. W. Weisstein. Square line picking. from Math World– A Wolfram Web Resource, http://mathworld.wolfram.com/ SquareLinePicking.html.

[WER+03] L. Wischhof, A. Ebner, H. Rohling, M. Lott, and R. Halfmann. SOTIS - A Self-Organizing Traffic Information System. Proceedings of the 57th IEEE Vehicular Technology Conference (VTC '03 Spring), Jeju, Korea, 2003.

[Wie74] R. Wiedemann. Simulation des Straßenverkehrsflusses (in German language). *Schriftenreihe des Instituts für Verkehrswesen der Universität Karlsruhe*, Heft 8, 1974.

[WL93] Pierre Wolper and Denis Leroy. Reliable hashing without collosion detection. In *CAV '93: Proceedings of the 5th International Conference on Computer Aided Verification*, pages 59–70, London, UK, 1993. Springer-Verlag.

[WNC87] Ian H. Witten, Radford M. Neil, and John G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, June 1987.

[WTP+06] 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 Magazine*, 2006.

[WZJ03a] Jainxin Wang, Yuan Zheng, and Weijia Jia. A-DSR: A DSR-based Anycast Protocol for IPv6 Flow in Mobile Ad Hoc Networks. In *Proceedings of the IEEE Vehicular Technology Conference*, Orlando, FL, USA, October 2003.

[WZJ03b] Jainxin Wang, Yuan Zheng, and Weijia Jia. An AODV-based Anycast Protocol in Mobile Ad Hoc Network. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communication*, Beijing, China, September 2003.

[XGB03] Kaixin Xu, Mario Gerla, and Sang Bae. Effectiveness of RTS/CTS handshake in IEEE 802.11 based ad hoc networks. *Elsevier Ad Hoc Networks*, 1(1):107–123, 2003.

[XS01] S. Xu and T. Saadawi. Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks? *IEEE Communications Magazine*, 39(6):130–137, Jun 2001.

[YGM02] Beverly Yang and Hector Garcia-Molina. Improving search in peer-to-peer networks. In *ICDCS '02: Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, Washington, DC, USA, 2002. IEEE Computer Society.

[ZC06] J. Zhao and G. Cao. VADD: Vehicle-Assisted Data Delivery in Vehicular Ad-hoc Networks. In *Proceedings of the 25th Conference on Computer Communications (INFOCOM06)*, 2006.

[ZPM04] Andrea Zanella, Gianfranco Pierobon, and Simone Merlin. On the limiting performance of broadcast algorithms over unidimensional ad-hoc radio networks. In *Proceedings of 7th International Symposium on Wireless Personal Multimedia Communications (WPMC04)*, Padova, Italy, 2004.

[ZS05] Thomas Zahn and Jochen Schiller. MADPastry: A DHT Substrate for Practicably Sized MANETs. In *Proc. of 5th Workshop on Applications and Services in Wireless Networks (ASWN2005)*, Paris, France, June 2005.

[ZS06] Thomas Zahn and Jochen Schiller. DHT-based Unicast for Mobile Ad Hoc Networks. In *PERCOMW '06: Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops*, pages 179–183, Washington, DC, USA, 2006. IEEE Computer Society.

Index

W

Die hier vorgelegte Dissertation habe ich eigenständig und ohne unerlaubte Hilfe angefertigt. Die Dissertation wurde in der vorgelegten oder in ähnlicher Form noch bei keiner anderen Institution eingereicht. Ich habe bisher keine erfolglosen Promotionsversuche unternommen.

Düsseldorf, den 25.3.2009

Christian Wewetzer