Position-Based Packet Forwarding for Vehicular Ad-Hoc Networks

Inauguraldissertation
zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften
der Universität Mannheim

vorgelegt von

Dipl. Wirtsch.–Inf. Holger Martin Füßler
aus Stutensee

Mannheim, 2007

Dekan:	Professor Dr. Matthias Krause, Universität Mannheim
Referent:	Professor Dr. Wolfgang Effelsberg, Universität Mannheim
Korreferentin:	Professor Dr. Martina Zitterbart, Universität Karlsruhe

Tag der mündlichen Prüfung: 30. April 2007

Vorwort (Preface)

> Was Du ererbt von Deinen Vätern,
> erwirb es, um es zu besitzen.
>
> *(Johann Wolfgang von Goethe)*

Bis zu meiner Diplomarbeit hatte ich mich nie mit wissenschaftlicher Arbeit beschäftigt. Ich bin nie Hiwi gewesen, noch hatte ich irgendeine Ahnung, wie ein Lehrstuhl, sei es im Hinblick auf Lehre oder im Hinblick auf Forschung, funktioniert. Damit ist klar, dass ich alles, was ich heute kann, weiß, und bin, meinen geistigen Vätern verdanke.

Der in mehrerlei Hinsicht wichtigste Vater dieser Dissertation ist mein Doktor-Vater Professor Doktor Wolfgang Effelsberg. Ohne ihn wäre ich zum einen nicht auf die Idee gekommen, nach dem Diplom eine Promotion zu versuchen, und zum anderen wäre dieses Werk sicher nicht das, was es geworden ist. Auch bedankt sei an dieser Stelle meine Korreferentin, Frau Professor Doktor Martina Zitterbart von der Universität Karlsruhe, die trotz immenser zeitlicher Belastung zugestimmt hat, meiner Dissertation Pate zu stehen.

In fachlicher Hinsicht gilt mein Dank in erster Linie der bei meiner Ankunft frisch geschmiedeten Ad-Hoc Netzwerk Gruppe am Lehrstuhl Effelsberg. Hier sind ganz besonders Doktor Martin Mauve und Doktor Hannes Hartenstein (damals `NEC` Network Labs Europe), beides heute ordentliche Professoren, zu nennen. Sie hatten den Weg mit der bereits erfolgten Beantragung des FleetNet Projektes geebnet und mit der Idee, Routing positionsbasiert zu machen, den Grundstein für alle späteren Publikationen gelegt. Zusammen mit meinem Zimmernachbarn Jörg Widmer haben sie mich ganz neu Lesen und Schreiben, wissenschaftliche Methodik, Gutachtenerstellung und etwa das Betreuen von Studenten gelehrt. Besonders in dieser Hinsicht war ich ein williger Schüler und habe während meiner Promotion alleine 15 Abschlussarbeiten und viele Hiwis und Praktikanten betreuen dürfen. Diese Arbeit hat mir sehr viel Spaß und auch wissenschaftlichen Erfolg bereitet, haben es doch die Ergebnisse mancher Abschlussarbeit in Publikationen und letzten Endes in diese Arbeit geschafft. Als jemand, der seine Heimat sehr gerne hat, hat mir die Arbeit mit Studenten auch immer wieder Hoffnung gemacht, dass doch nicht alles schlecht sein kann, wenn es so junge, intelligente, und fleißige Menschen gibt, die sich im Team allen Herausforderungen gestellt haben. Einen besonders großen Einfluss auf

meine Arbeit haben hierbei Michael Käsemann, Christian Lochert, Wolfgang Kieß, Michael Möske, Angelika Leibscher und Matthias Transier gehabt. Ganz besonders stolz bin ich auf die "meiner Studis", die nun selbst nach einer Promotion streben. So ist Matthias Transier ein wunderbarer Kollege geworden, dem ich fachlich und persönlich sehr viel verdanke. Von den vielen anderen Kollegen am Lehrstuhl, die immer wieder viel zum Werden meiner Gedanken beigetragen haben, sei besonders Gerald Kühne erwähnt, der mir gerade als sehr junger Doktorand, unheimlich freundlich und mit großer Geduld beim Lösen von Programmieraufgaben geholfen hat.

Stellvertretend für die vielen, nationalen und internationalen, Kollegen, die mir durch ihre Kommentare und Erklärungen wertvolle Beiträge zu meiner Arbeit geleistet haben, möchte ich Richard Gold von der Universität Uppsala in Schweden nennen, der mir unter anderem UNIX-Netzwerkprogrammierung auf einem ansonsten langweiligen FleetNet-Treffen beigebracht hat.

Einen extra Absatz möchte ich unserem "staff", insbesondere Frau Betty Haire-Weyerer widmen: Ihr ist es zu verdanken, dass ich diese Arbeit auf Englisch verfassen und damit einem größeren Publikum öffnen konnte. Dieselbe Leistung, nämlich die sprachliche Kontrolle meiner Arbeit, hat sie während der ganzen Promotion für die meisten meiner Aufsätze erbracht. Außerdem war sie vielen am Lehrstuhl die "Mutter der Kompanie" und hat nicht selten als Schnittstelle zu Prof. Effelsberg funktioniert. Auch ihrer Nachfolgerin Ursula Eckle und unserem Systemprogrammierer Walter Müller gilt mein Dank für deren Unterstützung.

Wie oben schon angedeutet hat das FleetNet Projekt lange Zeit die Rahmenbedingungen und Ziele meiner Arbeit vorgegeben. Damit nicht genug, hat es mir die Chance eröffnet, an bedeutsamer Stelle am Entstehen einer neuen Technologie mitzuwirken und hierbei viel über Entwicklungs- und Entscheidungsprozesse der deutschen Automobilindustrie zu lernen. Hierbei möchte ich besonders Herrn Doktor Walter Franz von DaimlerChrysler erwähnen, der mit großem Geschick FleetNet eine Richtung gegeben hat. An dieser Stelle muss natürlich auch NEC Network Labs Europe genannt werden, die lange Zeit meine Stelle kofinanziert haben. Neben Hannes Hartenstein war hierfür später Dr. Andreas Festag verantwortlich, immer unter der Führung von Dr. Heinrich Stüttgen. Vielen Dank für Dach und Brot und die wunderbaren vielen Ressourcen. Der wunderbarste Projektpartner im FleetNet-Nachfolgeprojekt "Network-on-Wheels" war für mich die Universität Karlsruhe mit dem wunderbaren Team von Prof. Hartenstein, zunächst Marc Torrent-Moreno und später Felix Schmidt-Eisenlohr. In ihnen leben viele unserer Ansätze weiter.

Keineswegs weniger wichtig für meine Arbeit waren mein privates Umfeld. Meinen Eltern, Oskar und Waltraud, verdanke ich neben meinem Leben meine Ausbildung und Förderung, Meine Mutter Waltraud, die leider während meiner Doktorandenzeit verstorben ist, hat mir vor ihrem Tod einmal gesagt, wie stolz sie mein aka-

demischer Erfolg macht. Mama, an Dich, im Himmel: Ich hab's geschafft. Vielen Dank!

In über fünf Jahren kommen und gehen viele Menschen. Ganz besonders wichtig sind hierbei die "alten Freunde" und die Familie, die immer da zu sein scheinen, wenn man sie braucht. Meine Schwester mit ihrer wunderbaren Familie, Herbert, Silas und Maika. Meine lieben Freunde Michael, Schmacko, Maddin und Axel. Meine Mentoren Michael Schmiady und Karl-Heinz Dubronner. Ob es nur Dampf ablassen, das versuchte Verhindern von Fachidiotie oder einfach wunderschönes Privatleben war. Ihr habt immer an mich geglaubt; manchmal mehr als ich selbst.

Holger Füßler, Mannheim, im Februar 2007

Abstract

Mobile Ad-Hoc Networks, or MANETs, are data communication networks between (potentially) mobile computer systems equipped with wireless communication devices and — in their purest form — in complete absence of communication infrastructure. Usage scenarios for these systems include communication during disaster recovery or battlefield communications.

One of the great research challenges concerning MANETs is the *Packet Forwarding Problem*, i.e., the question to which neighbor node a data packet should be handed over to reach non-neighboring nodes. While this problem has been previously solved by the adaption of classic routing algorithms from wired networks, the availability of GPS enables to include information about the geographic position of nodes into the routing decision, by selecting forwarders that are geographically closest to the destination. While these algorithms have been shown to improve communication performance in networks with a high degree of node mobility, they require (a) a beaconing service that allows every node to build a table of its neighbors and (b) a so-called *Location Service* that allows to acquire the current position of non-neighboring nodes in the network.

In this thesis, we propose *Contention-Based Forwarding* (or CBF), a greedy routing heuristic that is no longer in need of a beaconing service. Moreover, a forwarding node running CBF does not at all select the next forwarder explicitly but broadcasts the packet containing its own position and the position of the destination. The selection of the forwarding is now done in a contention period, where every possible forwarder, i.e., every receiver of the packet, considers its own suitability to forward by calculating the geographical progress for the packet if forwarded by itself. Then it waits for a time reciprocal to this suitability before simply retransmitting. If the retransmission of a packet is overheard, the own postponed retransmission process is canceled. In this thesis, we demonstrate that CBF outperforms beacon and position-based routing by delivering packets with constant overhead, almost ignorant of mobility. Also, we introduce two strategies to cope with the problem of packet duplication.

A problem left open by greedy routing heuristics is routing in the presence of local optima, or *voids*. Voids are node placement situations, where — in spite of an existing route — no neighboring node is geographically closer to the destination than the current forwarder. In these situations, greedy forwarding fails and standard graph-based recovery well known from classical Position-Based Forward-

ing cannot be applied due to the lack of the beacon-based construction of neighbor tables. As a solution, we propagate *Contention-Based Distance Vector Routing*, a contention-based adaption of AODV that acquires topology information in the area of the void and does contention on the topological distance to the forwarder.

Besides the forwarding algorithms, we extend position-based routing by two location services. The first, the *Reactive Location Service* or RLS is simple, purely on-demand and very robust to mobility, the second *Hierarchical Location Service*, is more complex but outperforms RLS in scalability.

The second big column in this thesis is ad-hoc multi-hop communication in the context of *Vehicular Ad-Hoc Networks*, or VANET, i.e., networks where the communication system is carried by vehicles. These systems very elegantly fit into the propositions and requirements for our more general routing approaches since they have (a) easy access to position information an (b) "suffer" from high mobility. For VANETs, we separate the routing problem into highway and city scenarios and study various routing algorithms in both. In the end, we advocate the usage of position-based routing in both scenarios; moreover, the contention-based approaches are most promising.

While a lot of ad-hoc research has been deemed to be theoretical, we have also built a multi-car communication system. For this system, we provided the network and system architecture and provided the communication software. In this thesis, we will describe these efforts as a proof-of-concept and provide measurement results.

Zusammenfassung

Mobile Ad-Hoc Netzwerke (oder MANETs) sind Netzwerke, bei denen mobile, mit Funktechnik ausgerüstete Netzwerknoten sich spontan zu einem Netzwerk organisieren. In seiner reinsten Form wird hier keine Infrastruktur in Form von, zum Beispiel, Sendemasten oder *access points*, verwendet. Die grundsätzlich akademische Ausrichtung der Erforschung dieses Netzwerktyps sieht unter anderem militärische Gefechtsfeldkommunikation oder etwa Kommunikation nach einen Ausfall der Kommunikationsinfrastruktur, zum Beispiel bei einer Naturkatastrophe, vor.

Eine der großen Herausforderungen für MANETs ist das *Problem der Paketweiterleitung* oder des *Routings*, d.h. die Fragestellung, an welchen Netzwerknachbarn ein Datenpaket weitergeleitet werden soll, um nicht-benachbarte Systeme zu erreichen. Während grundsätzlich Lösungen für dieses Problem schon früher beschrieben wurden - im Wesentlichen durch Adaption von Verfahren aus der Welt der drahtgebundenen Kommunikation - hat die Verfügbarkeit von preiswerten GPS-Empfängern und damit die Fähigkeit, die geographische Position zu ermitteln, einer neuen Klasse von Routing Verfahren den Weg geebnet. Diese Verfahren, positionsbasierte Routingverfahren genannt können nachweislich die Zustellzuverlässigkeit erhöhen, bzw. die dazu erforderliche Netzwerklast senken, insbesondere in Netzwerken, bei denen die Knoten hoher Mobilität unterliegen. Grundsätzlich funktionieren sie derart, dass jeder Knoten sogenannte *beacon*-Pakete schickt, die die eigene Position enthalten. Empfänger dieser Pakete erzeugen aus diesen Daten eine Tabelle über ihre eigene Netzwerk-Nachbarschaft. Wird Kommunikation mit einem Nicht-Nachbarknoten angefordert, wird ein sogenannter *location service*, das ist ein Netzwerkalgorithmus zur Ermittlung der Position von beliebigen, transitiv verbundenen Knoten, bemüht. Mit Hilfe dieser Zielposition und der Position der Nachbarn wird dann die Weiterleitungsentscheidung getroffen.

In dieser Dissertation schlagen wir ein neues Verfahren zur Weiterleitung von Datenpaketen in MANETs vor - das *Contention-Based Forwarding (CBF)*, oder etwa *Wettbewerbsbasiertes Weiterleiten*. Mit dieser Heuristik, die grundsätzlich nach dem *greedy* Prinzip funktioniert, wird der Einsatz von *beacon*-Paketen überflüssig. Dies wird dadurch erreicht, dass CBF den nächsten Weiterleiter nicht explizit berechnet, sondern das Paket an alle Nachbarn geschickt wird. Diese verwenden dann die im Paket-Kopf enthaltene Position des letzten Knotens und des Zieles, um die eigene Eignung zum Weiterleiten des Paketes als Funktion des Distanzfortschritts zu be-

rechnen.[1] Jeder potentielle Weiterleiter, d.h. jeder mit positivem Distanzfortschritt, wartet nun eine Zeit reziprok zu seiner Eignung. Hat er bis zu diesem Zeitpunkt keine Weiterleitung mitgehört, nimmt er an, der am besten geeignete Knoten zu sein und sendet das Paket. Das Mithören einer Weiterleitung führt hingegen dazu, den eigenen Sendewunsch zu verwerfen. In dieser Arbeit zeigen wir unter anderem, dass die Zustell-Leistung von `CBF` die des klassischen Routings mit Positionen übersteigt, mindestens jedoch - bei gleicher Leistung - das Zustellen mit beinahe mobilitäts-unabhängigen Kosten erreicht. Außerdem stellen wir zwei wirkungsvolle Strategien vor, mit denen Paket-Duplikate vermieden werden können.

Ein Nachteil von *greedy* Routing-Heuristiken ist die fehlende Vollständigkeit. Dies bedeutet, dass es Knotenkonstellationen, sogenannte *voids* geben kann, bei denen diese Heuristiken keine Route zum Ziel finden, obwohl es eine solche gibt. Das ist genau dann der Fall, wenn keiner der Nachbarn eines Weiterleiters näher am Ziel liegt als er selbst. Für diesen Fall sieht das klassische Routing mit Positionen sogenannte *recovery* Strategien oder Rettungsstrategien. Diese beruhen allerdings auf der verteilten Planarisierung des Nachbarschaftsgraphen, der bei unserem wettbewerbsorientierten Verfahren nicht aufgebaut wird, was die direkte Verwendung dieser Verfahren ausschließt. Deswegen stellen wir in dieser Dissertation *Contention-Based Distance Vector Routing* oder *CBDV* vor. Hierbei handelt es sich um eine wettbewerbs-version des bekannten `AODV`. Dieses Verfahren gewinnt - um den *void* herum - Topologie-Informationen, die es dann dazu benutzt, diesen zu überwinden.

Neben den Verfahren zur Weiterleitung präsentieren wir zwei der oben als *location service* (`LS`) bezeichneten Verfahren zur Ermittlung der geographischen Position von Knoten, die sich irgendwo in der transitiven Hülle befinden. Der erste, der *Reactive Location Service* oder `RLS` ist ein sehr einfacher `LS`, der nur dann Pakete austauscht, wenn er tatsächlich angefordert wird. `HLS`, oder *Reactive Location Service* hingegen, tauscht permanent Informationen aus. Dies fördert zum einen die Anfragegeschwindigkeit in großen Netzen und wirkt sich zum anderen positiv auf die Skalierbarkeit aus, offensichtlich auf Kosten von Implementierungskomplexität.

Während Routing in *Mobile Ad-Hoc Networks* den Schwerpunkt der Arbeit darstellt, ist das zweite große Standbein die Anwendung dieser Verfahren bei `MANET`s, bei denen die Netzwerknoten in straßengebundenen Fahrzeugen montiert werden. Diese vehikulären Ad-Hoc Netzwerke, oder *Vehicular Ad-Hoc Networks* (`VANET`s), sind erst kürzlich in den Blickpunkt der `MANET` Forschung gerückt. Sie sind für Routing-bezogene Forschung deswegen besonders interessant, weil sie zum einen hoher Mobilität unterliegen und zweitens der Zugriff auf Positionsinformationen in modernen Fahrzeugen durch integrierte Fahrzeugnavigationssysteme kein Problem darstellt. In dieser Arbeit zeigen wir, dass sich eine getrennte Betrachtung des

[1] Die Ermittlung der Position des Zieles wird hierbei wie im Standardfall einem *location service* überlassen.

VANET-Routingproblems, zum einen für Autobahnen und zum anderen für Städte, als sinnvoll erweist. Danach zeigen wir für beide Teilbereiche einsetzbare Verfahren und diskutieren Stärken und Schwächen. Am Ende empfehlen wir die Verwendung von positionsbasierten Verfahren. Insbesondere kommen wir zu dem Schluss, dass wettbewerbsbasierte Verfahren in vielen Fällen vorteilhaft sind.

Viele Forschung in MANETs ist vorwiegend theoretischer Natur. Während der Anfertigung dieser Arbeit hatten wir das Privileg, im Praktischen ein Kommunikationssystem zur Fahrzeug-zu-Fahrzeug Kommunikation entscheidend mit zu gestalten, zu implementieren und zu erproben. Dieses, und einige gewonnene Evaluationsergebnisse, werden wir in dieser Arbeit ebenfalls darstellen.

Contents

List of Figures

List of Tables

Abbreviations

ACK	Acknowledgement
ACM	Association for Computing Machinery
ACT	Active-Selection Suppression (In Context of CBDV)
AG	Aktiengesellschaft (german for *corporation*)
AHN	Ad-Hoc Network
ALOHA	a Channel-Access Protocol
ANSI	American National Standards Institute
AODV	Ad-Hoc On-Demand Distance-Vector Routing
AR	Area-Based (In Context of CBDV)
ATP	Ad-Hoc Transport Protocol
BAS	Basic Suppression (In Context of CBDV)
BEXP	Beacon Expiry Interval
BINT	Beacon Interval
BLR	Beaconless Routing
BMB+F	Bundesministerium für Bildung und Forschung - German Federal Ministry for Education and Research
BMW	Bayrische Motorenwerke (german car manufacturer)
CAN	Car Area Network
CBDV	Contention-Based Distance-Vector Routing
CBF	Contention-Based Forwarding
CPU	Central Processing Unit

CRC	Cyclic Redundancy Check
CSMA/CA	Carrier Sense Medium Access / Collison Avoidance
CTF	Clear-To-Forward
CTS	Clear-To-Send
CXCC	Cooperative Crosslayer Congestion Control
DARPA	Defense Advanced Research Projects Agency
DCF	Distributed Coordination Function
DES	Discrete Event Simulation
DLM	Distributed Location Management
DREAM	Distance Routing Effect Algorithm for Mobile Ad-Hoc Networks
DSDV	Destination-Sequenced Distance-Vector Routing
DSR	Dynamic Source Routing
DV	Distance Vector
EAC	Expected Additional Coverage
EASE	Location Service Algorithm
EE	Electrical Engineering
FARSI	Driver Behaviour Simulation Tool
FCAN	FleetNet Car Area Network
FEC	Forward Error Correction
FND	FleetNet Demonstrator
GG	Gabriel Graph
GHT	Geographic Hash Table
GLS	Grid Location Service
GOAFR	Greedy Other Adaptive Face Routing, pronounced as “gopher”
GPCR	Greedy Perimeter Coordinator Routing

GPRS	General Packet Radio Service
GPS	Global Positioning System
GPSR	Greedy Perimeter Stateless Routing
GRSS	Geographic Region Summary Service
GSB	Geographically Scoped Broadcast
GSR	Geographic Source Routing
HLS	Hierarchical Location Service
ID	Identifier
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISM	Industrial, Scientific, Medical Band
ISO	International Organization for Standardization
LAN	Local Area Network
LAR	Location-Aided Routing
LEXP	Location Expiry Interval
LLC	Logical Link Control
LS	Location Service
LUNAR	Lightweight Underlay Network Ad-Hoc Routing
MAC	Medium Access Control
MACA	Multiple Access with Collision Avoidance
MACAW	Multiple Access with Collision Avoidance for Wireless
MANET	Mobile Ad-Hoc Network
MB	Mega-Byte
MFR	Most Forward within Radius
MIT	Massachussetts Institute of Technology

MLS	Mobile Location Service
NEC	Nippon Electric Company
NFP	Nearest Forward Progress
NIC	Network Interface Card
NP	Non-deterministic Polynomial time
NT	Neighbor Table
OLSR	Optimized Link-State Routing
OS	Operating System
OSI	Open Systems Interconnection Reference Model
OSPF	Open Shortest Path First
OTCL	Object Tool Command Language
PBF	Position-Based Forwarding
PBR	Position-Based Routing
PC	Personal Computer
PCF	Point Coordination Function
PDF	Probability Density Function
PHY	Physical Layer
PRNET	Packet Radio Network
QoS	Quality of Service
RAD	Random Assessment Delay
RC	Responsible Cell (In the context of HLS)
RFC	Request for Comment
RIP	Routing Information Protocol
RLS	Reactive Location Service
RNG	Relative Neighborhood Graph

RTF	Request to Forward
RTS	Request to Send
RWP	Random Waypoint Model
SNR	Signal-to-Noise Ratio
SOTIS	Self-Organizing Traffic Information System
TBRPF	Topology Broadcast based on Reverse-Path Forwarding
TCL	Tool Command Language
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
TDMA	Time Division Multiple Access
TSB	Topologically Scoped Broadcast
TTL	Time-to-Live
UDG	Unit Disk Graph
UMTS	Universal Mobile Telecommunications System
UNI	Unicast Recovery Forwarding (In Context of CBDV)
VANET	Vehicular Ad-Hoc Network
WSN	Wireless Sensor Networks
WTCP	Wireless TCP
ZRP	Zone Routing Protocol

Chapter 1

Introduction and Overview

> For a successful technology, reality must take precedence over public relations, for Nature cannot be fooled.
>
> *(Richard P. Feynman)*

"Always On" in a Mobile World

Today's society — at least in the western world — is strongly influenced by two major technological developments of the last two decades. First, the invention and the deployment of the Internet has opened a variety of new ways for communication, economics and the ability to perform almost any task remotely. Moreover, a lot of the facilities offered on the Internet have taken and are taking over services that have been around for much longer, like voice telephony with Voice-over-IP [58] or paper-based mail with electronic mail. Another interesting fact to note is the behind-the-scene domination of network technology by protocols developed for and used on the Internet. For example, a lot of today's voice telephony connections already run over IP—the Internet Protocol, the old circuit-switched telephone serving as a mere front-end. In fact, it seems as if every non-IP protocol is an endangered, if not already extinct, species.

The second development is the increasing mobility of society, its technological enabler being cellular radio technology and Wireless LAN. The former started in Germany in 1986 with the bulky and expensive analog C-Net phones, went over the digital D/E-Networks and is now approaching third generation UMTS networks. The latter, Wireless LAN, is more short-range, but high-bandwidth oriented, its main economical difference lying in the usage of ISM (Industry, Scientific, Medical) bands [42]. Both technologies together have convinced users to work with the Internet not only at their desktop PC at work, but also in wireless mode at home, at their favorite coffee place, in airport lobbies, or just anywhere in cellular-covered areas.

Combining both technologies, users not only use the Internet for numerous things both in their private and economic lives, they do it without necessarily connecting to a network cable. Technology to do that is available and is becoming increasingly inexpensive.

From Infrastructure to Ad-Hoc Networks

So far we have analyzed the evolution of networking in an infrastructure-rich environment, i.e., wherever we find a Wireless LAN access point or a cellular base station. Now what happens in environments, where people might want to do networking, even have a wireless network interface, but do not have connectivity to the infrastructure? While this seems to be a bit far-fetched to someone living in a modern first-world city, rural areas with less infrastructure do exist. Besides that, catastrophes like Hurricane Catrina in 2005 are quite persuasive examples demonstrating that an infrastructure-less communication network would have been handy even if only for use as a backup.

The solution to the absence of infrastructure is intuitively clear: Wherever a network device may be, it looks for other devices in its radio vicinity and establishes means to communicate with them in an ad-hoc fashion. And whenever nodes have disjoint sets of radio neighbors, they can use intermediate nodes to communicate with transitively-connected nodes by using others as forwarders. These spontaneous networks need special protocols to cope with the absence of infrastructure and central organization.

As in many industrial innovations, the DARPA [30] (Defense Advanced Research Projects Agency) played a crucial role when it ordered a feasibility study on the provision of mobile hosts with network connectivity in the PRNET (Packet Radio Network) project — with a military intention. Also military scenarios — until today — play an important role in U.S. research on the topic.

After the kick-off in this area of research, new fields have emerged with new applications and special challenges, the most prominent being MANETs, which stands for *Mobile Ad-Hoc Networks*. What MANETs are was very much defined by the homonymous working group of the Internet Engineering Task Force (IETF). In their view, a MANET is a wireless network in which (network) nodes are potentially mobile, and in contrast to the Internet, always functioning both as router, i.e., a packet forwarder, and an end system, i.e., a node using the network to communicate. Since the thinking came from IETF, the idea was to transparently send IP over a network formed by wireless-equipped hosts (notebooks or hand-helds) who happen to be in radio range of each other. In this context, it is no wonder that the main challenge identified was "packet routing", it being the traditional IETF area of work. Furthermore, wireless network interface cards already existed as did TCP/IP with its wonderful range of applications and compatible systems, making it seem evident that the "missing module" is multi-hop packet routing and the major challenge to these protocols is node mobility, creating constant changes in network topology.

The MANET effort led to an explosion of publications on routing, later accompanied by extending to protocol layers below or above or examining operating system issues. Simultaneously, the IETF MANET group put out RFCs (Requests for Com-

ments) and researchers started implementing and evaluating protocols but, even today MANET protocols are rarely deployed in current (civil) networking products. The reason for this is asked by many researchers, and the argumentation varies. Some say, that research is too theoretical and the protocols rarely work in reality [295]. A different argument could be, however, that honestly they are hardly ever needed by most of today's civilian network users. Remote areas with no networking wires are rather covered with satellites than with MANETs where forwarders not only have to exist but also have to be activated as well as be in the proper geographic position to be able to form a routing chain to the desired communication end-point.

From MANETs to VANETs and back

Fortunately, the less than satisfactory deployment of MANETs is not a dead end. From the original idea, a multitude of scientific and industrial spin-offs have emerged. Instead of focusing on transparent IP unicast for MANET-to-MANET or MANET-to-Internet communication, these fields focus on special flavors of networks and their specific challenges, most of them exploiting the locality of the desire to communicate.

One of these "special" *Mobile Ad-Hoc Networks* is called *Vehicular Ad-Hoc Network*, or VANET. As the name indicates, the typical node in a VANET is a car or a truck that is traveling on a road. While Internet connectivity is also an issue in these networks, the most likely mode of communication will be node-to-region rather than node-to-node or node-to-Internet, the basic focus being on Vehicular Safety. Additionally, a special feature of VANETs with their very high node mobility and their unique movement patterns.

The development of protocols to facilitate communication in *Vehicular Ad-Hoc Networks* was our main mission at the beginning of the work on this thesis in late 2001. We took this task as a subcontractor of *NEC Network Labs Europe* within the framework of the BMB+F (the German Federal Ministry for Education and Research) project "*FleetNet*—Internet on the Road". In this project, the desired mode of communication was, as in MANETs, to create transparent IP connectivity between two vehicles or one vehicle and the Internet. Without going into details here, this task proved to be very difficult since traditional IETF MANET protocols perform poorly in the presence of the movement behavior typical of vehicles, in particular their speed. However, given the assumption of the availability of the vehicle's geographic position, we could show that the usage of this position information does the trick in vehicular highway scenarios. Nonetheless, in cities, with their obstacles and their maze-like structure, the notion of position is much harder to transform into a packet-forwarding decision. Intuitively, knowledge about the

"street topology" can be of use there. Thus, we propose different protocols for use in cities, with and without the assumption of having a digital street map.

An orthogonal but very important aspect of networking in general is the implementation and evaluation of a network system in action. We have led the technical part in the *FleetNet* project's effort implementing transparent IP connectivity and special VANET protocols both on the system/implementation side and on the practical evaluation side and fed the results back into protocol development.

Some of the insights gained in the process of building a real vehicle-to-vehicle communication system have lead to the development of a rather (r)evolutionary manner of packet forwarding using geographic positions. This method — called CBF for Contention-Based Forwarding — is also one of the major contributions of this thesis. CBF is also a good example of how VANET-inspired protocols can be generalized for generic *Mobile Ad-Hoc Networks*. For this thesis, we have chosen the approach to proceed from general to special topics.

Thesis Overview

The thesis is structured as follows: The next chapter (Chapter 2) gives an overview of the research area, explaining the foundations this thesis builds on. Here, we will focus on spreading out the variety of MANET flavors and their idiosyncrasies. Then, Chapter 3 will dive into our contributions to the field of routing based on geographic positions for general MANETs. In this chapter, we will also introduce Contention-Based Forwarding as the heart of this thesis. Proceeding from general to special, Chapter 4 first lists VANET-specific challenges and then shows how to solve them with position-based packet-forwarding algorithms. Furthermore, we show how unicast data forwarding extends to other forwarding modes. Chapter 5 moves on from theory to practice, describing our achievements in the area of real-world implementation and measurements of *Vehicular Ad-Hoc Networks*. Finally, Chapter 6 concludes this thesis, summarizing its content and pointing out thoughts on further exploration of the topic.

Chapter 2

Fundamentals and Methodology

> It is a miracle that curiosity survives formal education.
>
> *(Albert Einstein)*

Chapter Outline
This chapter is all about introducing aspects of mobile ad-hoc networking necessary to understand the later chapters, especially Chapters 3 and 4.

We define *Mobile Ad-Hoc Network*s as *"self-organizing networks consisting of wireless-equipped computers that may be able to change their position"*. With this very general definition, all subsequent definitions of networks made here are a subtype to `MANET`s. Fundamentally, this thesis is about routing, i.e., the usage of multiple network nodes to transport a data packet from one node to another [284]. Thus, the following foundations are focused on routing, especially *unicast routing*, where one specific node, identified by a unique identifier, is exchanging data packets with another node, also identified by a unique identifier.

The chapter is organized as follows: First, we give a brief history of mobile ad-hoc networking from a research perspective. Then, we classify *Mobile Ad-Hoc Networks* by first naming and describing six possible dimensions of classification and then listing and classifying four major types of *Mobile Ad-Hoc Networks*. The following Section 2.3 talks about the lower-than-networking layers and their implications for the layers above. Section 2.4 classifies well-known routing strategies while Section 2.5 describes examples. Since the network layer provides forwarding services for transport, we have also included Section 2.6 describing transport in *Mobile Ad-Hoc Networks*. Concluding the chapter is a section (2.7) on the methodology used in this field of research and relevant for this thesis.

2.1 A Brief MANET History

`MANET`s were brought into the world by the `DARPA`, the U.S. Defense Advanced Research Projects Agency, a U.S. governmental institution that coordinates and funds military-related research. `DARPA` started the `PRNET` [166] (Packet Radio Network) project in 1972, desiring to create a network that served to connect mobile wireless

nodes among each other and to infrastructure networks, its basic purpose being to supply reliable communication in a combat environment, demonstrating the feasibility by the early 1980s[1].

While the DARPA project continued, not many people worked on the subject and aside from PRNET, publications mainly focused on theoretical results (e.g. [282]).

The second echelon of MANET research was launched in 1995, when the ACM (Association for Computing Machinery) created the "Annual International Conference on Mobile Computing and Networking" or short MobiCom [2] conference hosting all sorts of wireless computing and networking papers. From then on, many researchers with backgrounds in both electrical engineering and computer science started to work in the field. When MANETs constituted a significant number of the research papers, a MobiCom workshop named MobiHoc [3] went independent focusing solely on *Ad-Hoc Networks*. Today, many workshops exist slicing the field of *Ad-Hoc Networks* into many sub-topics, and every networking conference hosts sessions or even entire tracks about topics covering MANETs.

The focus of MANET research was — at the beginning — very much on MANET routing since the network layer was identified as the research challenge. Mobility and the different link properties encountered in a mobile wireless system were new features to cope with, while the problem routing tried to solve was conventional unicast or multicast. Each of these transport modes believed in an explicit allocation of network participants to the communication process, like a pair of unique IDs in the case of unicast, or a one-to-many or many-to-many approach in multicast, where groups are still explicitly formed by a join/leave algorithm.

A very important landmark in MANET research is the paper by Gupta/Kumar [144] which analytically shows that even with optimal node placement the throughput of wireless networks grows in the order of $\Theta\left(\frac{W}{\sqrt{n}}\right)$, where W is the radio bit-rate and n is the number of nodes and $\Theta()$ is an asymptotically tight bound for the algorithmic complexity[2]. This directly implies that communication should better be locally contained, pretty much destroying the vision of an *Ad-Hoc Network* on a global scale. Today, people very much acknowledge this law, which is even worse in reality [295, 143] with the consequence that unicast/multicast communication should be used for local communication only, meaning that the hop-scope for communication should be limited in order to keep the network from collapsing.

Another direction of research evolved on protocols that do not bi-directionally connect node tuples at all but rather disseminate information through the network [162]. In accordance with this is the practice of addressing nodes not by their ID but by node properties such as the node's position [229, 180] or by its capabilities. However, while these networks, usually called Wireless Sensor Networks (or

[1] An overview of military involvement in *Mobile Ad-Hoc Networks* can be found in [120].
[2] The big-O, big-Θ is discussed in, e.g., [141, 20]

WSNs) are of an ad-hoc kind of nature and share some properties with MANETs, they are usually treated seperately.

Besides the practical confinement to a limited number of hops for the scope of communication, MANETs have evolved from theory to reality, opening new fields of research with regard to system issues and even the architecture of protocols for these networks. Therefore, the third big MANET symposium of the ACM is the "Conference on Mobile Systems, Applications, and Services" or MobiSys. The evolution toward real systems also brought into play layers above and below networking.

In the remainder of this chapter, we will focus on unicast routing, keeping in mind that communication will occur within a limited hop range. First, however, we will classify *Ad-Hoc Networks*, identifying several distinctive properties and then discussing the protocol layers below the network layer as the foundation of every multi-hop protocol.

2.2 Classification of Mobile Ad-Hoc Networks

As you will see in the remainder of this section, a great variety of of *Ad-Hoc Networks* has emerged from the original "extending the Internet" idea. This multitude of MANET types calls for a classification. In this classification, we will sort different types of MANETs into buckets according to their similarity with regard to communication protocols. To achieve this, we will treat the entirety of ad-hoc networking similar to a vector space, where the dimensions, often on an ordinal or even nominal scale, represent a certain network characteristic such that any network can be represented as a tuple in this space. After describing the dimensions we will exemplarily list important network types according to the previously established classification.

2.2.1 Dimensions

As depicted in Fig. 2.1, we have chosen to structure the dimensions into three groups, namely "Node Mobility", "Node Resource Restrictions", and "Communication Requirements", which we are going to discuss in the next three sub-sections.

Node Mobility

Mobility seems to be the most distinctive property of MANETs when compared to wired networks. The notion of mobility and ad-hoc has been agreed upon to inherently include the usage of wireless network adapters since wired networking in this context would imply plugging and unplugging connections. However, in the context of a classifying dimension, we refer to the *degree* of mobility assuming that even in a networks where nodes do not move, they would have been brought into position

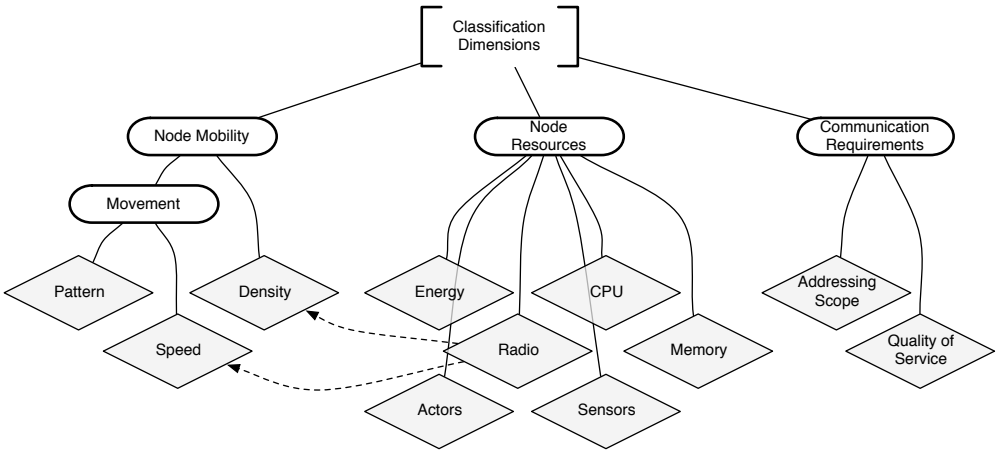


Figure 2.1: MANET classification dimensions

by a mobile process and need to establish the network in an ad-hoc fashion. It is important to keep in mind that even in a non-moving network, nodes can change neighborhood relations due to node failure or radio fluctuations.

However, these neighborhood relations are the essence of node mobility for communication protocols since they define the topology of the network. As the main properties determining node mobility, we identify the measures for **Node Movement** and **Node Density**. Node movement again can be classified (a) by the pattern the node movement follows and (b) the speed in which these patterns are executed. In both cases, the actual mobility is to be seen in reference to the area covered by the node's radio transceiver. The reason can easily be seen in Figure 2.2: Assume node M moves along the blue arrow to M' at a given speed and has the nominal radio range depicted by the dashed circle. While doing that, it gains 8 new neighbors while losing 7 old ones, only one neighbor stays the same. However, assuming the greater dotted radio range; the number of new neighbors between M and M' is 16, while 18 neighbors are lost and 20 neighbors at the intersection. Thus, movement speed has to be understood in relation to radio range. The same applies for understanding the relevance of a certain node density, which is usually given in $\left[\frac{\text{nodes}}{\text{area}}\right]$.

So far, we have not really talked about movement patterns themselves but more on the effect movement has on an *Ad-Hoc Network*. Examples for special movement patterns are, e.g., cars moving on a highway or in a city (we will hear a lot more about these in Sections 4.3 and 4.4), or pedestrians in a mall, or sensors dropped from a plane and spread out over an area (in this case, mobility is reduced to position).

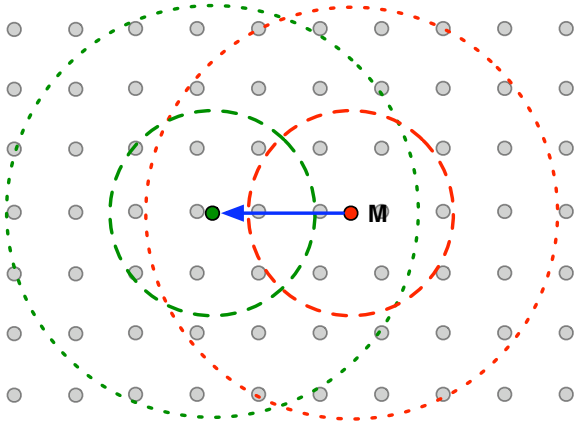


Figure 2.2: Effects of node movement with respect to radio range

When talking about these special motion scenarios, we must also take a look at virtual movement patterns used in research only. Reality, however, is sometimes hard to model or even hard to observe; certainly it is not easy to get a grasp of analytically. All these are reasons for simple node movement models that can be easily parametrized, and can be analytically understood with reasonable effort. The most famous model of this kind is the so-called *Random Waypoint Model* (or `RWP`), which was first described in a `MANET` context in [165] and has been extensively used since in a great variety of simulations for protocol evaluation. In addition to its usage, a multitude of papers address its analytical properties and advice on its usage [72, 73, 74, 307]. A survey of ad-hoc mobility models can be found in [91].

The pure vanilla *Random Waypoint Model* works as follows, an example depicted in Figure 2.3: We assume a rectangular plane of size $X \times Y$. For every node, we pick a position (x, y) from this rectangle, following uniform distribution for each coordinate marking the initial position or waypoint wp1. Whenever a node is at a waypoint, it stays there for a certain interval (called pause time), which is either fixed or also random. After that pause time, the next waypoint is randomly selected. Also, a speed is selected randomly out of a given interval. When this is done, the node moves at the selected speed to the next waypoint. Thus, the time wp2 is reached depends on the speed selected at wp1.

Out of the many analytical properties of the `RWP`, we just list two rather obvious ones (also to be found in the analytical papers listed above): First, if a node has reached a waypoint close to the network border (e.g., at wp2), it is much more likely to select a waypoint bringing it back towards the center rather than one moving it closer towards the border (denoted by the gray area). Thus, the first observation is that over time the node density is likely to converge towards the

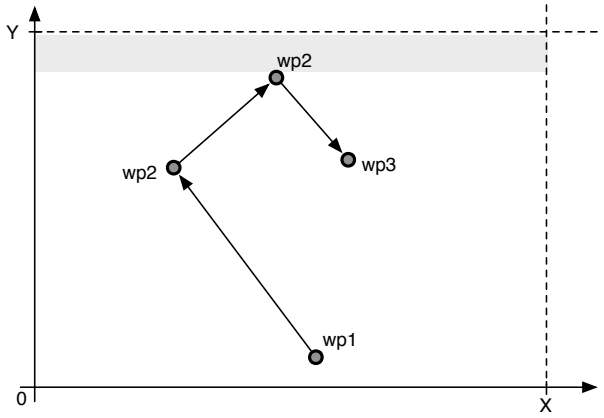


Figure 2.3: Example of *Random Waypoint Model* mobility

center. At the beginning, nodes are evenly distributed. The second observation concerns the speed. Every node selecting a slow speed will need more time to reach its next waypoint, which is the point where we select the next waypoint/speed tuple. Consequently, over time, the average speed of the network will slow down. Put to the extreme, if a node is allowed to set zero as a speed, it will never move at all. While this is something that RWPs designers were certainly aware of, there are simulation studies that allow for zero mobility to happen, which results in a network whose mobility dozes off over time.

Both observations of RWP listed above can be desired for a study. We are just mentioning them to keep them in mind. Fundamentally, there are two main parameters to influence mobility using RWP: The distribution of speeds and the distribution of the pause intervals. A lot of studies keep one of both for all simulations and change the other, e.g., simulating an increase in mobility by means of a decrease in (average) lengths of the pause interval.

Here we directly see a fundamental advantage of virtual mobility patterns: the possibility to easily adjust mobility as a constraint for *Mobile Ad-Hoc Networks*. Real movements in contrast, cannot easily be accelerated or decelerated without losing their validity.

Node Resource Restrictions

This dimension deals with the resources available to a single node in the network, mostly related to computing and networking, but also to other node qualities like self-propelling or sensory capabilities. Especially important in the context of communication are the following:

Energy Most MANETs are implicitly not connected to external power supplies [3]. Thus, energy capacity is defined by a local power supply such as, e.g., a battery, a fuel cell, a solar panel or a dynamo, each having its own limitations for energy provision. On the other hand, communication — including transmitting *and* listening , and computation all consume energy, thus defining the *energy endurance* of a node. We will see later that this is a crucial factor for many flavors of *Mobile Ad-Hoc Networks*, especially for nodes that cannot be re-charged and thus live only for their energy lifetime.

Radio The radio is the main delimiter for the node's capability to communicate, and with every node, it defines the network. Slightly ignoring the details of a radio system, we subsume the whole system providing the transmission of a bit stream to a neighboring node under this item. Just to list some basic qualities, a radio could have directed / undirected antennas defining the shape of radio propagation, we have systems using different physical encoding schemes, and we have more basic properties like the frequency band used or the emitted power and the modulation scheme that define important properties such as the radio range, error and collision resilience and the ability to penetrate obstacles.

Sensors Sensors fundamentally observe the environment of a node and transform their perception into digitally accessible data. Their greatest impact as part of a mobile and communication-enabled node is thus on applications with the purpose of sensor data collection. However, they also affect the communication system mostly through the sensors' acquisition of position information about a node, which can help e.g., packet forwarding a lot (see Chapter 3).

Actors Actors empower a network node to influence its environment in this context by means other than by communication. An example of this is a robot that is able to e.g., mechanically move things around or a car that is able to turn on lights etc. As an actor with an influence on the communication system, self-propulsion is a good example since nodes could arrange their position to form a different network.

CPU The processing power of a node delimits its capability of using complex algorithms for communication and data processing or encoding. In a lot of cases there is a trade-off between "intelligent" and "less-intelligent" communication, basically describing that the communication protocols can be improved trading in more processing.

Memory The storage capacity on a network node similarly delimits communication protocols as with the processing power. E.g., having a big storage capacity

[3] Unless you would build a MANET consisting, e.g., of trains.

could empower a node to store geo-topological information about the node's environment, thus enabling it to improve route selection.

Communication Requirements

The third group of properties defining a *Mobile Ad-Hoc Network* is named "Communication Requirements". In this division we list requirements that applications impose onto the communication protocol for a given MANET. The first big item here is the "Scope of Addressing", which basically answers the question of which node is talking to which other nodes. Trying to narrow this down, we first define some node properties usable for addressing, being (i) a unique node identifier, (ii) the nodes situation or context, and (iii) the physical or logical type of the node. The first, a unique identifier, is simply a number or a symbol different from all others in the network and thus able to serve as a distinguishing property.[4] The second, call it "node situation", is a digital feature describing the environment situation, or the context, a node is in. Such a feature might be the node's position or the nodes environmental temperature. The third characterizes nodes by their physical or logical type, the former distinguishing, e.g., a green car from a blue one or a node with a temperature sensor from one without. Logical type is used for artificial types imposed on the nodes, such as "being a location server" etc.

With the differentiation properties described above, we now describe typical addressing schemes, visualized in Fig. 2.4. In this figure, we assume node 1 to be

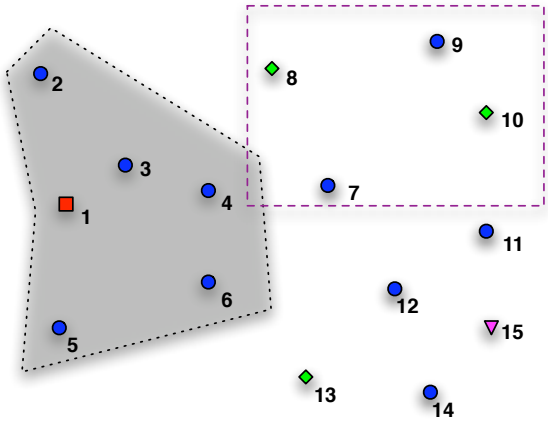


Figure 2.4: Addressing examples

[4] For security purposes or the purpose of address auto-configuration [297], uniqueness is often required only in a local context. For this work, however, we assume IDs to be globally unique (see also Section 2.5.1).

a node wanting to communicate, and the number to represent the global identifier. The other nodes 2 to 15 are located according to the figure and have different descriptive properties depicted by the geometric shape and color.

Unicast is communication between two uniquely described nodes, e.g., in the picture, if node 1 communicates with node 15.

Multicast (recipient list) is communication where the sender knows all recipients and addresses them directly. In our example, this would occur if 1 sent a packet to, say, 8, 10, and 13, and included their IDs in the packet.

Multicast (group) : in this addressing mode the sender does not necessarily know about the recipients but sends his packet to a multicast group. Recipients use a so-called group management algorithm to display their desire to participate in a certain group's transmissions.

Geocast is the addressing of nodes qualified by their geographical position. Usually, this is accomplished by stating an absolute area by means of a (desirably simple) geometrical figure. In our example, 1 might address the dashed rectangle, implicitly selecting nodes $7 - 10$ as recipients.

Anycast is the addressing of the "next-best" node that has a certain descriptive property, e.g., the "next-best" node carrying a temperature sensor. Assuming all blue nodes carry a temperature sensor, anycasting, depending on the algorithm used, might resolve to node 3. A special case is to do anycast based on the geographical position. In this geo-anycast, the "first best" node within a geographic region would be addressed.

Broadcast usually refers to addressing all nodes. However, in the context of MANETs, broadcast is almost always limited by the number of hops a packet is allowed to traverse. E.g., the dotted polygon depicts a two-hop neighborhood of node 1, namely nodes 2 to 6.

Tab. 2.1 summarizes the addressing schemes listed above. At this point, we want to emphasize the fundamental difference between "the nodes which are addressed" and "the nodes which are reached" with regard to the forwarding algorithm chosen and the mobility / radio situation. Any one addressing node using one of these schemes has to be aware that the underlying forwarding algorithm might not reach the addressed nodes at all or only a subset of them.

Understanding the quality of the service offered by a communication system, we come to the second column of the "communication requirements" postulated in this section. In any given MANET, this quality-of-service (or QoS) describes the requirements towards QoS parameters such as reliability or timeliness. While in a

	1-to-1	**1-to-many**
ID	unicast	rcpt. list multicast
Logical Property	anycast	group multicast
Position	geo-anycast	geocast / flooding

Table 2.1: Addressing mode schematic

standard MESH network with IP services neither reliability nor timeliness would be required above certain limits, both parameters could be crucial for a car notifying other cars of an accident in a VANET.

For the sake of completeness, we will state another fundamental distinction with regard to communication requirements, i.e., the distinction between packet forwarding and information forwarding. We define packet forwarding as the end-to-end delivery of an unaltered packet payload, i.e., between the sender and addressed recipient(s), which is the classical approach taken by Internet protocols. However, in some MANETs such as sensor networks, this notion is changed to "information forwarding", meaning that the packet payload might be changed on its way through the network. With this generalization, sensor nodes are able to aggregate information, i.e., merge their local knowledge with the knowledge contained in the packet.

Having concluded the classification dimensions, we now classify MANET instances.

2.2.2 Classic Mobile Ad-Hoc Networks

The first example we discuss is the "classical MANET". In such a network, MANET protocols are mainly meant to extend an existing datagram network such as the Internet, originally often in a military context. Hence, the typical mobile hardware platform would be a notebook equipped with a wireless NIC (network interface card). Thus, they face rather low hardware resource constraints when compared to, e.g., sensor networks as discussed below . Mobility would be determined by human mobility when carried by a person, up to vehicle mobility when installed on a truck or a plane. Also, the zero mobility case was considered in these networks, both as a hybrid part together with mobile elements and as a completely static scenario. Sensors and Actors are not a focus in classical MANETs, it being their center of interest to run IP-style applications.

Consequently, research activities in this type of *Ad-Hoc Network* mainly focused on unicast routing transparently integrated in the IP layer, i.e., one node should be able to communicate with another node via a globally unique address. These algorithms should be able to cope with arbitrary node movements while energy is not the most important concern. While hardware was believed to be custom-made in the early days, off-the-shelf wireless hardware is now the assumed radio

system. Since the basic goal for these systems is to be used with TCP-style transport protocols, best-effort datagram traffic is usually sufficient for the IP layer.

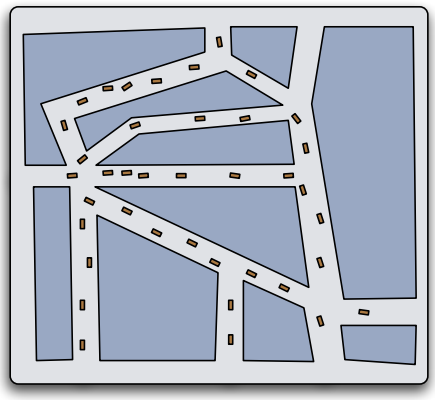
2.2.3 Vehicular Ad-Hoc Networks (VANETs)

Vehicular Ad-Hoc Networks (or VANETs) are MANETs in which communication systems are installed in surface-bound (usually street-bound) vehicles. For this description of VANETs, we will first address the mobility characteristics of such street-bound vehicles.

Mobility Characteristics

At least in first-world countries, vehicle traffic is mostly confined to roads. Whenever it is not, this is usually accompanied by a dramatic loss in node density, resulting in the lack of the possibility to communicate locally. Thus, VANET research concentrates on roads, more so on roads with some reasonable node density. From a birds-eye perspective, we identify two basic VANET mobility patterns: (a) highway movement and (b) city movement (see Figure 2.5).

(a) Highway



(b) City

Figure 2.5: Vehicular movement scenarios

Vehicular highway movement (Figure 2.5(a)) is largely characterized by one-dimensionality since junctions are infrequent and even if they do occur, they do

not interfere with flowing traffic. Furthermore, the width of a highway is negligible compared to the length. Traffic moves in both directions, creating very high inter-direction mobility, while intra-direction mobility is comparably low. Usually, highway scenarios are also assumed to be free of radio obstacles. Figure 2.5(a) shows a typical section of a highway scenario with two lanes per direction, the white arrows denoting the bearing. The green stripe denotes the road median separating the directions.

The second basic movement pattern is called city movement or urban movement. It is characterized by lower absolute car speeds than those on the highway, including a significant number of cars that do not move at all, e.g., because they have stopped at traffic lights. In addition, movement really is two-dimensional. Thus, in a VANET context, these patterns are sometimes also called 2D patterns. Another important property of these scenarios is the presence of radio obstacles. While often neglected on highways, obstacle influence is an important factor in cities.

Communication Scenarios

Communication scenarios in *Vehicular Ad-Hoc Networks* can be grouped into safety-related applications and non-safety applications. The former create most of the motivation behind VANETs because governments, manufacturers, and customers are willing to spend money on technology to reduce injuries/fatalities on roads. Some safety-related applications and their respective requirements are listed in the following.

Electronic Brakelight The electronic brakelight alerts following cars to an intense or abrupt reduction of speed by means of communication. Consequently, addressing is usually geocast. Low delay is critical. Possible consequences for the receiving car are to notify the driver, to pre-condition the brakes by building up brake fluid pressure, or — in case of an unavoidable crash — precondition the passive safety system.[5]

Emergency Vehicle Approaching In this class of applications, traffic warning information is spread electronically that has been spread before via audio visual signals, e.g., by means of flashing lights and sirens. In modern cars, the sound insulation from the outside is getting stronger and stronger. Thus, such an application can help to transport a possible threat to the driver. In a slightly broader sense, this protocol could be extended to electronically spread the information contained in traffic signs.

[5] The term *active safety* is often used for technology related to accident prevention. Classical examples are brakes, road performance related gear, but also technology like night vision, radar, or communication enhanced perception boosting. In contrast, *passive safety* is technology to increase safety when a crash is unavoidable, such as the crumple zone, airbags, etc..

Road-Condition Warning With this application a car notifies others of current road conditions, e.g., a slippery road. Addressing is also geocast, timeliness is less critical than for the electronic brakelight. The reaction to such a warning would usually be a map-related driver notification.

Inter-Vehicular Collision Warning A substantial number of accidents occur at rural road junctions. Inter-vehicular collision warning tries to predict potential collisions by frequently emitting the car's own position. On receipt of these packets, the system can predict potential collisions and issue warnings. The communication requirements for this are usually single-hop only.

Cooperative Driving This class of applications is similar to the latter in terms of communication. However, the position updates are used to help one drive cooperatively, especially on highways. E.g., cars can automatically keep their distance to the vehicle ahead (this is sometimes called automatic cruise control, or platooning in case of trucks), or the system may assist in lane changes.

While the communication requirements listed above refer to packet-oriented data traffic, recent research drifts toward information-based data forwarding [266⋆]. However, in discussing packet forwarding algorithms, we will not focus on this here.

The second group of communication scenarios concerns the convenience-related non-safety aspects. Again, we list possible scenarios.

Vehicle-to-Vehicle IP Applications Put to this group are car-to-car applications that are basically working with a standard — usually unicast—`IP` stack. There could be (rear seat) chat applications between cars in the vicinity of each other. Here, one usually does not know by heart the `IPv4` or `IPv6` address of a car. Thus, lookup-services have to help to identify cars by more natural properties, e.g., the gray `BMW` with Mannheim license plates.

Internet IP Applications With the last set of application being car-to-car, this group concerns anything connected to the Internet. E.g., a car could access an Internet-based traffic management system of a city to reserve a parking place in advance. Since we are talking of an *Ad-Hoc Network*, this requires Internet gateways. As a business model, these gateways might be put up by roadside restaurants, providing connectivity coupled with advertisement. [82] talks about protocols for multi-hop gateway access. Online `VANET` applications directly compete with infrastructure-based systems like `GPRS` or `UMTS`. Hence, industry tends to provide Internet connectivity only at hot spots and then use single-hop only.

Virtual Paper Chase The name of this application surfaced in the FleetNet project and is a form of cooperative driving. Here, two or more cars hook up for the virtual paper chase, one setting the pace, the other(s) following. The leading car creates a mark on the map whenever it changes roads, so the following car can easily keep up, even if there is some distance to the leader, without directly seeing it. While this was created as a toy application, it seems to be useful when platoons of cars travel in loose formation to a common destination [111, 112].

Congestion Monitoring This application collects and aggregates location-dependent information about traffic density. It is information-driven and uses application-to-application single-hop broadcast. This application class is sometimes also called *Decentralized Floating Car Data* [111, 112], the most prominent application protocol being SOTIS (or Self-Organizing Traffic Information System [304, 303, 305].

Resource Constraints

A very nice property of VANETs is the availability of energy. Consequently, the design bottleneck of a VANET system is the usage of the radio channel and a desire to make these communication systems as cheap as possible. At this point in time, the cost of an on-board communication unit should not be higher than tens of dollars. VANET radio systems will most likely be 802.11p [1]-based, a special subset of the 802.11 wireless LAN standards group. The United States has already reserved a channel spectrum for vehicular safety between 5 and 6 GHz. This is likewise going to happen in Europe, although with less bandwidth.

2.2.4 Wireless Sensor Networks (WSNs)

When researchers believed MANET routing to be reasonably solved, they started looking for the next challenge. A classical engineering problem is to make things smaller or cheaper while conserving functionality. This, combined with advances in computer and radio miniaturization and battery technology led to the advent of *pWSNs* (WSN) and has created a stunning variety of possibilities and challenges [113, 99].

The main descriptive property of a WSN is the automated processing of digitized sensor information. A typical sensor basically consists of a battery and a small board containing the chips and the sensors. While in traditional sensing the actual sensor is wired to a computer, sensor networks operate by distributing many of these self-contained devices, letting them self-organize and thereby exploiting the spatial sensory diversity. A typical application is the distributed sensing of a vehicle outdoor and by means of acoustics. While one sensor might catch a vehicle's audio

signature, only certain sensors are able to locate it. Even more, using distributed signal processing they can also share the load of signature comparison.

An important design goal for WSNs is low energy consumption to maximize system lifetime. Mobility, however, is not a big issue in most of WSNs applications.

While it fits inside our definition of ad-hoc networking, WSNs have their own community and are usually treated separately from MANETs.

2.2.5 Mesh Networks

In principal, the MANET community provides protocols that deal with wireless networks that are mobile, or at least organize themselves without the need of central coordination. Mesh networks are networks that do not move, but rely on self-organization. Also, the temporal unavailability of nodes appears to routing algorithms similarly as node mobility would appear. Thus, the protocols used in mesh networks are usually class MANET algorithms.

The definition of Mesh Networks we are going to use here is an IP-based wireless network with rather low (mostly none at all) mobility. These systems include e.g, campus-wide rooftop networks. The main property distinguishing them from static rooftop networks is the self-organization. Mesh routing protocols should be able to deal with dead stations or even adapt to load changes or to a fluctuating radio situation.

Communication is usually unicast communication with TCP/IP, and energy preservation is not an issue. Often, the mesh network system itself or at least its antenna is on the roof.

The most prominent example of a rooftop mesh network is MIT RoofNet [16], an ongoing research project for a campus rooftop mesh network.

2.2.6 Network Classification

At this point, we emphasize that the classification we have introduced reflects only mainstream definitions. The multitude of researchers creates a multitude of special meanings. E.g., some people add the word “mesh” to all kinds of *Ad-Hoc Networks* or even algorithms. The words “wireless” and “mobile” are sometimes also added or omitted.

The MANET case studies presented represent only samples of the vast vector space of classification. We tabularize them in Figure 2.6. On the y-axis you will find the MANET types, and on the x-axis, the classification groups. This presentation focuses on the main descriptive properties, trading in completeness. E.g., we are aware that there are also WSNs that are highly mobile, like smart dust injected into a storm. However, we believe that WSNs generally face much lower, even static, mobility.

	Typical Scenarios	Node Mobility	Node Resources	Communication Requirements
Classical MANETs	Internet Extension/Cloud Battlefield Communication Desaster Relief	**Rel. Density**: any **Mobility**: 0 to medium speed **Pattern**: mostly unrestricted	**Energy**: minor issue **Actors**: none **Sensors**: none	**Addr. Scope**: unicast **QoS**: best-effort
VANETs	Active Safety Communication Convenience Communication.	**Rel. Density**: high **Mobility**: city / highway **Pattern**: city / highway	**Energy**: almost unlimited **Actors**: many **Sensors**: many	**Addr. Scope**: uni/geo/broadcast **QoS**: min. latency / best-effort
WSNs	Distributed Sensing	**Rel. Density**: high **Mobility**: low **Pattern**: random /physical	**Energy**: strictly limited **Actors**: none **Sensors**: many	**Addr. Scope**: anycast, info forwarding **QoS**: best-effort
Mesh Networks	Internet Extension	**Rel. Density**: any **Mobility**: zero **Pattern**: n/a	**Energy**: pract. unlimited **Actors**: none **Sensors**: none	**Addr. Scope**: unicast **QoS**: best-effort

Figure 2.6: Classification of MANET types

2.3 Link-Layer and Below

After having described MANET classes and their respective properties, we now dive into network architecture. Derived from the ISO/OSI reference model [161], most current protocol architectures are organized as stacks. As depicted in Figure 2.7, the lowest layer is the so-called physical layer. In a wired network, this comprises the wire itself and the means to transmit raw bits to a neighboring host (depicted by the solid blue arrows). Consequently, in a wireless network, the physical layer

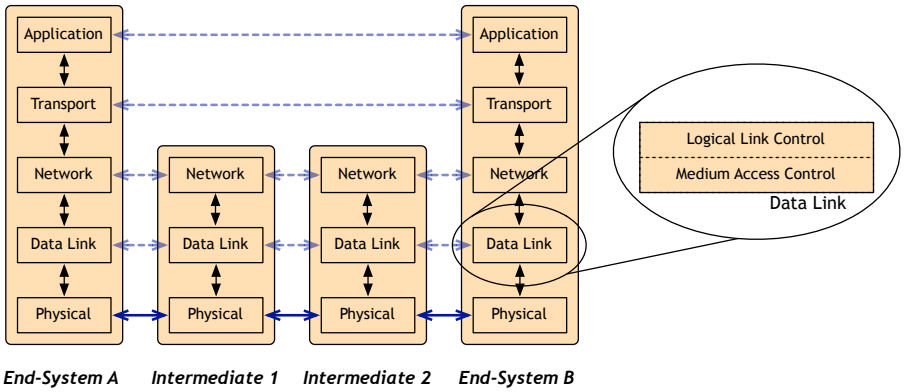


Figure 2.7: Protocol stack architecture

encapsulates the radio and the modulation. The next layer above is the data link layer. This layer logically encapsulates communication with our network neighbors. It enables communication to these neighbors *free of undetected errors* [284]. Moreover, it conceals transmission errors of the physical layer by mechanisms such as transmission repetition.

In networks where the physical medium is shared, i.e., more than one host might receive a transmission and two submissions are able to collide, the data link is divided into the logical link control (LLC) and the medium access control (MAC) sub-layers. While LLC performs the classical data link functions, MAC controls the access to the medium.

Let us now take a closer look at radio communication at layers 1 and 2. Since the EE part of radio communication is very complex, we will confine ourselves to basic assertions [249]:

- The strength of the received signal largely depends on the transmitted signal strength and on the quality of the antennas. It is lower, the further the receiver is from the sender. E.g., the so-called *free space propagation model* [249] describes the received signal strength $P_r(d)$ with respect to the distance be-

tween transmitter and receiver as follows:

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{2.1}$$

Here, P_t is the transmitted power, G_t/G_r are the antennas gains, L is the so-called system loss factor ($L \geq 1$) not related to propagation, and λ is the wavelength in meters.

- The receiver's ability to decode a packet depends on how strong the signal is compared to other signals, or noise. Consequently, whenever a receiver is simultaneously hit by two transmissions, the particular construction of the system determines which one of the packets — if any at all — can be decoded [107, 181]. Usually, the overlapping of incoming signals and the ensuing erroneous decoding is referred to as *packet collision*.
- Following the last argument, a single transceiver cannot simultaneously transmit and receive a packet.
- Besides its effect on the received energy, the frequency dominantly influences the ability of a signal to penetrate concrete matter. As a rule-of-thumb: The lower the frequency, the better the ability to penetrate concrete matter . E.g., radar [48] has a much higher frequency than wireless `LAN` because maximum reflection is desired here.
- Likewise, obstacles both reflect and absorb radio energy. This leads to multiple parts of the signal adding up at the receiver. Since these signals are time-shifted, they sincerely reduce the receiver's ability to decode the signal.
- The distance at which a transmission can be decoded is potentially lower than the distance at which the transmission might block other nodes from correctly decoding a concurrent transmission.

These simple assertions serve to help us to understand the radio channel for routing protocol development. In reality, this matter is much more complicated due to energy fluctuations at high frequencies, complex signal reflections on rough surfaces, Doppler effects due to moving nodes etc. However, as we will learn in Section 2.7, we require a certain level of abstraction.

After this quick introduction to radio propagation, we skip modulation techniques and proceed to medium access.

2.3.1 ALOHA

Developed in the 1970s at the University of Hawaii, `ALOHA` [59] is a medium access control protocol developed to enable multiple stations to communicate via a single

radio relay. In its pure form a station transmits whenever the need arises and listens to the back channel if the transmission was successful. If not, it waits for a random period of time and transmits again. When the transmissions from two stations overlap — even if only for a single bit — both packets will be garbled. While it is a simple and functioning approach to medium access, a pure ALOHA system suffers when many stations wish to send frequently. In fact, pure ALOHA systems can easily be choked.

A famous improvement to ALOHA is to divide time into packet-length slots (hence the term *slotted ALOHA* [252]) and to start sending only at the beginning of a slot time. This reduces the collision probability since packets can only collide during a complete slot or not at all.[6]

2.3.2 CSMA

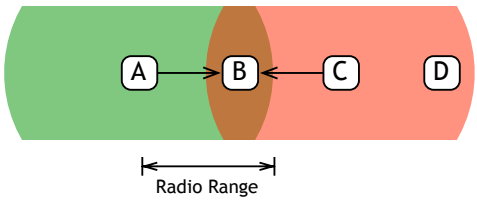
The next step in MAC protocol development is CSMA or *Carrier Sense Multiple Access* [178]. The key idea her is to listen before talk, i.e., to check if another node's packet transmission is in progress, and only start transmitting if this is not the case. While this class of protocols is slightly more complex than the ALOHA variants, it is more stable with regard to increasing packet load.

Pure CSMA protocols are classified according to their *persistence*, i.e., due to their behavior in seizing the channel just after it becomes idle: With 1-persistent CSMA, a station sends immediately when the channel becomes idle and waits for a random time before retransmission after a collision or in other words, 1-persisting CSMA continues to seize the channel immediately. On the other hand, non-persistent CSMA does not send immediately after the channel becomes idle but waits for a random time and repeats the algorithm.[7]

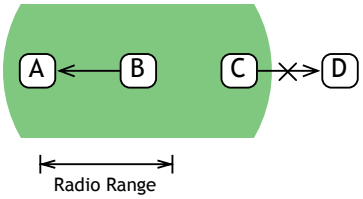
While CSMA increases packet delivery, especially under high loads, it cannot deal with the following two wireless transmission situations, the *hidden station problem*, and the *exposed station problem*. As depicted in Figure 2.8(a), the first problem occurs when the geometrical situation allows for node B to receive packets from A and C, but A and C cannot hear each other. When A transmits to B (the green circle depicting the reception range of the transmission), any transmission from C would destroy packet reception, regardless of the destination of C's packet. Since C is not in A's transmission range, CSMA will not be able to solve the problem. On the other hand, in the *exposed station* scenario depicted in Figure 2.8(b), node C senses B's transmission to A and does not transmit to D although neither packet would interfere at the receivers.

[6] For textbook explanations and a performance comparison of ALOHA and its slotted variant check [284, 190].

[7] There is another approach of CSMA called p persistence which we will not go into here.



(a) Hidden Station Problem



(b) Exposed Station Problem

Figure 2.8: Wireless transmission problems (as in [284])

2.3.3 MACA and MACAW

To cope with the hidden terminal problem in wireless networks, `MACA` (Multiple Access with Collision Avoidance) [168] lets the receiver send a short frame before the actual packet exchange in order to notify the nodes in the receiver's transmission range to remain silent for the duration of the actual packet exchange. Following Figure 2.9, the packet exchange works as follows: Station A wants to transmit a data packet to station B. To do that, it first sends an `RTS` (Request to Send) frame that is received by B and D, which is A's other neighbor. The `RTS` contains the data packet's destination (B) and a number denoting the time period the whole data transmission process is going to take. D reacts to the `RTS` by remaining silent for this time period, denoted by the red arrow between t_2 and t_4. B—since it is the destination indicated in the `RTS`—answers to the `RTS` with a `CTS` (Clear to Send) packet that denotes the remaining time for the packet exchange. This packet is now overheard by all stations in B's radio range (like `C`). In terms of the hidden station problem, these are exactly the stations that could interfere with the data packet. With the time contained in the `CTS`, nodes like C keep quiet for the rest of the transmission. Thus, A can safely send the data packet to B. In case of a collision, e.g., when two `RTS`s are transmitted simultaneously, `MACA` waits for a random time following a binary exponential back-off mechanism, i.e., the time interval from which the random time is chosen is doubled on each collision. This increases the scalability with regard to the number of stations.

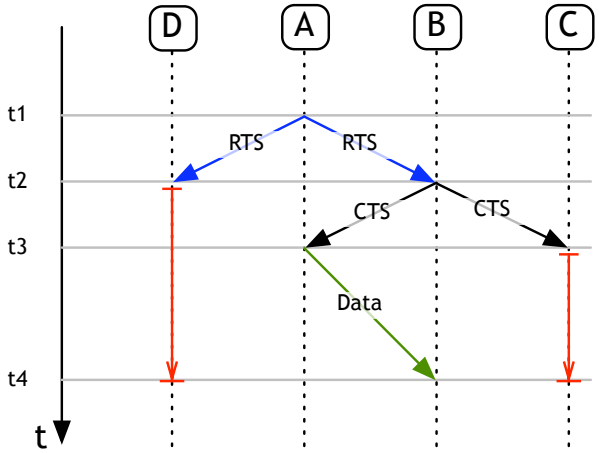


Figure 2.9: Multiple Access with Collision Avoidance (MACA)

Pure MACA uses RTS/CTS to cope with hidden stations. However, MACAW (MACA for Wireless) adds link-layer acknowledgments. [75] introduces this RTS-CTS-DATA-ACK scheme and shows the improved performance by means of simulation.

2.3.4 The 802.11 Protocol Family

Before coming to IEEE 802.11, we now take a moment to talk about IEEE 802.3 or Ethernet, as it is widely known. Ethernet [274, 35] is by far the most successful technology for wire-bound local area networks (LANs). The basic MAC algorithm is a variant of CSMA (see Section 2.3.2), and the supported LAN transmission modes are unicast, broadcast, and multicast. While originally designed for a shared medium speed of 10 $\frac{MBit}{s}$, it has been scaled to more-than-Gigabit speed, still using the same interfaces and the same basic algorithm.

When the need for a wireless LAN standard arose, IEEE started working on a CSMA/MACAW-based standard that was finally issued in the late nineties [17, 61, 1, 137]. An important concept was to "look (to the OS) like" Ethernet, enhanced by some necessary system management functions. This idea proved to be a key factor in the success of the protocol family compared to competitors like Hiperlan [246, 41].

Similar to Ethernet, the 802.11 family evolved over the years. Table 2.2 lists the sub-standards, their frequency bands, and their bandwidth modes. (Since the basic MAC protocols are the same, we will treat those sub-standards[8] equally throughout the rest of this work.)

[8] There are many more 802.11 sub-standards for different kinds of functions, like authentication, or stronger security.

Standard	Year issued	Frequency Spectrum	Bandwidth
802.11b	1999	2.4 GHz	$\leq$ 11 MBit/s
802.11a	1999	5 GHz	$\leq$ 54 MBit/s
802.11g	2003	2.4 GHz	$\leq$ 54 MBit/s

Table 2.2: 802.11 sub-standards

Standard	Channel Access	LL ACKs	RTS/CTS	Collision Avoidance
unicast	CSMA	yes	optional	bin. exp. back-off
multicast	CSMA	no	no	contention
broadcast	CSMA	no	no	contention

Table 2.3: 802.11 MAC variants

As the main mode, all 802.11 share the ability to connect to a so-called *access point* that serves both as a MAC coordinator inside the wireless LAN and as a gateway to the wired network. The main MAC access mode when working with an access point is the so-called PCF, or Point Coordination Function. However, since this thesis is about *Ad-Hoc Networks*, the far more interesting mode is the DCF, or Distributed Coordination Function, which enables stations to communicate without a coordinating instance.

The DCF mode for unicast transmissions is a variant of MACAW, i.e., CSMA with a binary exponential back-off scheme in case of transmission errors (potentially inflicted by packet collisions). Additionally, link-layer acknowledgments / retransmissions in combination with a CRC checksum [279, 28] help to conceal link-layer errors from the network layer.

Also, the DCF optionally offers MACAWs RTS-CTS-DATA-ACK packet exchange to cope with a hidden terminal situation.

In addition to these protocol features, 802.11 supports WEP (Wired Equivalent Privacy), a shared key cryptography [116] method to protect data transmissions. However, this method has been practically broken [84]. Thus, there have been extensions to the standard (802.11i) defining a better security system.

2.3.5 Other Ways to access the Medium

There are many ways to access a shared medium. E.g., in TDMA or Time Division Multiple Access [284, 56], a time period is sliced up into time slots, and these slots are assigned to certain stations. There are no collisions if the assignment algorithm is such that no two stations are given the same slot within their respective collision range, i.e., the range where transmissions can harm each other. However, precisely this coordination is the critical part of these protocols. In the German FleetNet project, a TDMA-based protocol called UtraTDD served as a candidate for a VANET

system [109, 211]. However, the main equipment supplier decided not to pursue the system further. Thus, the German VANET community turned over to the 802.11-based American system.

Besides CSMA and TDMA, there are many more fundamentally different approaches. The selection of the underlying link-layer protocol is highly connected to and thus greatly affects network layer performance [66]. However, because of the incredible dominance of the 802.11-family protocols even for VANET applications, the remainder of this work is mainly built on top of these kinds of protocols.

2.4 Fundamental Unicast Routing Strategies

As we have seen before, there are many types of *Ad-Hoc Networks*. Consequently, a routing algorithm that works well for one of them might not be very good in another one. In this section we talk about the major routing strategies, i.e., the fundamental means to find a route from a source to a destination. Every routing algorithm known to us can be seen as a mix of the strategies listed below and most of them have scenarios where they perform better than others. Hence, in the course of this section we will discuss the suitability of the strategies in certain scenarios. From a different perspective, these routing strategies are the toolbox of a MANET protocol designer.

In the course of this thesis, we will furthermore introduce a novel fundamental strategy, called *opportunistic* or *contention-based next-hop selection* to serve as a tool within that box. However, for Chapter 2, we will stick to strategies that were previously known.

The problem definition for routing is as follows: We assume a wireless (potentially mobile) network. For an arbitrary node tuple S, D of a source S and a destination D, we are seeking a (directed) path, i.e., an ordered sequence of nodes $N_1, \ldots, N_I$ transitively connecting S and D. E.g., in Figure 2.10, a desirable path would be $S \rightarrow 1 \rightarrow 4 \rightarrow 5 \rightarrow 7 \rightarrow 8 \rightarrow D$. Usually, we do not even need the whole

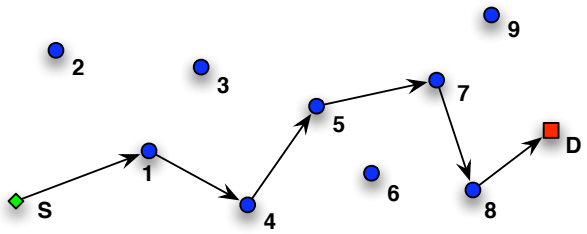


Figure 2.10: Routing example

sequence but only the so called next hop, i.e., assuming we are somewhere on this

chain, the one node to which we are going to hand over the packet for further forwarding. In the following, we will call any node handling a packet (except the packet's destination node) a forwarder. The path from S to D will be called a route.

We note here that prior to CBF (see Section 3.3), one of the main contributions of this thesis, the next hop was computed by the routing algorithm and then the packet was sent to this next hop explicitly. For the remainder of Section 2, we will examine previous work following that mechanism.

2.4.1 Topology-based Strategies

The fundamental abstraction used to apply topological strategies is to view the MANET as a (usually undirected) graph $G(V, E)$ with the set of nodes, or vertexes V, and the set of edges, or links E. Due to the physical presence of links, this very much reflects the reality in wired networks. In wireless networks however, two nodes 'share links' if they can successfully receive packets from each other. While edge weights are possible, they are mostly set to one for MANET routing, since the number of hops — or packet retransmissions — is usually believed to be the decisive cost factor.

After having constructed this graph, we can apply the power of graph theoretical algorithms [104, 96]. E.g., a breadth-first search from a node S in an undirected and unweighted graph gives us the shortest routes from S to all nodes in the network. If the edges are weighted and directed, Dijkstra's algorithm [96, 32] does the trick. Well known from the wired world, routing protocols based on topology are classically separated into two classes, namely into *link-state* [284, 43] and *distance-vector* [284, 34] methods.

Link-state routing is the approach where an algorithm like Dijkstra's is applied to the whole network topology, meaning that every node has to be made aware of the whole topology by means of packet exchanges (due to this reason, it is sometimes also called *full-topology routing*). E.g., one node sends a packet containing his set of radio neighbors to everyone else in the network, and every node records these links upon reception. In a connected *Ad-Hoc Network* with N nodes, this would cause at least N retransmissions of N packets, because every node sends one packet which is repeated once by every other node. Thus, the number of packet transmissions[9] is in $O(N^2)$. Once the information has been gathered, a complete route to the destination can be obtained with one of the above-mentioned algorithms, and the packet is simply sent to the next hop on the route. Link-state routing is used in many places, the most prominent being the Internet's OSPF or *Open Shortest Paths First* routing protocol [226].

[9] For details to the $O()$ or Big-Oh notation check [141, 96]

Distance-vector routing, the second important routing strategy, works slightly differently. Instead of collecting information about the complete network topology, nodes build distance-vector tables for every possible destination id, i.e., they record the hop distance h to every node D by evaluating the hop count of passing packets.[10] So whenever a packet passes by we know (a) which neighbor n_i has sent it to us, (b) the packets origin S, and (c) the number of hops h it took the packet to reach us. Thus, we assume that whenever we want to send a packet to S ourselves, n_i knows a next hop on a route to s with $h-1$ hops. Distance-vector routing's most prominent representative is the Internet's *Routing Information Protocol* or `RIP` [214]. The basic algorithm can be found in [69].

Another method for routing based on topology is the so-called *source routing*. Here, the source of a packet puts the packet's route inside its header as an ordered list of traversed nodes, so nodes can extract partial routes from passing packets and put them into their local routing tables. As for the computation of the route, either full-topology routing can be used or a route request scheme as in `DSR` (see Section 2.5.4) .

Both distance-vector and source routing protocols also create $O(N^2)$ packets when we want every node to be able to send packets to every other node. In the case of source routing, it is slightly worse since the packet header's contents grows at each hop. When assuming that its growth is only limited by N itself, the complexity of the bytes transmitted is even $O(N^3)$. Considering the size of the packets for link-state routing would bring its complexity to $O(K \cdot N^2)$ where K is the average number of radio neighbors.

Discussion We list some well-known fundamental advantages and disadvantages about the above-mentioning routing strategies: Link-State routing offers the advantage that when one link breaks down, every node is capable of calculating alternative routes, whereas in distance-vector routing one has to wait till incoming information establishes a novel route. This is slightly better with source routing, where we at least have a chance to reconstruct alternatives from the obtained route fragments. However, source and distance-vector routing protocols have the advantage that a rather limited view of the network is enough for two nodes to be able to communicate, whereas link-state requires complete topological knowledge, at least about a part of the network.

[10] *Hop Counting* is a simple network protocol mechanism where a counter starting at zero is included in a packet header and is incremented at each retransmission. This allows to record how long the packet has gone so far. Alternatively, this counter starts at a certain value and is decremented. In most protocols there is a certain limit to the hop count, resulting in packets to be discarded, when the limit is reached. In the Internet, the corresponding field is called *Time-To-Live* or `TTL` [244].

Inherent Problems of Topology-Based Routing The most imminent problems of topology-based routing for MANETs is the temporal instability of the neighborhood, or "sharing a link" property of two nodes. I.e., whenever we receive a packet from a node, we cannot safely assume that we, too, can reach it, and even if, in the presence of mobility, the temporal validity of the node's radio neighborhood is even more reduced. So if communication is long-term, it is very likely that re-routing will be necessary, creating transmission overhead.

Over time, the community has attended to most of these problems by creating adaptive and highly specialized routing protocols to deal with all kinds of problem scenarios pushing topology-based routing for MANETs to its limit. We will further discuss some of the state-of-the-art protocols, as well as their solutions to the problems, in Section 2.5.

2.4.2 Position-Based Strategies

An obvious advantage of *Mobile Ad-Hoc Network* topology over wired ones is the strong correlation of network neighborhood with the position of the nodes[11]. Hence, an obvious strategy for a source node S with its known position to find an arbitrary node D with position is to bring the packet physically close to D. Following a greedy heuristic [40], this global goal can be approximated by handing the packet over to some neighbor n_i^S out of the set N_S of S's neighbors. Formally, the set of greedy neighbors $N_S^{\text{greedy}}(D)$ of node S with respect to destination D is given as.

$$N_S^{\text{greedy}}(D) = \{n \in N_S \quad \wedge \quad \text{dist}\,(n, D) < \text{dist}\,(S, D)\} \tag{2.2}$$

Hence, the set of greedy neighbors of S with respect to a given destination D is the subset of S's neighbors that are geographically closer to D than S itself ($\text{dist}(A, B)$ is here the Euclidean Distance [36] between A and B). In a network with sufficient node density, any strategy that selects a next hop out of this greedy set will eventually lead to the radio neighborhood of D, from where it can be easily reached. We call every algorithm of this class *greedy position-based routing* or simply PBR. The main advantage of this strategy is that it operates mainly on local information about the neighborhood. The only non-local information needed is the destination's location. This strategy is indifferent to changes in the link structure along the path, as long it always finds a greedy neighbor.

To explain the problems encountered by a greedy routing heuristic, we want to emphasize its similarity to free space navigation of ships or planes. Namely, in an *Ad-Hoc Network* with reasonable node density the choice of a next-hop is analogous to the selection of a navigation reference point [45]. Similar to PBR, this method

[11] The simple *Unit Diskgraph Model* (UDG) assumes that two nodes are able to communicate, whenever their Euclidean distance is smaller than a threshold called the radio range.

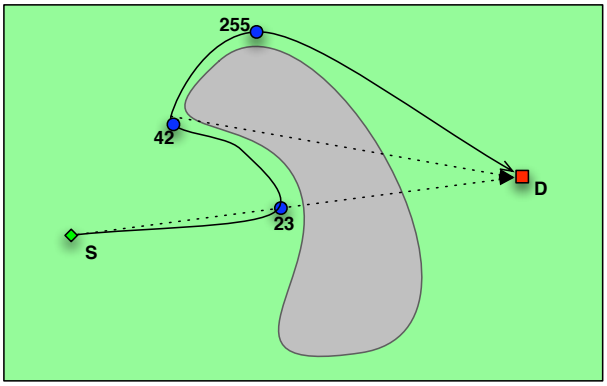


Figure 2.11: Void situation in position-based routing

requires that (a) one knows one's own position relative to some reference system, (b) one knows about the position of one's destination and (c) one should be able to assume that the line of sight between the own position and the destination is not blocked. E.g., when Christoph Columbus [25, 272] tried to find India by traveling West, he made this assumption ignoring the presence of America, and he falsely assumed he had reached his destination. However, had he wanted to reach America, and had he perfectly known the position, there would have been no better way to go there than by means of position-based navigation. Also, while this anecdote seems trivial, it perfectly reflects a main problem of position-based routing in *Mobile Ad-Hoc Networks*. Let us assume that every node knows its own position and the position of the destination node. Let us also assume that there is a valid path to our destination. Still, routing on the basis of position would fail if we would hit an area on our projected path without nodes. Figure 2.11 shows a scenario where position-based routing would fail without the geometrical formation of the network. The green area marks the extent of the network. Here, we assume to have an adequate number of nodes to allow for packet forwarding. Out of the many nodes, S and D mark the source and the destination. The usual approach of free-space navigation is to draw a line segment (the dashed line) from S to D and follow this line or — in terms of networking — use forwarders along that line to finally reach the destination. However, this fails at the blue node when progress to D is blocked by the gray void formation (at node 23). Nevertheless, there is a heuristic that still will find the destination: Let us be a ship again and assume we turn left, keeping the void in the vicinity to our right hand, and just follow. Additionally, we keep our sight directed to our destination and follow this line, whenever possible, i.e., whenever they do not lead us into the void. At node 42, we still cannot follow the direct path, but at node 255, we are safe to follow it again, allowing us to reach D. While this is very intuitive, it is a bit more complicated when the void is just "the

absence of nodes", and keeping the void to our right hand is also a bit complicated to formalize. (Sometimes, an algorithm within position-based routing that trades in single-hop greediness to find a non-greedy route is called *recovery strategy*.)

The problem of voids in position-based routing has already been solved for *Mobile Ad-Hoc Networks*, allowing for position-based forwarding under the following assumptions:

1. Each node has the means to acquire its own position relative to a coordinate system.

2. Nodes proactively send beacon packets containing their own position, allowing every node to keep track of its direct neighbors and their respective positions.

3. There is a way to acquire the position of every node in the network, even if it is not our direct radio neighbor.

As for item 1, modern technology enables every outdoor node to assess its current position by means of comparison of satellites emitting very precise time signals. The most available and prominent example of this technology is the Global Positioning System `GPS` [167, 39]; another effort is Galileo [38]. This technology is accurate to within a few meters and is inexpensive to buy. While this technique is largely limited to outdoor usage, various indoor positioning systems do exist [154].

As positioning is available, item 2 can be easily accomplished with a constant number of packet transmissions per time ($O(1)$). Unfortunately, this does not hold for item 3. Here, distributed algorithms have to be used that scale worse due to their non-local information they have to acquire. However one can always ask everybody in the network to give us its current position, which would create $O(N^2)$ packets if everyone wants to know the whereabouts of everyone else. The class of algorithms solving the "acquire location" problem are usually called *location services*. Some prominent examples are described in Section 2.5.

Discussion Even the simple example above shows that the efficiency of position-based routing largely depends on the type of network we are dealing with. In case of densely and evenly populated scenarios, we expect the *line-of-sight* heuristic to be very successful. In the case of a void situation — which is more likely to occur in sparse networks — routing becomes more difficult and can become very ineffective, e.g., when turning south would have been shorter than turning north. But the dominating advantage is to be able to route without global knowledge of the network and (almost) regardless of local changes in the neighborhood situation as long as there are enough suitable candidates for forwarding. E.g., [171] shows that a position-based routing method outperforms a prominent topology-based one, especially in dense and highly mobile settings.

2.4.3 On-Demand vs. Pro-Active

While topology vs. position is the dominating differentiator throughout this thesis, an orthogonal property with a major influence on routing protocol complexity is the notion of *on-demand*, or *reactiveness* vs. *proactiveness*.

Following a proactive routing scheme, each node maintains routes to every possible receiver node in the network, even if it is currently not communicating with them. In a strictly reactive scheme, however, a node only starts acquiring routing information about a certain destination when it is told to send a packet there. In rather static systems like the Internet, the method of choice is traditionally proactive since the overhead of the protocol is rather low compared to the transported load. Also, nodes tend to communicate with a varying communication partners around the globe, and more importantly, the delay of acquiring a route in the Internet for each connection would have been unacceptable.

So, when Internet routing was transferred to `MANET`s as with algorithms like `DSDV` [240], the assumption was that a routing protocol should provide routes just as those in the Internet. However, this uncovered a major problem. Given a network with N nodes running a proactive routing protocol and facing different levels of mobility, the routing has to keep up with the changing links. Thus, one can find a mobility borderline where a proactive routing algorithm itself completely chokes the network. However, it seemed to be unnecessary to stock up possible routes to everyone. Moreover, communication in an *Ad-Hoc Network* seems to be rather local and focused on few communication partners. Consequently, on-demand methods were designed with the property of only acquiring a route to a destination when required.

Discussion Following recent research, it can clearly be seen that people focus on on-demand protocols for `MANET`s due to the reasons given above. However, stating that on-demand is superior to proactive is clearly not true. In contrast, the more nodes in a network tend to communicate briefly with many different partners, the better a proactive algorithm potentially performs.

2.4.4 Soft-State vs. Hard-State

The next major design aspect in protocol engineering is the question of soft- vs. hard-state [164]. In principal, in a hard-state protocol, all local states are set or unset explicitly. E.g., a node would tell another that it is now its neighbor and would also tell it when this is no longer the case. Thus, hard-state protocols are sometimes called to use *explicit signaling*. Soft-state methods, however, usually use *set + time-out* rather than *set till unset*, i.e., in our neighborhood example, one neighbor would notify the other about its departure.

Figure 2.12 depicts the example. Here, we regard the state of some node being our

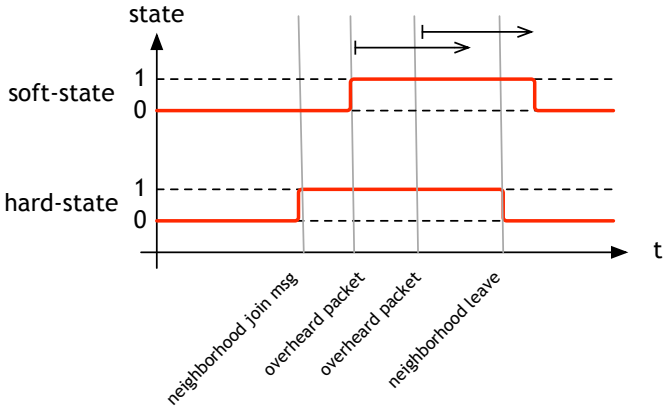


Figure 2.12: Soft vs. hard signaling

neighbor (0 meaning that it is not and 1 meaning that it is). The x-axis denotes the time, listing some events. The red line is the hard-state protocol; the green line denotes the soft-state variant. In a strictly soft-state protocol, neighborhood association is set explicitly. Consequently, the hard-state protocol sets the state to 1 when such an explicit neighborhood packet arrives and keeps it until the state is explicitly removed. In contrast, the soft-state variant sets its state based on implicit knowledge, such as an overheard packet. When hearing the packet, it also sets a time-out (the black arrow), reflecting the time a neighbor from whom we have received a packet is believed to stay in our radio range. On every subsequent packet heard, this timer is reset, keeping the state. When the timer expires, the state is reset.

Discussion The advantages of soft-state signaling for the listing of a node's neighbors are obvious because they take the neighbor's absence, i.e., the failing to hear, as an indicator that it is no longer a neighbor. However, it trades in inaccuracy over time, since the neighbor has usually moved out of range before the timeout expires. Here, the closer the soft-state timer settings match the behavior of the system, the better hard-state signaling is approximated.

In this context, it is favorable to (re)use any received information, e.g., even a packet not destined for us can be used as an indicator to node neighborhood, or it can help us implicitly construct routes by exploiting overheard header information.

2.4.5 Cross-layering

In stack-like reference models like ISO-OSI or the TCP/IP stack, every layer has its clearly separated domain of responsibility. The fundamental idea behind these architectures is to pass data and control information to the layers above and below using interfaces that are as abstract as possible, therefore allowing for a high degree of freedom of implementation inside the layers. E.g., the link-layer should get data from the upper layers and should only be told the neighbor(s) to which the data should be transmitted. Following this strict abstraction, the upper layers do not need to care, or even know if the link-layer is 802.11 or Hiperlan, some infrastructure service, or a wired Ethernet. Global networks like the Internet absolutely require that layers be treatable in an abstract manner to allow for easy integration of protocols.

However, when resources are low, it is quite clear that a protocol can be optimized when the implementation details of adjacent layers are known and exploited. In terms of decision theory [267, 106], strict layering is calling for separate optimization of each layer. However, when finding a global objective function and allowing for each layer to sacrifice local performance for global optimality, the overall system performance can be improved.

Cross-layering will be an issue for most of the rest of this thesis. Whether it be just accessing the internal state of an adjacent layer or even the combination of channel access and forwarding, it is extensively used for our protocol designs. However, we are aware of the trade-of in cleanness and separability, and we will discuss the inherent trade-offs.

2.4.6 Purity vs. Hybridity

Summarizing dozens of algorithm proposals, one can safely say that the early proposals were pure and different, and that over the years the methods became more hybrid, combining strengths and alleviating weaknesses. E.g., reactiveness might be a good idea for nodes that are many hops away, but if communication is usually two-hop, then a hybrid approach with proactive local routes and reactive global routes might make a lot of sense. The same applies for position and topology. As we will see in Section 2.5, there have been proposals to enrich topology-based protocols by the use of position information and vice versa.

So another standard principle in protocol engineering is to understand which protocol features interact with which performance effect under which circumstances and — if possible — detect these circumstances and select a protocol variant that performs best. E.g., a protocol could perform in a topology-based manner only as long as position information is not available, and try to exploit position information as soon as a GPS is added to the system.

2.4.7 Caching vs. Discarding

A method very well known from computer science is caching [22, 70]. In principal it is the re-use of previously computed — or to be more general — acquired information. E.g., route requests can be recycled to extract (partial) routing information to other nodes or to extract location information about a neighboring last hop. Usually, caching always offers a trade-off because the re-use of previously extracted and cached information helps only then, when this information is stable enough over time. Otherwise, a new and time-instant information acquisition would be more useful.

2.4.8 Evaluation Criteria for Routing Protocols

While we have discussed design alternatives, we have yet to talk about the evaluation criteria of routing protocols, i.e., in terms of decision theory, what the structure of the objective function is like. Like many real-world problems, routing performance is a multi-goal optimization problem. In other words, you cannot easily order routing protocols absolutely on one performance scale. Moreover, even for each of the criteria it is hard to decide if protocol A is better than B. E.g., when trying to find out whether protocol A or B shows a higher resilience to mobility, one would set up simulations with different mobility settings and would estimate performance indicators for every setting. However, due to the complexity and the adaptability of routing protocols, the behavior could swap order along the way (see Section 2.7). In principle, two routing protocols can be compared only when they are told to solve the same problem, i.e., when they have to work in the same setting. Ultimately, this would mean always having to simulate all protocols for every scenario and then compare all criteria.

Still, while this is rarely feasible, we can generate interesting statements by analyzing and comparing some opponents in some settings. Since routing evaluation is a problem that requires some standardization, the `IETF MANET` group defines some criteria in [98].

For the remainder of this thesis, we will judge routing performance mainly by the property of how reliably and how quickly a protocol can deliver packets to a random destination. These properties being on the objective side, we will simultaneously look at the cost in terms of transmitted packets/bytes. In most important studies, cost is measured at the bottom of the network layer. However, this ignores the fact that a “bad” routing protocol might select neighbors with bad connectivity, resulting in frequent link layer retransmissions. Thus, we measure cost in terms of packets/bytes on the channel rather than packets/bytes the link-layer was told to transmit.

2.5 Selected Algorithms

While we have discussed fundamental strategies in the last section, we will now discuss some selected routing methods for *Mobile Ad-Hoc Networks*, most of them following concepts described in Section 2.4. The first, flooding, is the simple approach to reaching a destination, principally exploiting topology, i.e., the fact that a broadcast in wireless networks simultaneously reaches each of a node's neighbors . We explain it first because it is also a building block for most of the subsequent methods. Sections 2.5.1–2.5.5 describe topology-based routing methods; Section 2.5.6 describes location services as a building block for position-based routing and Sections 2.5.8, and 2.5.9 describe position-based routing approaches.

2.5.1 Flooding

Flooding means to resend duplicates of a data packet on all outgoing links except for the one on which the packet was received [284]. To bar the packet from circulating in the network forever, a time- or hop-based expiry mechanism is used.

Transformed to a wireless context, flooding is consequently a simple re-broadcast of a packet, the most simple version depicted in Figure 2.13. The gray box depicts

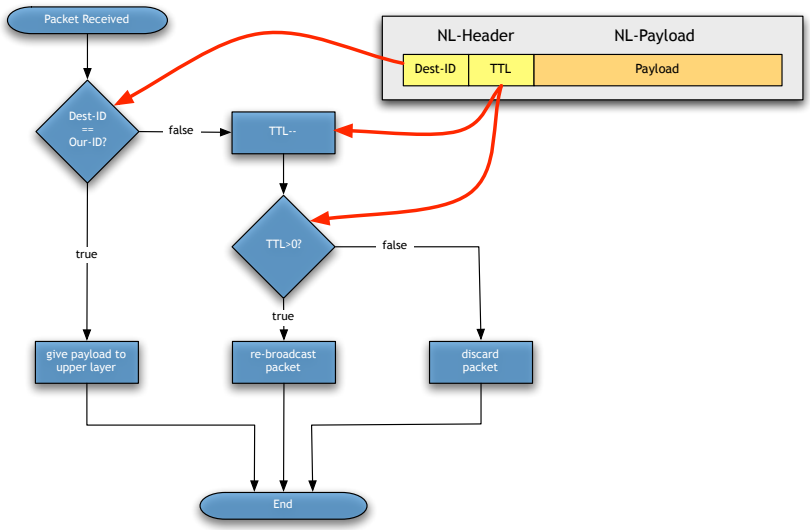


Figure 2.13: Flooding as a routing algorithm

the message format and follows the usual notation for packet headers [284]. From left to right, the packet parts correspond to higher layers, i.e., the leftmost header belongs to the link layer (the physical layer's preamble is intentionally left out). The picture, however, starts with the network layer header (yellow), because the

link layer header has been stripped before the packet was handed over to the routing entity. Hence, the flow chart entry "Packet Received" is the entry point at which the link layer hands over the packet, the left-most remaining header being the network layer header.

The header itself consists of two header fields[12]. The destination `ID` field contains a number uniquely identifying the destination node of the message.[13] The second field, called `TTL` or time-to-live, is a non-negative integer number denoting the remaining number of hops the messages is allowed to travel. The initialization of the transmission process is by building such a packet, setting the destination `ID` field to the message's destination, and the `TTL` field to a positive integer. Then the packet is transmitted to all neighbors by means of link-layer broadcast. When this packet is received on an arbitrary node, the process depicted in Figure 2.13 kicks in. First, the receiver checks whether it itself is the final destination of the packet. If it is, it simply hands the packet over to the upper layers. If it is not, it decrements the time-to-live value and checks whether it has become zero, which would mean that the packet's lifetime is over and it is to be discarded. If the `TTL` is still positive, the packet is rebroadcast; the initial `TTL` value determines the maximum hop range of the packet, i.e., if a node is more than this value in hops away, it can not be reached.

Figure 2.14 illustrates communication from S to D using flooding. As outlined above, S receives the payload and the destination — in this case D—from its upper layers. Then it builds a packet containing this information plus the default `TTL` and broadcasts it (denoted by the grayish circle with `TTL`5). This packet is received by Ss neighbors *1* and *2*. Following the simple flooding algorithm, they both check whether they are the destination, which they are not, reduce the `TTL` value by one and re-broadcast (the two blueish circles with `TTL`4). Apparently, *1*'s transmission is overheard by D, which hands the packet to its upper layers. The end-to-end data exchange is accomplished.

Discussion While this simple example illustrates the algorithm, it also uncovers possible problems: First, considering the second round of transmissions both also reaching S. With this algorithm, S does not know it has already transmitted the packet before and does so again until the `TTL` tells it otherwise. This applies for every node; e.g., every neighbor pair will send copies of the same flooded message to each other until the `TTL` expires. Second, how will all the other nodes know of the

[12] A header field is an area of bits in the header with a corresponding encoding, i.e., it can be accessed as a computer variable.

[13] For the remainder of this work, we will assume that every station receives such a unique `ID` when manufactured, as in wired Ethernet cards. We acknowledge the fact that there are methods for providing locally unique identifiers [297]. However, integration of these methods into our routing schemes is out-of-scope of this thesis.

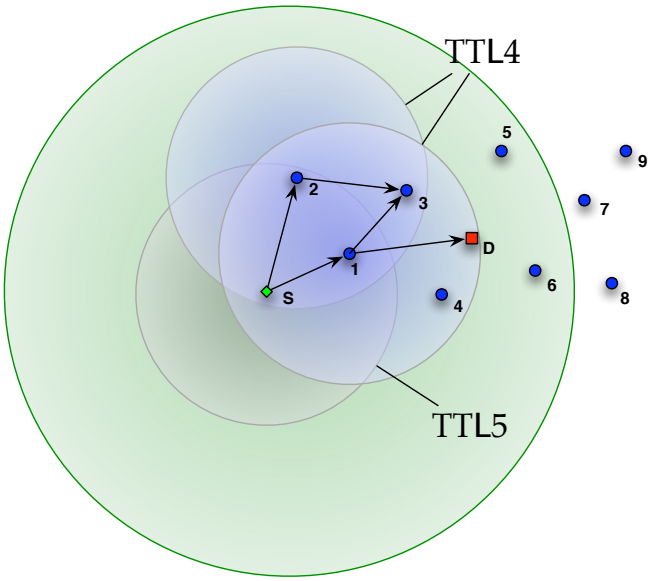


Figure 2.14: Flooding example

packet reaching its destination? The answer is that the packet will always circle during its TTL, no matter after how few hops the destination is actually reached. The third problem originates from the necessity to access the radio channel: Whenever one flood packet reaches more than one node, these will almost simultaneously try to re-transmit, causing an almost-simultaneous contention on the channel. Moreover, the next round will again (exponentially) increase the simultaneous attempts at channel access. Considering that usually the interference range of a single transmission is higher than the range in which it can be successfully received, channel contention literally explodes. The effect described above has been described as the *Broadcast Storm Problem* [231]. In the example, the greenish circle around S depicts the interference range of a transmission from S as modeled with the ns-2 (see Section 2.7) free space radio propagation model of roughly twice the radio range. This circle covers all relevant nodes, meaning that there can be no simultaneous transmission that can be received successfully. Another problem of flooding — especially in the context of MANETs — is how to set the TTL. A long TTL allows for a big network but also creates many redundant retransmissions, especially in small partitions.

Most of the problems above have already been addressed in the literature. E.g., the introduction of packet (payload) IDs allows the network layer to identify packets and thus to guarantee to re-transmit a packet only once. To do this, it stores the IDs of all packets retransmitted and discards such a packet whenever it receives it again.

The broadcast storm can be alleviated by the introduction of a random timer, i.e., every node waits for a random period of time before re-sending the packet. Thus, the local channel access explosion can be distributed over time. The TTL problem is often attacked by using an expanding ring search technique, i.e., first, a flood is started using a small TTL like 2. Then, if the node does not answer, the same is repeated with a higher TTL, often following linear or even exponential schemes. On later packages, a destination-adapted TTL can then be used. This mechanism requires a density and mobility-adapted setup but achieves some scalability when communication is mainly local. Some flooding improvements are discussed and evaluated in [302]. Moreover, any algorithm using flooding as a building block (like some of the methods discussed later) also has some flavor of "optimized flooding".

However, redundancy reduction also has its downsides. Considering a real radio channel with fluctuations and random errors, possibly combined with a sparse network. Then, redundancy could be highly welcome, especially because radio broadcasts are usually unacknowledged so failure is invisible.

Sophisticated Flooding Techniques Since this technique is so fundamental to all kinds of routing methods, there has been a lot of research on flooding, with emerging new methods all offering some performance gain by, e.g., trading in flexibility with regard to mobility. One class of methods tries to create a network backbone, i.e., a (minimum) dominating set of nodes [33] that can reach all other nodes. The idea is that controlled flooding over this set minimizes redundancy without losing the ability to reach every node. Since this graph-theoretical problem is NP-complete even with global knowledge, the distributed network algorithms rely on heuristics, e.g., [191].

2.5.2 Destination-Sequenced Distance-Vector Routing

The first example of a "real" routing algorithm, i.e., an algorithm obtaining route information and using it for efficient unicast data transport is *Destination-Sequenced Distance Vector Routing (DSDV)*, one of the earliest MANET-adapted routing methods stemming directly from distance vector routing as known in the Internet (see Section 2.4.1). In this algorithm, every node proactively broadcasts its table of distance vectors, i.e., the table containing the hop distance D_i^{recv} for destination node i. On the receipt of such a packet, the receiving node compares for every destination its own distance to every destination with $D_i + 1$, which is the one hop needed to reach the node having the distance vector. If $D_i^{\text{recv}} + 1 < D_i^{\text{own}}$, then the new value is entered and the node we have received the update from is marked as the next hop for this destination.

While the algorithm fragment described above would hopefully converge in a stable network, it would not be able to handle routes that "get worse". DSDV offers a

clever two-part sequence number mechanism to handle this. First, whenever a node starts a broadcast of its own routing table, it includes an increasing even sequence number. Thus, nodes can distinguish between newer and older information. To be more precise, the comparison outlined above is enriched by the comparison of the sequence number, preferring the higher over the lower, regardless whether the metric is worse. The second part are the odd sequence numbers; they are used to mark broken links, i.e., if a node on some DSDV route cannot transmit a packet to the next hop on the route, it marks this distance vector with a hop distance of ∞ and uses the next (odd) sequence number. When this message is included in the node's route update, any receiving node that would also use the same hop can disable the corresponding route entry, being enabled again by the next route update from the destination running the next even sequence number.

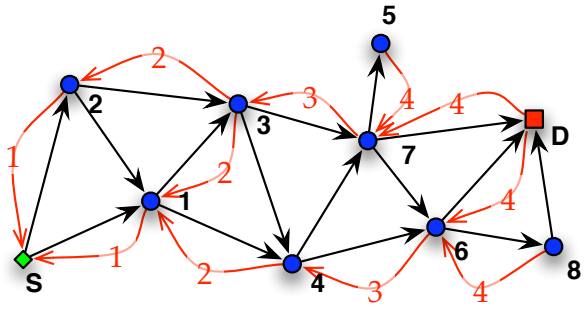
Thus, the basic advantage of DSDV over former wired protocols is the faster propagation of link breaks and worse routes, making for the "Highly Dynamic" in the original paper title [240]. Also, it is capable of letting every node know which nodes can be reached, i.e., reside in the same network partition, which is a tremendous time saver as opposed to the reactive protocols below which may start an expensive lookup process for nodes that cannot be reached. While DSDV has been somewhat overtaken by later proposals, it is still the original pro-active method and is used by research groups that look into routing metrics [101, 102].

2.5.3 Ad-hoc On-Demand Distance-Vector Routing

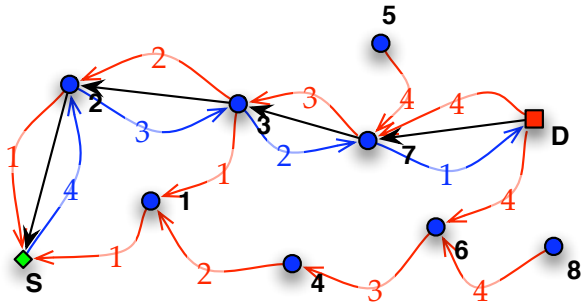
Ad-hoc On-Demand Distance-Vector Routing, or AODV, is DSDV's reactive sibling; i.e., if no packet transport is requested, the protocol does not maintain routing information and, on request, it only provides information about how to reach the communicating nodes. Thus, the main difference from DSDV is the algorithm to create routing information, which works as depicted in Figure 2.15. During the route request, the destination node starts flooding the network with special AODV routing control packets denoting an AODV route request. Basically these packets contain the source and destination IDs and a TTL. Whenever a node receives such a packet (the transmissions are denoted by the black arrows), it adds an entry to its table of distance vectors, so after the first packet, nodes *1* and *2* know that S is directly reachable, i.e., only one hop away; after the second round of packets, nodes *3* and *4* know that S can be reached in 2 hops via node *2* and *1*, respectively .[14] So after this route request cycle, all nodes that got the route request packet have a route to the initiator of the request.

The second part of route creation is the *route reply*. It is initiated by the request destination when it is reached by the request cycle. It then stops re-broadcasting

[14] It depends on the implementation, if node *3* takes *2* or *1* as a next hop to S, since they are equally suited.



(a) Route Request



(b) Route Reply

Figure 2.15: AODV route setup

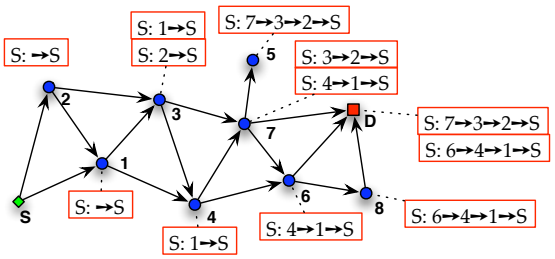
and sends a route reply to the request originator by creating a packet containing the sender id and broadcasting it to the next hop on the route to it, i.e., the smallest distance vector it has got. In the example there are two equally good routes so it just picks one, e.g., node *7-3-2* finally reaching *S*. En route, all nodes set their respective distance vectors to *D*, using the route reply's TTL field. After this, there is a bi-directional route connecting *S* and *D*.

This only covers very basic routing. Actually, AODV is a lot more complex, employing DSDV's sequence numbers for route repair combined with route solicitation messages making the sender initiate new request cycles. In fact, the development of AODV reached enough maturity to acquire IETF RFC [49] status [239]. It is now of the most common reactive routing methods, numerous protocol enhancements have been developed, such as, e.g., scenario-specific protocol adaption [83], a better support for unidirectional radio links [217] or geocast enhancements [268], and even includes multicast operation [255].

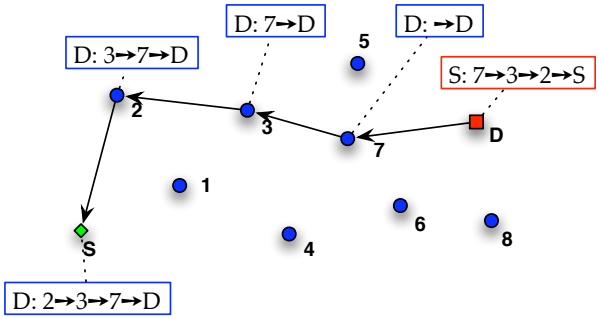
As required by the IETF, there are also numerous implementations of AODV [212, 256].

2.5.4 Dynamic Source Routing

As high in popularity as AODV is *Dynamic Source Routing*, or DSR [165]. The basic mode of operation is quite different from that of AODV. However, the route setup can also be separated into a request and a reply phase. Figure 2.16 displays its functioning: The DSR route request also starts with a flooded request packet



(a) Route request



(b) Route reply

Figure 2.16: DSR route setup

containing the destination ID. However, instead of creating distance vector tables, it records the explicit route inside the request packet's header. Technically, the request forwarders do not need to record this route inside their local routing table but do so for caching reasons. The route caches created by the request circles are represented by the red boxes. Multiple red boxes per node represent alternative equal-length paths. The destination, when reached by the request, is provided with a full-fledged list of hops to the source, which is used to route the route reply and any unicast packages. Consequently, whenever a packet is sent from D to S, the node list $S \rightarrow 7 \rightarrow 3 \rightarrow 2 \rightarrow S$ is included in the packet header. Intermediate nodes then do not consult their local routing tables but just execute the packet's predefined

path. Thus, when the route reply reaches S, the reverse routes to D are set up and can be used for unicast packets.

As outlined in Section 2.4.1, source routing's general advantage is the control and knowledge it provides over the whole route. Also, there is a smaller state requirement in the interim nodes because the whole information needed for routing is contained in the header. The main disadvantage is a node list inside the header that has to support a size of at least n node IDs, where n is the maximum number of hops as opposed to one ID with in AODV).

Having been extensively researched, DSR also offers a huge variety of enhancements that are hard to keep track of. [160] exploits the source routing property for security, while [159, 248] gets rid of the header lists by using path labels. [273] improves DSR by the usage of two-hop neighborhood information to optimize routes; [158] looks into the effect of caching strategies. Also, the corresponding multicast strategy has been described [197].

Since the competition between DSR and AODV needed to be put into perspective, numerous studies [247, 87, 247, 215] evaluate and discuss AODV and DSR, at least as an opponent for new routing methods. In our own studies and work tutored by us [135⋆, 133⋆, 126⋆, 125⋆, 192′, 209⋆, 208′, 210⋆], we have done so as well.

Also real-world implementation and testbeds exist and have been described in [216].

Location-Aided Routing While [67] describes a location-based routing extension to DSR using GPS, we want to devote a few words to *Location-Aided Routing* or LAR [179, 93] because it does *not* fit our definition of location-based/aided routing. LAR is basically source-based routing like DSR. However, since the flooding used for route requests has the nasty property of generating $O(N)$ messages and using resources all over the MANET, LAR improves this route request by geographically limiting it to a so-called *request zone*. The idea is that once the position of the destination node is known and the source route breaks, the destination can only have moved inside a certain area, assuming a maximum node speed. Thus, the flood is geographically contained.

2.5.5 Other Important Topology-Based Routing Methods

Facing the substantial number of routing proposals — most of them topology-based, we want to at least pay some respect to some others that dominantly came to our attention. The LUNAR [293, 294] or *Light-weight Underlay Network Network Ad-hoc Routing* protocol is a rather simple topology-based and proactive routing method that is deeply rooted in real-word ad-hoc research and shows remarkable real-world performance compared to AODV and DSR. ZRP [146] or the *Zone Routing Protocol* is a famous candidate for a hybrid routing protocol, combining proactive local and

reactive global routes. The two main IETF players in the proactive camp are called OLSR or *Optimized Link-State Routing* [95, 94] and TBRPF [235, 234], or *Topology Broadcast Based on Reverse-Path Forwarding*, both also on the RFC track.

For even more MANET routing methods, please check for the surveys, e.g., in [257].

2.5.6 Location Services

In Section 2.4.2, we have identified a *location service* as a building block in location-based routing. The task of the location service is to map an arbitrary node identifier to the node's (reasonably) current position by means of an ad-hoc protocol. [219*] classifies location services according to which node(s) hold location information about other nodes. E.g., a method by which every node would hold location information about every other node would be an *all-for-all* location service, while a location service where only the node maintains its own location for a *one-for-one* approach. These two points mark the extreme characteristics of a location server, the *all-for-all* approach constantly flooding the network with position information. Here, the consequences are similar to those in topology-based routing: Maintaining everything constantly creates a high overhead but provides a low startup delay for routing, while the more reactive a location service gets, the lower the constant overhead and the higher the startup delay. Methods intelligently distributing or even aggregating position information are a compromise, reaching higher scalability at the cost of some proactive overhead. In contrast to topology information, where a lost link can totally invalidate a route, position information remains valid for a certain time. Assuming a maximum node speed, the area where the node can be after a certain time period, can exactly be calculated. In addition, the exactness of position information becomes less important the further away from each other two nodes are. Or more colorful — the precise city block in downtown Manhattan is rather insignificant for getting a flight bearing from, say, Atlanta. This effect has been called DREAM or *Distance Routing Effect Algorithm for Mobility* [68, 93] and was exploited for a location-based directional flooding protocol.

Grid Location Service

The probably most prominent location service is the *Grid Location Service* or GLS [204], which exploits distance aggregation by structuring the network into a grid structure. To do this, the network is first covered by an appropriately large square. Then, this square is quartered (halved in every dimension) into smaller squares until the grid is so small that every node inside the smallest square is guaranteed to hear every other node sharing this smallest square. Thus, when position information is proactively and locally broadcast, every node inside this square is informed about every other node's position, or in other words every node serves as each others

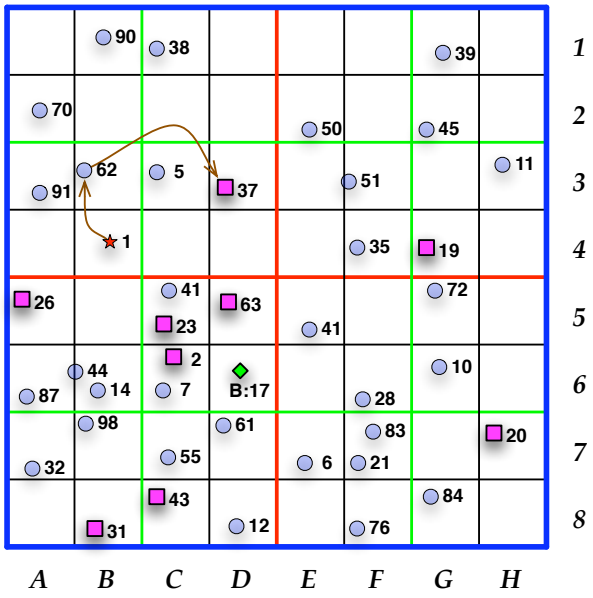


Figure 2.17: GLS example (from [204])

location server. Now for longer-distance position propagation, only some nodes in some other squares serve as a server. In the example from the original GLS paper depicted in Figure 2.17, the grid-like structure formed by the ongoing division of the blue top-level square is depicted. The 64 smallest squares of order $L = 0$, every four of them forming one of the red 16 $L = 1$ squares, again shaping the green 4 $L = 2$ squares, finally making the blue square of order $L = 3$, which covers the complete network area.

Now on every level, every node picks exactly one node for any of the three neighboring squares of the same next-order square, i.e., in the example, node *17* residing in square D6 is also a member of the higher-order (green) square C5-D6 and of A5-D8 (red). Thus for *17*, the three of the $L = 1$ location servers reside in C5, D5, and D6 etc. If there is more than one candidate for holding the location information, the one with the next higher node ID (wrapped-around at the maximum of the ID space) than the node concerned is selected. In the example, *23* is selected in square C5 because it is higher than *17* and closer than *41*. Routing to these squares works solely position-based on the basis of the squares' positions. In the picture, all location servers for *17* are marked with a square.

Now whenever a node, say *1* wants to know about *17*'s position, it climbs up inside the hierarchy until a location server for it is found. For this, it uses the knowledge that the location server has to hold the least higher node ID than *17*, which would be *62* in *1*s green $L = 1$ square. *62* is no location server for *17*.

However, it happens to be *37*'s location server, which is the ID-closest node in the red $L = 2$ square. Since *37* is indeed a location server for *17*, the resolution cycle is over. While the above describes only the basic functioning, the GLS authors' implementation is a very complicated one and covers a number of special cases like, e.g., nodes moving to other squares etc.

Analyzing GLS, one can say that it reaches scalability by thinning out the number of far-away location servers is both limiting the number of locations one node has to hold and the number of servers it has to provide its location for. However, a fundamental problem can be seen in the grid structure, which has been applied with complete unawareness of the real node distribution. E.g., if most of the communication is between nodes near in the network's center but only sharing its highest order, location queries can be quite inefficient.

We have extensively studied Grid as a candidate for our own routing protocols (see [195⋆, 194⋆]).

Geographical Region Summary Service

The second location service we describe is the *Geographical Region Summary Service* or GRSS [157], which also uses a virtual quad-tree grid structure, but using an all-for-all approach, i.e., every node knows about the location of every other node in the network. To make this method scalable, the creators of GRSS make use of so-called region summaries. A region summary is a bit vector with one bit for every possible node ID in the system. If it is set for a certain region, the node is located there. As in GLS, every node resides in exactly one square for each order or level. E.g., the red star node in C1 of Figure 2.18 is contained in four squares, a black, a green, a red and a blue one. For each of these squares, the region summaries are sent to the neighboring tree squares. After that, any node would at least know in which quadrant of the network every other node resides. If in the same quadrant, it would know the sub-quadrant and so on. Consequently, if the green diamond node wants to communicate with the red star node, it already knows it to be in the northwest quadrant and can immediately send packets there using position-based forwarding. When entering the destination quadrant, the granularity of information improves, and the destination region is adapted until the red star is reached.

While the advantage of this method is to be able to immediately send packets, the obvious drawback is its scalability with respect to the number of supported node IDs requiring one bit per ID. To alleviate this, the authors propose the usage of Bloom-Filters [21, 81, 224, 88], i.e., a data structure that has properties like a bit vector but needs less than one bit per possible value trading in some probability of error.

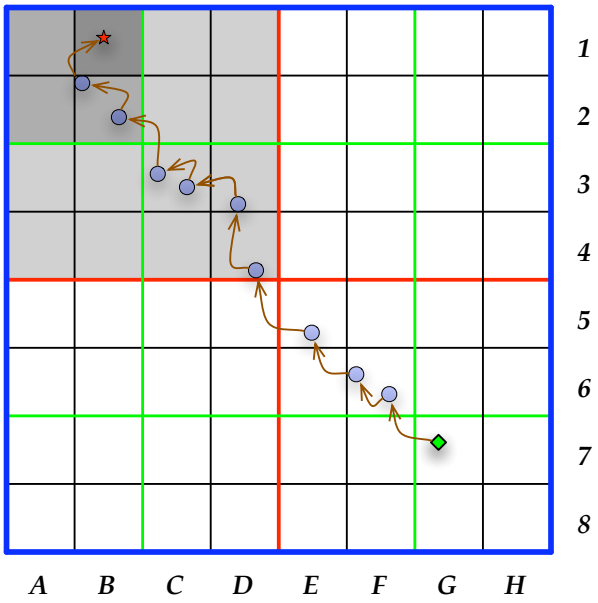


Figure 2.18: GRSS example (like in [157])

Other Location Services and New Directions

As we will see for many MANET-related problems in the course of this thesis, the playground for location services is rather crowded. [223] already lists some more relevant approaches, like the *Virtual Home Region* [138], where every node has one home region hash-calculated upon its globally unique ID. The node stores its location information in this region, and others retrieve it there. Another approach is the use of quorum systems [213, 147, 278], which is a well-known concept known from database and distributed systems research. In principal, nodes select a subset (quorum) of all nodes to be position servers, while for querying, another (intersecting) subset is chosen. The maybe most illustrative example is a north-south quorum, where every node propagates its position information to the north and to the south, and all queries are sent to the east and the west, intersecting inevitably.

DREAM [68] (see next Section) features a complete position-based routing system, also featuring an *all-for-all* location service part. Similar to GRSS, the scalability factor of DREAM's location service relies on the so-called distance effect, aggregating node positions to directions, stating that directions change the less, the further away from each other two nodes are.

A scheme more similar to GLS is DLM or *Distributed Location Management* [306], which also uses a grid structure laid out over the network.

A new direction of thought was introduced by *Geographic Hash Tables* or GHT [250], also allowing hash-based information retrieval that could be position information. Another example of current thoughts on location services is EASE [142], based on random walks and the location trace a moving node leaves in an *Ad-Hoc Network*. MLS [118], finally, is a MobiHoc '06 paper giving theoretical bounds for querying and node speed. Additionally, it has a very up-to-date related work section for the curious.

While much research has been and still is focused on location services, we will discuss two of our own proposals in Sections 3.1.1 and 3.2. For further studies on the performance of location services, please check [92].

2.5.7 Restricted Directional Flooding with DREAM

Restricted Directional Flooding [68] could also be called geographically optimized flooding and is thus to be seen in between flooding (Section 2.5.1) and position-based unicast routing. As stated in the last Section, the DREAM protocol keeps track of all nodes in the network by means of their relative direction. On the basis of this information and an assumed maximum speed, DREAM calculates a circle where the destination node can be located. Then, it connects this circle with the source node's current position. The forwarding algorithm inside this geometric form is now for every node to re-broadcast the packet as in flooding. However, there could be situations where re-broadcasting outside this flooding zone would be necessary to reach the destination (see recovery modes below). However, this case is not included in the DREAM specification.

2.5.8 GPSR / Face-2 and Descendants

Greedy Heuristics As stated earlier, the strength of using position information for MANET routing is in its strong correlation with the node position inside the graph, combined with a certain stability with regard to changes along the route. For many computationally expensive optimization problems like shortest paths, heuristics have been employed to find a good solution, trading in some optimality. In the same context, position information empowers us to apply greedy heuristics [40, 96] to find a path or route to the destination. Such a heuristic tries to approximate a globally optimal solution by making a locally optimal decision, or at least a local decision that achieves global goal improvement. In terms of position-based forwarding, the global goal would be to minimize the remaining distance to the destination to zero. Consequently, the corresponding local decision would be to select a next hop such that the remaining distance is minimized, or at least shorter. In this context, we generally define *greedy forwarding* as a strategy to achieve a local decrease in the remaining distance to the destination. Under the assumption that

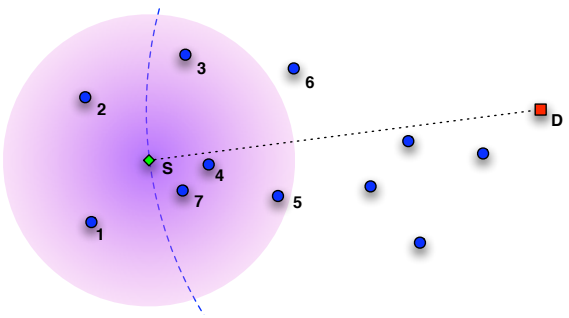


Figure 2.19: Greedy routing strategies (similar to [223])

the algorithm always finds a radio neighbor satisfying the constraint to be closer to the destination than the forwarder, this leads to a strictly monotonously decreasing target function. With the additional assumption of a minimal hop distance covered, the goal will ultimately be reached. Moreover, routes are inherently loop-free in a static setup.

While [117, 282] describe greedy forwarding, Figure 2.19 visualizes different well-known strategies for next hop selection. In this figure, node S is the source, or current forwarder, the shaded area its radio range. The blue dashed circle originates ad D and depicts the iso progress line to D, i.e., every position on this circle has the same distance to D, and since it goes through S, every node on D's side of the circle is closer to D than S itself. Thus, the nodes *3*, *4*, *5*, and *7* are possible candidates for forwarding. *2* and *1* do not provide distance progress, and all others, like *6*, are out of range. As in [282], the `MFR` or *Most Forward With Region* heuristic is the ultimate greedy method, selecting the maximum packet distance covered per hop. This strategy promises a small number of hops to the destination and is thus often preferred for greedy forwarding. In the example, `MFR` would pick node *5*. In spite of the desired short routes, [156] shows that the usage of this heuristic causes many packet collisions, whereas when nodes can adapt their signal strength, the heuristic *Nearest Within Forward Progress*, or `NFP`, is superior to `MFR`. This heuristic also selects nodes with greedy progress, but those closest to S itself (in the case of the example node *7*). Compass Routing is another heuristic discussed in [182]. While the above-mentioned methods calculate a target function based on node geometry, [230] proposes to randomly pick a positive-progress neighbor to minimize the operations needed for the routing process.

Void Situations When heuristics are simple rules to achieve good solutions for average cases in hard problems, they tend to fail to provide good solutions for some cases, or provide none at all. The same applies to greedy position-based forward-

ing for *Mobile Ad-Hoc Networks*, a typical example of which is being depicted in Figure 2.20. Again, we assume that S wants to send packets to D and already has

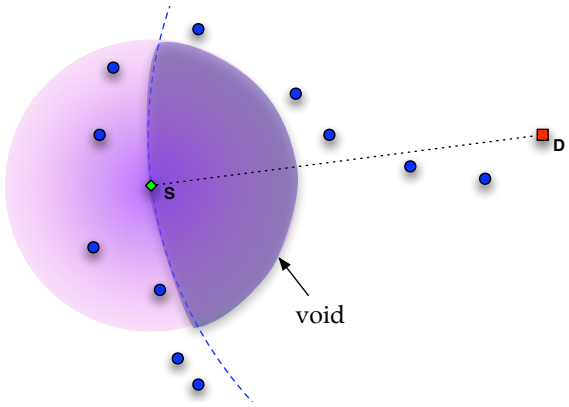


Figure 2.20: "Greedy" void

acquired position information about D and all its radio neighbors, i.e., the nodes inside the shaded radio range area. Calculating the distance progress these neighbors would provide towards the destination, S finds that all reachable neighbors are farther away from D than S itself. Consequently, no greedy heuristic can be applied. In the language of position-based routing, this is called a void situation, since the greedy part of the current hops radio range is empty of neighbors.

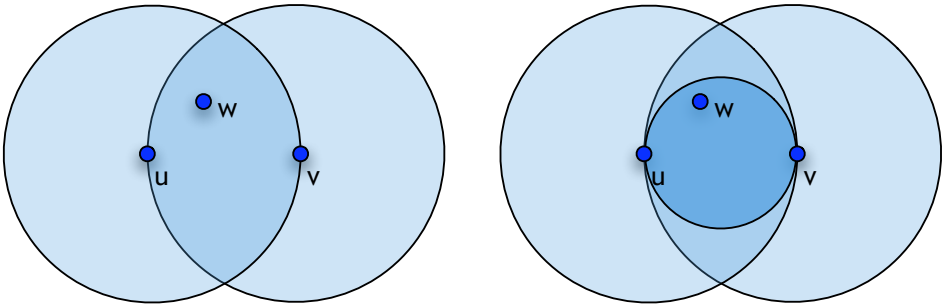
Graph-Based Recovery Strategies Greedy forwarding has been shown to outperform topology-based methods in a couple of general scenarios [171, 169]. However, simply not being able to find a route to a destination is not acceptable. Here, one can always step back to a topology-based method in a non-greedy situation. However, what helped position-based routing break through were methods that allowed the circumvention of voids without the acquisition of multi-hop topological knowledge.

Two similar and parallel proposals were published in [171, 85], the former proposing a complete protocol called `GPSR` or *Greedy Perimeter Stateless Routing*, with greedy forwarding being the standard forwarding mode and the so-called *perimeter mode* for recovery. The latter proposes an algorithm called face-2 with the same algorithm, but without greedy as the default.

The recovery strategies of both work very similarly. First, they distributedly calculate a planar but still connected subset of the network graph, i.e., a graph still containing all vertexes but only a subset of edges that do not cross each other.

To construct these planar graphs, `GPSR` is able to use two different methods known from graph theory, the *Gabriel Graph* [136] and the *Relative Neighborhood*

Graph [286]. Construction works as follows. Take the graph constructed by the local radio neighborhood and remove edges violating the constraints depicted in Figure 2.21. On this planar graph, the so-called *Right Hand Rule* [50] is applied,



(a) RNG Graph: edge (u, v) is only included when there is no visitor w in the intersection.

(b) GG Graph: edge (u, v) is only included when there is no visitor w in the small circle.

Figure 2.21: Distributed graph planarization methods (as in [171])

i.e., a graph traversal strategy that allows to stay in touch with the void.[15] Methods based on this graph traversal have been shown to be complete for the static case, i.e. they find all existing routes.

`GPSR` combines the graph-traversal strategy plus greedy forwarding whenever possible to a solid `MANET` protocol. To generate information about the radio neighborhood, position-beacon packets are used.

Known Problems The first obvious problem with these graph-based strategies is that the particular efficiency depends on whether the packet turns right or left when confronted with a void. Here, tne route could be very short while the other could lead along the edges of a potentially huge network. Secondly, [170] describes the fact that loop freedom is not guaranteed in case of mobility. Moreover, inconsistencies in recorded neighbor positions can also spoil routing success.

With face-2 and `GPSR` coming up front, there is a whole class of derived protocols [188, 189, 186, 187], the most modern being `GOAFR+` which limits recovery graph traversal to ellipsoid areas, changing direction when the routing success is endangered. This protocol has been shown to perform worst-case optimally and still is average-case efficient.

An imminent class of problems arises when the underlying assumptions are significantly violated, such as the unit-size circular shape radio propagation model.

[15] The graph strategies are not crucial to this theses. Thus, we only list and explain how they work in principle.

Especially using the MFR heuristic for greedy selects — in reality — poor-link neighbors for forwarding.

2.5.9 Other Approaches

While the graph-based approaches outlined above are still state-of-the-art for theoretical MANET routing, researchers have tried to find heuristics dealing with the problem, attempting to find good solutions for practical applications. [205, 206, 277] discuss alternative routing methods of this kind.

For an overview of position-based routing methods, please refer to the surveys [223, 139].

2.5.10 Classification with Respect to Fundamental Strategies

This section's selection of algorithms make use of the strategies outlined in Section 2.4. Figure 2.22 shows for some of the categories, how much the different algorithms use them.[16]

In detail, all algorithms except DREAM and GPSR work solely on the concept of topology. While flooding does so implicitly by forwarding the packet repeatedly to every node's topological neighbor, DSDV, AODV, and DSR do so explicitly by gathering topological knowledge about unicast paths. DREAM is very similar to flooding, except it geographically contains the flooded area.

The second column characterizes the proactiveness of the algorithms: Flooding, AODV, and DSR have no proactive elements, while DSDV prefers to compute all routes regardless of whether or not they are needed. GPSR also routes proactively. However, it has the proactive beaconing component to gather neighborhood information. DREAM has a small proactive component that keeps position information up to date.

Concerning the usage of hard-state protocol elements, we mark Flooding with a white arrow because it is utterly stateless. Since DREAM is similar to flooding, it has only information about the destination's position to complement the flooding algorithm. This information, however, is acquired with a soft-state protocol. DSDV, AODV, and DSR on the other hand, make use of some hard-state signaling when it comes to route solicitation messages. However, in every case this is backed up by a soft-state expiry process. GPSR, in its pure form, does not use any kind of hard-state signaling. All protocol state is set by packet content and reset after an expiry interval. Ordering the amount of state that the protocols use, it is Flooding that uses no state, with DREAM's additional position-based information, followed by AODV. DSR is slightly worse because they use more state in the packet header (the source routes) and they store whole paths instead of AODV's lean distance vectors.

[16] We are well aware that this is a simplification of facts and represents our view on things. Also, the figure should be treated as having an ordinal rather than a rational number scale.

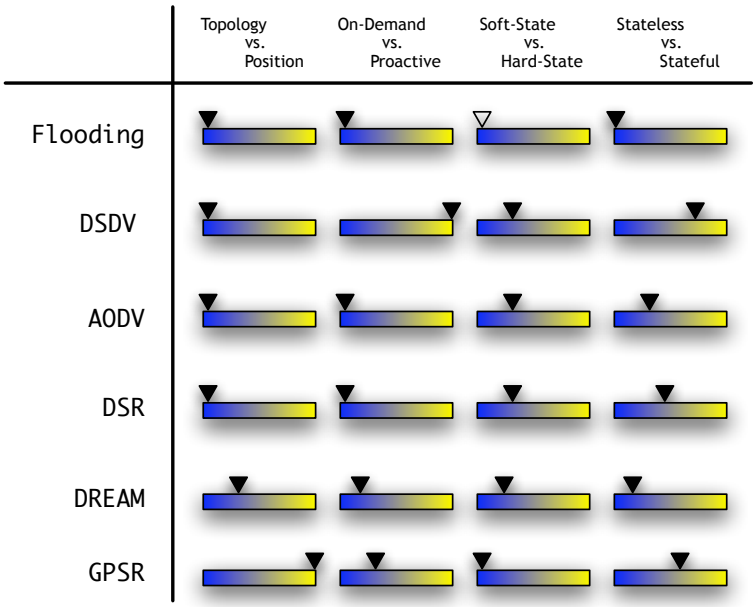


Figure 2.22: Level of routing strategies within algorithms

In our ranking, GPSR is next because of they neighborhood state proactively kept current. The 'stateless' in GPSR's name refers to the perimeter mode, which is able to calculate a next hop without storing other information than the neighborhood position information. The most stateful protocol is DSDV because it keeps routing information to every node in every node. While the information itself is a lean distance vector table, it still requires a single entry for every node in the same partition.

2.5.11 Protocol Strengths and Weaknesses

Quantitative Comparison of MANET protocols is the holy grail of MANET research, however there is probably no existing protocol where a motivated researcher would not be able to find a scenario in which his protagonist would not outperform its opponents. To help the reader to get a feeling for the strengths and weaknesses of the protocols, we list some general remarks on protocol performance with regard to four parameters "resilience to mobility", "resilience to number of nodes in the network", "resilience to number of communicating node pairs", and "resilience to offered load". The grades we give are backed up by basic algorithmic properties and by results published in [100, 171, 68, 240]. In the rest of this subsection, we will argue for the grades we have given.

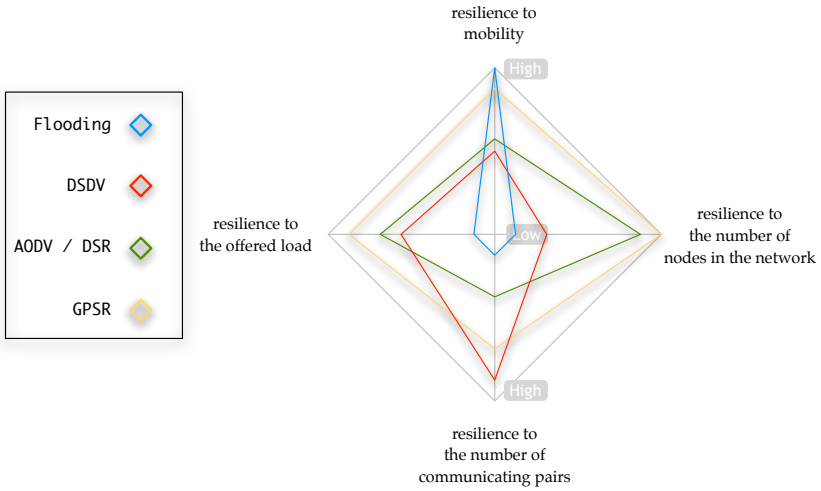


Figure 2.23: Routing protocol performance

Figure 2.23 shows our performance grades as a cobweb diagram. Every corner of the square represents the respective category and every quadrangle the performance profile of one of the algorithms (`AODV` and `DSR` share the same line because we gave them the same grades.) The higher the grade, the further away from the center the algorithm's corner is located. In contrast to the last section, `DREAM` is not graded here because it is not able to find a route for every packet.

resilience to mobility The winner in this category is clearly Flooding, because it does not any state that is outdated in the presence of higher mobility. Out of the real unicast protocols, `GPSR` is the next best, because it does not keep routing state along the route. `AODV/DSR`, however, do so and this state is frequently invalidated when nodes move. Consequently, these routes break reducing protocol performance. An even lower grade has `DSDV` because mobility alone, without even tasked to transport a packet, can break the network.

resilience to number of nodes in the network Here, `GPSR` scores best, `AODV/DSR` being worse because of their flooding component in the route request. To understand this, `GPSR` assumes the knowledge of the destination node's position, giving it an unfair advantage. However, for this comparison, we have studied pure `GPSR` without a location service. Here, `GPSR` would loose performance accordingly when complemented by a location service. `DSDV`, however, is behaving badly when the number of node grows since the number of proactive update messages grows quadratically. With Flooding this is even worse, because, packet transmissions grow linear in the number of nodes and in the number of packets to be sent.

resilience to number of communicating node pairs This section is the strength of DSDV, because the number of node pairs that actually communicate is irrelevant. For GPSR, it is not, but due to the relative statelessness and the re-usability of position-information, we have ranked it better. While communicating node pairs increase the number of floods with Flooding, AODV/DSR also suffer from their route request floods that are necessary per node pair.

resilience to offered load While clearly Flooding is the bad boy of this category with his $O(N)$ packets per end-to-end payload, the other algorithms behave similar with their O(path-length) packets. However, due to the different protocol overhead, they reach the network's limits at different times. Thus, the different grades.

2.6 The Transport Layer in Mobile Ad-Hoc Networks

Coming back to the network layer model at the beginning of this chapter, we have now discussed layers 2 and 3, the latter being the natural focus of this thesis. However, since the so-called upper layers supply the network layer with problems to solve, we will have a quick look at the transport layer and above.

The network layers we face in *Ad-Hoc Networks* are usually unreliable and independent datagram packet layers, i.e., every packet is completely addressed and handled separately by the forwarding hosts. Following the transport layer's definition (as in [190]), the transport layer has to provide logical communication between end systems, e.g., by offering a transparent data stream like TCP [55, 245, 275]. Then, the protocol transparently hides the network's properties and notifies its users if it cannot fulfill its task.

Another important task usually handled by the transport layer is congestion control. From a user's perspective, the transport protocol chooses the rate at which the packets are sent and slows down the application if necessary. The ultimate goal of this is to get the best of a given network, fairly sharing bandwidth between different streams. With TCP, this is done by increasing the rate until a packet remains unacknowledged; then, the rate is drastically reduced, regardless whether the packet was lost due to a transmission problem such as a bit error or due to real congestion. Even in standard IP networks that include a wireless link, TCP fails to work efficiently because the high link failure rate lets TCP's sending rate drop far too often. [63] evaluates different TCP modifications w.r.t. to their capability to improve performance in the presence of wireless links. As expected, MANETs with their high variances in almost all network layer performance indicators additionally create problems for transport protocols. E.g., on-demand behavior in combination with link breaks leads to TCP stalling [215]. Similar problems exist in the presence of routes with asymmetric performance [62]. [108, 155, 121] all analyze TCP variants

over multi-hop networks. [145] models wireless links to capture transport protocol relevant properties. [253] discusses general interaction between *Ad-Hoc Networks* and TCP/IP networking.

As were almost all aspects of wireless networking, transport protocols have been subject to intensive research. Keeping the scope on TCP-like protocols, the following protocols give examples for typical methods:

Wireless TCP Proposed in [269, 270], Wireless TCP is meant to cope with the special problems of wireless links. As opposed to TCP's window-based transmission control, it rather calculates a desirably smooth sending rate based on the ration of transmitted vs. received packets. However different, WTCP is still a purely end-to-end protocol that does not require any changes in routers.

ATP In contrast, ATP [281] only shares TCP's upper service interface, while its functioning heavily relies on protocol changes inside the network. Being rate-based like TCP, ATP calculates its rate using information gathered while en-route. Additionally, congestion and error control are decoupled and are both treated with regard to the actual reason for their occurrence. To be able to do this, the sender is explicitly notified of congestion-related losses.

CXCC *Cooperative Cross-Layer Congestion Control* [7] is not really a transport protocol but an emerging project dealing with the question of how to dynamically maximize the utility of a MANET's available bandwidth. In principal, it uses queues of length 1 per transport stream inside the network, and utilizes cross-layer functionality to detect whether or not a packet was correctly forwarded. Only then is the next packet accepted.

PDP Path Density Protocol [236] is a protocol calculating the density of streams over routing nodes and throttling the sending rate at ingress nodes.

A good overview of many more methods can be found in [110]. Also, [264] has a good overview of MANET transport protocols, especially with regard to *Vehicular Ad-Hoc Networks*.

The Capacity of Wireless Networks

Fundamentally, a transport protocol's congestion control algorithm tries to get the most of a network's capacity while being fair to different streams. Ultimately, this has led to the question about the nature of a MANET's capacity, which was partly answered in [144, 143], the first of the two being one of the most-referenced papers in MANET research.

The main analytical result is, that even under theoretically optimal circumstances, the achievable bandwidth in an *Ad-Hoc Network* is of order $\Theta\left(\frac{W}{\sqrt{n}}\right)$, given

the wireless link bandwidth W and n nodes. These devastating results attenuated euphoria about global *Ad-Hoc Networks*. Moreover, the common opinion in the MANET community is that communication always has to be "as local as possible" to be feasible. This is backed up by [203], a simulation-based study that more deeply explores the practical issues of the capacity constraints of and *Ad-Hoc Network*.

2.7 Research Methodology

Having introduced the state-of-the-art research relevant to this thesis, we will now spend some time on our research methodology. As you will see at some parts during the remainder of this work, research in *Mobile Ad-Hoc Networks* is far from perfect in a mathematical sense. Moreover, in most of the cases we cannot ultimately decide which algorithms are better than others by means of decision theory nor mathematical analysis. We use the research methods described in this section to help understanding and arguing.

2.7.1 Mathematical Analysis

Being aware that logical or mathematical analysis is the silver bullet of research, we will use it wherever the situation permits. However, network algorithms are almost always very unhandy in analysis due to their discrete nature. If mathematical analysis is at all possible, assumptions have to be very harsh making results to be hard to compare to actual performance. Thus, most of MANET research is based on discrete event simulation.

2.7.2 Discrete Event Simulation

Discrete Event Simulation [254, 199] or DES is a technique to simulate complex systems by decomposing their state development over time into discrete steps and fast-forwarding the time between the steps. Since the logic of network algorithms behaves just in this manner, DES is most frequently used for the development of network algorithms, especially due to the benefits in the following categories:

Development DES helps in the development by visualizing the information requirements in a distributed setting, i.e., one cannot use data one did not acquire distributedly. Also, the sequence of events in more complex settings is something not easily predictable ex ante.

Teaching and Understanding As opposed to formal specifications, DES allows for a hands-on experience of network algorithms, better helping to understand very complex workflows.[17]

Quantitative Comparison Carefully evaluated statistics generated with DES can be helpful in comparing the performance of algorithms.

The ns-2 Network Simulator

The most popular network simulator for MANET research is the network simulator ns-2 [15, 114], which was originally developed for the simulation of Internet-style wired networks. In 1998, [6, 87] introduced Wireless Extensions to the originally wire-only simulation framework.

Ns-2 is a powerful tool based on C++ [27, 280] and OTCL [54]. Its features can be briefly outlined as

- DES framework with a fast C++ engine and comfortable scenario creation in C++.
- Rich set of available protocols, especially almost all RFC'd Internet Protocols.
- Basis for the vast majority of networking papers, especially MANET papers.
- For MANET simulations, ns-2 supports the following features:
  - Mobile nodes having linear movement segments.
  - Different Radio Propagation Models, both randomized and deterministic.
  - A built-in 802.11b PHY/MAC model.
  - Many MANET routing methods, such as DSDV, AODV, and DSR.

Mobility Modeling

For mobility modeling, we have relied on Random Waypoint scenarios with varying parameters (see Section 2.2.3), since it is used in all major work in the MANET sector. [72, 307] lists potential problems, especially in low-mobility settings. We have integrated them into our scenario planning. E.g., we have chosen short simulation run-times to avoid the concentration of nodes in the center of the simulated region. Also, we have avoided to choose the movement speed from intervals containing zero to avoid node stalling.

[17] In parallel work that is not part of this thesis, we have also developed tools to allow the visualization of network simulation [10, 259⋆, 261⋆, 260⋆, 258′].

A major part of this thesis (Chapter 4) deals with *Vehicular Ad-Hoc Networks*. In these scenarios, our network nodes followed realistic movement patterns which are described in the the corresponding sections.

Radio Modeling

While deterministic radio modeling is still used for most current simulation studies, the ratio of studies using probabilistic models is growing significantly. Deterministic Radio Modeling in this context means the application of a Unit Disk Graph Model [115, 188] or `UDG`, i.e., given a fixed parameter r denoting the nominal radio range, two nodes are considered to be able to receive each other's packets whenever their Euclidean Distance is shorter than r. On the other hand, probabilistic radio modeling introduces distance-dependent probabilities for packet reception, usually on the packet level. For a deeper view of radio modeling, please check [249, 264].

Simulation Statistics and Performance

Simulation studies have been frequently criticized [238, 151, 150, 283], mostly for modeling errors, unconsidered effects, and the lack of statistical validity. E.g., accounting for the theory of statistics-based decision making, almost all statistical decision tests are based on the law of great numbers, which needs a fairly high number of statistical repetitions even to start with. However, simulating *Ad-Hoc Networks* is computationally expensive. For example, the sending of a single radio packet usually scales with the number of nodes, since every node has to check whether or not it is in decoding distance. Combined with other effects, this means that many problems we have considered take hours to days per run, i.e., per statistical experiment. Since the estimators for mean and variance only converge to the true values, they are (almost) meaningless for so few experiments. However, we will use at least the mean extensively and will verbosely classify the stability of the values. But in terms of statistical significance, the sheer numbers are of little value.

2.7.3 Real World Evaluation

Real-world evaluation is expensive and sometimes not even feasible, especially when it comes to studying scalability, or when non-existent radio or `MAC` hardware is simulated. Consequently, there are only a few real-world projects, and the acquired results are often from very small networks and then mostly about routing protocols using standard wireless hardware/driver combinations. However, recent work [295, 212] lists some critical modeling errors that resulted in drastically reduced performance in reality. While many projects do not have the resources to study their protocols in real life, we have implemented and tested a significant

number of our protocols in real systems. However, one has to keep in mind that even with a high number of evaluation runs, the results can only tell about the special scenario setting chosen and can only hint at general protocol performance. E.g., there could be a significant environmental factor like the weather that, when changed, could completely change the evaluation results. Nevertheless, real-world testing helps to identify (a) crucial simulation modeling mistakes and (b) critical system dependencies like the necessity for positioning hardware etc. At the very least, a real-world study proves the feasibility of a concept while simultaneously showing its limits.

Chapter 3

Position-Based Packet Forwarding Algorithms for Mobile Ad-Hoc Networks

> Abstraction is selective ignorance.
>
> *(Andrew Koenig)*

Chapter Outline

In Chapter 2, we have set the back- and playground for our own research. In this chapter, we will start with network layer algorithms for "general" *Mobile Ad-Hoc Networks*, i.e., for `MANET`s with no special assumptions about node mobility. In simulation, we model these networks with random node placement and random node movement. Most of the time, the issue of energy consumption is ignored, and all nodes are believed to be equally equipped with sufficient `CPU` power.

Using the last chapter as a foundation, we will first spend some time proposing a simple, yet efficient location service called *RLS* (Reactive Location Service). While this work started to fill the gap of previous position-based forwarding schemes using simulator knowledge to acquire destination positions, Section 3.1 will introduce `RLS` and also show the advantage of this simple scheme, especially in the case of high node mobility. The subsequent Section 3.2 will briefly outline the proactive location service called *HLS* we have developed for scalability (rather than coping with high node speeds).

Having established the destination's location, we come to the heart and backbone of this thesis: *Contention-Based Forwarding*, or `CBF` (Section 3.3), a novel approach to greedy packet forwarding on the basis of positions with ground-breaking efficiency in highly mobile networks.

Working only in greedy mode, `CBF`'s non-greedy companion `CBDV`, or *Contention-Based Distance-Vector Routing*, is introduced in Section 3.4. This method combines

topology-based routing with a contention-based, i.e., opportunistic, way to chose a forwarder.

The foundation of this thesis is already listed in the previous chapter. However, there are proposals published later than our original work which we believe to be worth mentioning. This, we will do in Section 3.5. Concluding this chapter, Section 4.

Most of the work presented in this chapter has already been published. Readers interested in single concepts, are referred here to read the papers, namely Section 3.1/RLS [194⋆, 195⋆, 193⋆], Section 3.2/HLS [173⋆, 172′], Section 3.3/CBF [135⋆, 134⋆, 133⋆, 192′], and Section 3.4/CBDV [202, 200′, 201⋆].

3.1 A Reactive Location Service for Mobile Ad-Hoc Networks

In this section we present a *reactive location service* (RLS) that allows a mobile node in an ad-hoc network to inquire about the geographic position of another node in an on-demand fashion. We present ns-2 simulation results that show that RLS is a simple, yet efficient and effective location service for *Mobile Ad-Hoc Networks*.

Prior to [192′, 194⋆, 195⋆, 193⋆], work on position-based MANET routing separated the problem into (a) issues dealing with location services (see Section 2.5.6) and (b) those concerned with the actual packet forwarding. Mostly, the location service community presented systems that could also do the forwarding [68, 204]. The forwarding community, however [171], assumed knowledge of the position and thus used simulator knowledge to gather the destination's position. Since this is independent of location service selection, it creates unfairness in comparison to topology-based routing protocols.

The objective of the work described in this section was to devise and study a purely reactive approach that does not do any routing itself. The reactive location service we propose and analyze represents an adaptation of the route recovery mechanism of Dynamic Source Routing (DSR) [165] to the domain of location-based routing approaches. Since DSR's route discovery and, therefore, RLS' location request are based on flooding principles, the well-known 'broadcast-storm' problem [231] has to be addressed. We make use of strategies outlined in [302] to deal with this problem. In order to show RLS efficiency and effectiveness, we compare it with an omniscient, i.e. all-knowing, location service (OLS) as well as the Grid Location Service (GLS) [204, 195⋆, 194⋆] that represents an approach with a strong proactive component. In addition, we compare the combination of greedy location-based for-

warding and RLS with DSR. An in-depth analysis based on ns-2 simulations that evaluate the impact of node mobility and density is presented.

3.1.1 RLS — Algorithm Design

Baseline Algorithm

When a node wants to communicate with a another node by means of position-based routing, it needs to add the geographical position of the target node to the headers of all packets it intends to send to that node. The routing scheme then forwards these packets to the given position, which — in an ideal case — would be the current location of the destination node at the time of the packet's arrival. In a real-world situation the geographical position can never be 100% accurate, and it is the task of a location service to provide as precise a position as possible. To do this the location service may use any algorithm that is able to service an (id,location)-pair to an inquiring node.

In RLS, the algorithm works as follows: Any query for the geographical position of a certain node issues a *location query* packet. The query packet contains the source node's location and id as well as the id of the destination node. It is flooded throughout the network until it reaches the destination node or its time-to-live (TTL) expires. If the destination is not reached, RLS assumes unreachability, which represents one of the following cases:

Network Partitioning If the network is partitioned, with source and destination in different partitions, the destination can never be reached by the query and it will eventually expire.

Inactive Node The destination node does not exist or has (temporarily) been deactivated. Thus it can never be reached by the query.

Great Distance Source and destination may be farther apart (in hops) than the maximum time-to-live allows for.

The above cases are indistinguishable to RLS since error detection is based on a timeout mechanism at the sending node. Note that network congestion can have the same impact on RLS as network partitioning.

To avoid infinite packet looping and duplication during flooding, nodes must be kept from forwarding queries they have already processed. Therefore, the source node marks all location query packets it sends with a sequence number that increases with each attempt by the source node to acquire the destination node's location[1]. Each forwarding node then uses a sequence number cache to check whether

[1] Therefore, a sequence number is associated with a (source,destination)-pair.

or not it is permitted to forward this packet, by comparing the sequence number stored in the query packet to the one in its cache. If no cache entry exists or the stored sequence number is smaller, the node never had this query and updates its cache before rebroadcasting it. A cached sequence number larger than the one contained in the packet indicates a duplicate or looped packet and the query is discarded.

When a destination node receives a query packet that carries its id, it creates a *location reply* packet that is marked with the query's source id and location as destination information and carries the query's destination id and location as payload. This reply is sent back to the source by means of the underlying routing protocol (e.g., greedy unicast routing, flooding, etc.). Receipt of a reply packet at the querying source completes the location discovery cycle. The destination's location is inserted into all packets that are buffered for this destination, and the packets are sent out.[2]

If unreachability occurs or a reply is lost on its way back to the source node, a timeout for the data packet will occur at the source, and another location request cycle will be initiated: The sequence number is increased by one and a new location query packet is sent. The justification for this automated retry scheme is the the fact that all unreachability criteria mentioned above are subject to change in mobile scenarios. Furthermore, the loss of a reply packet is also indistinguishable from the location service's unreachability criteria. Thus, repeating a location query increases the chances of a successful location discovery. On the other hand, there are upper limits to how long a packet may be delayed before it becomes obsolete, and to how much additional load the location service should put on the network by itself. Therefore, the retry mechanism has an upper bound at which retries cease and the data packet is discarded.

RLS Extensions and Improvements

Flooding Schemes Flooding defines a simple packet distribution method: Each node re-broadcasts all the packets it receives. If we assume that the network is not partitioned, we can state that if the `TTL` of the packet is at least as high as the diameter of the network and no link layer failures occur, all packets will reach every node in the network.

This basic flooding approach suffers from a number of problems, e.g., non-reliability of link layer broadcasts or the overload of the link layer by the 'broadcast storm problem' [231]. In order to help with the latter, a number of proposals have

[2] The time of validity for a certain destination node's position is dependent on the scenario's mobility. A safe assumption is to believe it valid for $\frac{radiorange}{maximumspeed}$ s. However, if communication is bidirectional, the exchanged packets take care that both nodes have accurate information about each other's position. (A paper discussing this topic can be found in [174].)

been presented in the literature (see [302]). Other problems can be addressed by different modifications to the basic flooding algorithm.

For RLS we considered the following three options:

Linear Flooding This derivative floods a small neighborhood region first by limiting the packet TTL value, d_{max}, to a small number of hops. If a timeout occurs, d_{max} is increased by a constant, d_{step}, and the query is restarted. If the destination is not found before d_{max} reaches the allowed limit, the destination is labeled unreachable.

The biggest drawback that we encountered with linear flooding was that nodes were easily able to remain "hidden"[3] to a source node. The reason for this is that a node that moves away from the source can remain in front of the expanding query wave by moving fast enough to pass at least d_{step} hops in the timeout period needed by the source before it restarts the query with an increased hop limit.

Exponential Flooding To attenuate the phenomenon of "hidden" nodes, we tested exponential flooding. This method works like linear flooding, but instead of increasing d_{max} by an additive constant, it is multiplied by a factor. This limits the chance of nodes staying "hidden" because nodes would have to move fast to outrun the "query wave" after the first few retries.

Binary Flooding In many real-world scenarios, communication is often local, for example, at conferences or during courses on campus. It therefore makes sense to use a flooding scheme that discerns only two types of communication: *near* and *far*. We called this approach, which was inspired by the route requests of DSR, binary flooding. The source node first floods a close-range neighborhood (e.g., one or two hops) to see if near traffic is intended. If no reply is received, the traffic is classified as far, and d_{max} is set to the allowed limit rather than increasing it gradually.

For the evaluation of RLS, we chose binary flooding as the main flooding scheme, and only performed measurements with exponential flooding for comparative purposes. An in-detail study of exponential flooding was left for future work.

Caching Every node forwards queries, replies, and data packets each of which carries location information on their respective source nodes and maybe even on a destination node. By evaluating these passing-by packets and storing this information in a location cache, a node can acquire valuable location information on other

[3] A hidden node is a node that is part of the network, but whose position cannot be acquired by a node that queries for it.

nodes for free (i.e., without any extra costs in the form of packet overhead). If a node wants to initiate a communication, `RLS` then checks its location cache first and may be able to answer the location query of the data packet right away. This saves a location discovery cycle and reduces packet delay as well as network traffic.

In addition, it is also possible to evaluate any other type of passing packets for information that may be used to fill the local cache.

Cached Replies A query does not necessarily have to be answered by the target node itself. A cached-reply strategy allows nodes to answer queries not destined for them if they have the required location information available in their respective location caches. Since every node gathers location information for its own communications, or even from packets that are passing by (as described in Section 3.1.1), replies might be generated by nodes much closer to the source than to the intended destination. This reduces latency, but might provide the source with less accurate positions. Another problem of this approach is that it is likely to produce more than one reply, and the source node has to be able to deal with this fact.

Cached replies may also be used to reduce network load, but would have to be used in conjunction with a flooding scheme like exponential flooding, as described in Section 3.1.1, to do so. If the whole network is flooded, no relief in network load can be achieved due to the fact that the destination node is likely to be reached anyway, and a cached reply does not prevent the location query from being forwarded by other nodes.

Radial Flooding In all standard flooding schemes, each node within radio range that receives a location query packet rebroadcasts it as soon as it can acquire access to the wireless medium. This may lead to collisions that can be reduced by introducing a random backoff at each node. We extended this approach by adding a "radial" component whose purpose is to increase the expansion speed of the query flooding, while still providing the congestion alleviation of a random backoff. We achieve this by having each node that receives a query compute its distance to the last hop node and calculate a backoff time with respect to this distance by

$$t_{backoff} = t_{max} \cdot \left(1 - \left(\frac{d_{last}}{d_{rrange}}\right)^2\right) \tag{3.1}$$

, where t_{max} is the maximum delay a packet may be backed off for, d_{last} is the distance to the last hop node, and d_{rrange} equals the nominal radio range. Note that $d_{last} \leq d_{rrange}$ and that d_{last} is calculated as the distance between the own position and the position of the last node, which is contained in the packet header.

This ensures that the farther away a node is, the sooner it will rebroadcast the query. Assuming a two-dimensional uniform distribution of nodes in a circle with a radius equal to the radio range, the distribution function of distances to the center of this circle is quadratic. This means that there are potentially more nodes with great distances than nodes with short distances.

To prevent these nodes from trying to transmit at the same time, the resulting timer distribution should distribute uniformly over a certain time interval. The timer function shown in equation (3.1) has this property (as shown in Appendix A). This means that the intervals from which the timer values are taken increase with the distance from the last hop node since the number of nodes in a uniformly distributed scenario also increases with the distance from the source node.

Rebroadcast Suppression Having each node rebroadcast queries may lead to redundant packets, i.e., packets that reach few additional nodes[4] or none at all but congest the network. This phenomenon is called the *broadcast storm problem* and was discussed in [231], where a way to alleviate this problem by means of packet suppression has also been presented. Different suppression mechanisms were shown that keep nodes, which are unlikely to reach additional nodes, from rebroadcasting packets.

Based on the evaluations of these suppression mechanisms done in [302], we chose to implement a combined distance-/counter-based scheme for `RLS` to study the effect of suppression on the performance of a flooding-based location service. The results are described in Section 3.1.2.

The rebroadcast suppression itself is implemented as follows: The first time a node receives a location query packet, it checks its distance to the packet's last hop node. This distance is then evaluated against a threshold value, d_{thres}, to decide whether enough additional area coverage is expected to be reached so as to justify a rebroadcast. If that is the case the packet is scheduled for retransmission after a so-called random assessment delay (`RAD`), which needs to be long enough to receive packets from all nodes within the radio range. If the expected additional area coverage is low, so is the chance of reaching additional nodes, and the packet is discarded. During the `RAD` the node may hear the packet again from neighbors that have rebroadcast it. Every time this happens, the node calculates its distance to the source of the rebroadcast, and compares this distance to the one calculated the last time. The smaller distance of the two is then checked against the threshold, and again, a decision is made whether or not to discard the packet. However, this time the decision to rebroadcast only resumes the wait for the end of the `RAD`. This method ensures that a node will always associate itself with the closest (re-)broadcasting neighbor and calculate the least expected additional coverage (`EAC`)

[4] An additional node is a node that has never had the packet in question before.

it can achieve. Should the least distance ever fall below the threshold, the packet will be discarded.

If the node hears the packet more than c_{max} times, it discards it, regardless of any distances, because it is unlikely that any additional nodes will be reached by its rebroadcast that were not yet reached by one of the previous rebroadcasts or that they will be reached by one of the many neighbors. c_{max} is chosen according to the results in [231].

Passing-Packet Updates Another optimization that may be implemented in RLS is the passing-packet update service, which checks every packet received by a mobile node for positional information and updates it, should a query of the location cache produce more accurate (i.e., newer) information. The justification for this is that the closer a packet gets to the destination node, the more likely it is that forwarding nodes have more precise location information in their caches due to beacon and other localized traffic, and the time that has already passed since the packet was marked and sent.

3.1.2 Evaluation

Simulation Setup

To evaluate RLS, we compared it to two other location services, called GLS and OLS. All location services used GPSR [171] as the underlying greedy position-based routing strategy. We also compared the GPSR/RLS combination to DSR [165], which we chose because it uses a very similar flooding technique for the route discovery and thus highlights the differences between position-based and topological routing in *Ad-Hoc Networks*. DSR also has the advantage that it is often mentioned in other papers dealing with *Mobile Ad-Hoc Networks*, and thus our results become comparable.

GLS is our own implementation of the Grid Location Service, as it was introduced in [204] and briefly described in Section 2.5.6. Since the original authors used a greedy routing scheme based on a two-hop neighborhood and grid-based forwarding, specific to this location service, while we pair it with the standard GPSR scheme, it is very likely that our scheme does not represent the maximum performance of GLS. However, if we had implemented all optimizations, we would not have been able to extract as much information from the comparison of the one-for-one[5] scheme used in RLS with the some-for-some[6] approach of GLS as we could by keeping the differences between the two to a minimum .

[5] Each node is the only participant in the network that may answer queries on his location.

[6] Some nodes are selected as location servers for some other nodes and a location query may be answered by any such location server that holds the required information.

OLS stands for Omniscient Location Service, which is based on the assumption that each node can acquire positional information on any other node in the network without delay or inaccuracy. Although this can never be achieved in a real-world implementation, it is a legitimate assumption in a simulation environment and provides an impression of GPSR behavior and greedy connectivity. OLS behaves like the location database used in [171].

Simulator	ns-2.1b8a
Area size	2000 $m \times 2000$ m
Number of nodes	100 – 400
Mobility model	Random Waypoint
Node speed	10, 30, and 50 $\frac{m}{s}$

Table 3.1: RLS: Simulation setup

The basic simulation setup can be found in Table 3.1[7].

The communication patterns contained 20 node pairs during a time window of 25 seconds (starting at 15 seconds into the simulation and ending 40 seconds into the simulation) and sending 200 packets per connection from the source node to the destination node at a rate of four packets per second. A complete simulation run was 120 seconds in length. Consequently, all nodes had at least 50 seconds in which to send all of their packets, and 30 seconds in which to finish routing any packets that had not yet been delivered or dropped. For traffic, we used ping packets that were 128 bytes routing payload, i.e., without routing-header overhead. For each ping packet that arrived at the destination, an echo packet of the same size was sent back to the source[8]. GPSR, as well as location service headers, contained all fields necessary for a real-world implementation with appropriate field sizes (e.g., 4 bytes for the id or 3 bytes per location coordinate). The chosen field sizes match the default sizes of DSR as closely as possible to guarantee comparability.

Even though all code was taken from and written for the ns-2 version 2.1b8a, we replaced the MAC 802-11 implementation with a bug-fixed version from the ns-2 distribution 2.1b9 in which we, ourselves, fixed one more bug. The MAC had a data rate set bandwidth of 2 $\frac{MBit}{s}$ and a basic rate set bandwidth of 1 $\frac{MBit}{s}$.

For the setup of GPSR, please refer to Table 3.2.

The GLS parameters were chosen to correspond to those used in [204] as closely as possible. We even set the grid size to 250 m, even though we do not use two-hop neighbor tables.

[7] As outlined in Section 2.2.3, this model has to be handled carefully. However, we did so by disallowing pause periods and modeling increasing mobility solely by increasing the average movement speed.

[8] However, no echo packet is sent if the same ping packet arrives a second time, which might happen if packet duplication occurs.

implementation	ported from Brad Karp ([171])
perimeter mode	off
beacon piggybacking	on
beacon interval	2 s

Table 3.2: `GPSR` simulation parameters

In `RLS`, caching and radial flooding were always enabled. Cached replies were disabled because we considered it unnecessary for the given traffic patterns in conjunction with the use of binary flooding. Binary flooding, as described in Section 3.1.1, was used for most simulations, but some results for exponential flooding are also provided and will be explained here. We also did simulation runs with enabled broadcast suppression, because even though the broadcast-storm problem is not acute for the given traffic patterns, we wanted to know how much of an impact suppression would have on delivery and latency.

Other parameters we used in the `RLS` configuration can be found in Table 3.3.

Time-to-live:	
Max.Packet `TTL`	64 hops
Max.Query Distance (d_{max})	32 hops
Timeouts:	
Query Cycle Timeout	5 s
Location Cache Timeout	5 s
Sequence Number Cache Timeout	10 s
Broadcast Suppression:	
Random Assessment Delay (`RAD`)	0.02 s
Max.Receive Count (c_{max})	4
Distance Threshold (d_{thres})	45 m

Table 3.3: `RLS` parameter list

We measured delivery ratio, packet overhead, average single hop latency and average route lengths. Delivery ratio is given as the percentage of ping packets that were successfully delivered to the destination. Each echo is triggered by a ping packet and uses the route built just by the corresponding ping. To avoid conditional probabilities, the statistics thus considers the pings only. However, one should keep in mind that echo packets transport information back to the source node and thus improve the performance of location-based protocols. The packet overhead denotes the average number of kilobytes of routing protocol packets that were sent or forwarded throughout a simulation on the network layer. This means it represents the mere routing protocol overhead and not the kilobytes consumed by the ping and echo packets.

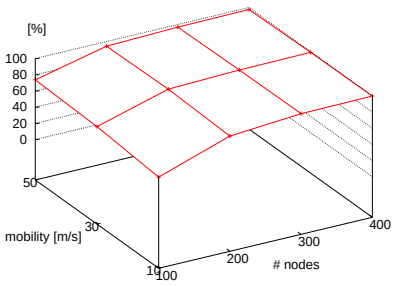
What we call the average single hop latency represents the total delay of a delivered ping packet from the source to the destination, normalized by the number of hops it has taken. Thus, it includes the delay in the route set-up phase, which is an order of magnitude higher than normal network layer hop-to-hop delays. This definition enables us to measure the quality of the route set-up, and underscores the importance of keeping intact routes in mobile scenarios, because any route break results in additional set-up phases that increase this value. Since the average single hop latency depends heavily on the number of delivered packets, as well as in the number of hops taken, we also measured the average route lengths and will use them, as well as the delivery ratio, to interpret the single hop latency graphs in Section 3.1.2.

Results

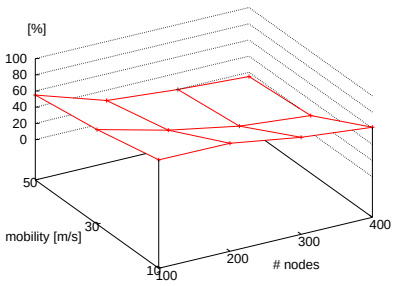
Figure 3.1 shows the average delivery ratios for all evaluated routing schemes (i.e., all combinations of GPSR with a location service, as well as DSR).

As can be seen in in the figure, DSR has an advantage of approximately 20% in the 100 nodes scenario at 10 $\frac{m}{s}$ movement speed, while its performance rapidly decreases for denser networks and higher speeds. In fact, the 100 nodes 10 $\frac{m}{s}$ case is the only one where DSR is not outperformed by GPSR/RLS. In scenarios with a lower mobility, routes found by DSR tend to be stable for quite some time, if not even for a whole connection. However, as movement speed rises, routes break often because the topological neighbors move away. This forces DSR to look for new routes, thereby increasing the load on the network. Since the main reason for low delivery rates is a congested network (as can be seen by correlating Figure 3.1 to Figure 3.2), heavy load beats down DSR's performance in fast-moving scenarios to delivery ratios of only 14%-50%. But DSR also has difficulties to scale well with rising node densities, which, again, is due to network congestion. This time the congestion is generated by the route request packets because they increase in size while being flooded throughout the network, and thus can get very large. Since all nodes participate in the flooding process, this means that many nodes in a close range try to forward these large route requests; when the node density rises, this quickly produces high stress for the wireless medium, which then may rapidly exceed its capacity.

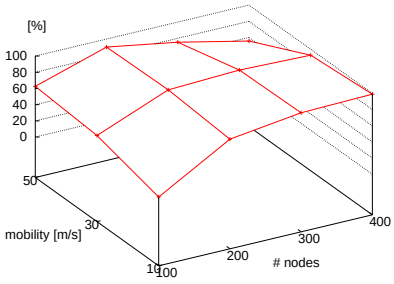
RLS combined with greedy position-based routing does better for both of these problems that limit the scalability and achieves delivery ratios of $> 90\%$ in all cases except for the 100 nodes scenarios. Position-based routing schemes – in comparison with the source routing approach – profit from high movement speeds because changing topology not only breaks routes on a regular basis, but also creates new ones. Since this type of routing scheme also does not care which neighbor forwards the packet as long as it progresses in the right direction, new routes are used more



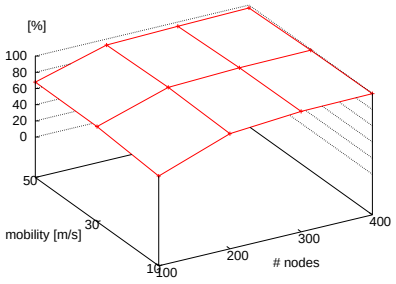
(a) RLS



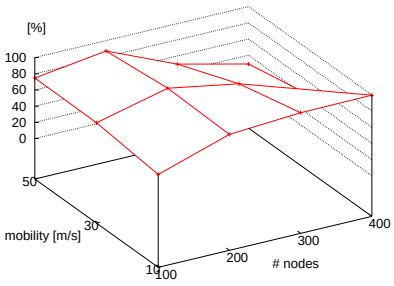
(b) DSR



(c) GLS

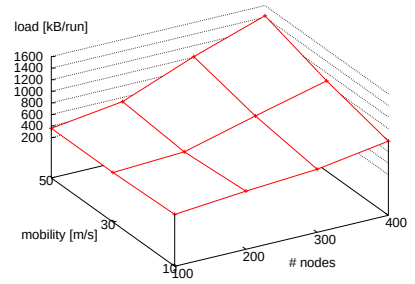


(d) OLS

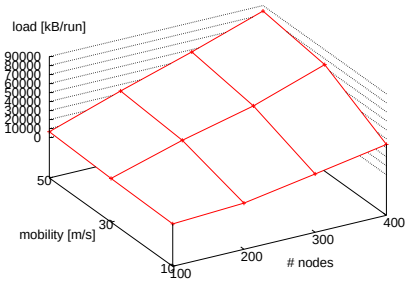


(e) RLS (Exponential Flooding)

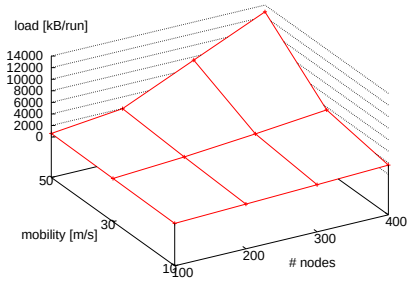
Figure 3.1: Ping delivery ratios for DSR and GPSR with different location services



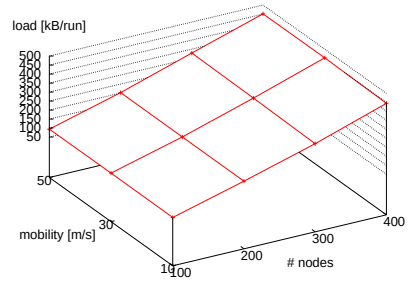
(a) RLS



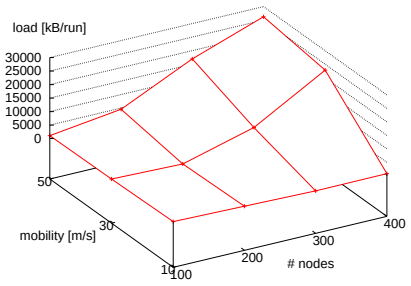
(b) DSR



(c) GLS



(d) OLS



(e) RLS (Exponential Flooding)

Figure 3.2: Packet overhead for DSR and GPSR with different location services

flexibly than in DSR, thus saving discovery overhead. Dense networks do not pose a problem either because position-based routing selects one neighbor to forward a packet, regardless of how many neighbors a node has. Finally RLS, while using a flooding scheme, only floods small constant sized packets that do not stress the wireless medium as much as DSR's route requests do.

However, the reason, why GPSR/RLS does not beat DSR in the 100 nodes scenario at movement speeds of maximally 10 $\frac{m}{s}$ and only achieves a 60%-75% delivery ratio in 100 nodes scenarios with higher movement speeds is also the greedy position-based routing scheme of GPSR. To successfully route a packet, GPSR needs greedy connectivity. Greedy connectivity denotes the subset of total connectivity[9] that represents all nodes reachable from a given source node by use of a greedy heuristic. Since the greedy heuristic might get stuck in local maxima, it can fail to find connectivity even if it exists. In scarce scenarios, like 100 nodes in a 4 km^2 square, greedy connectivity is relatively low, and many holes[10] exist that keep a simple greedy routing mechanism from successful delivery. Figure 3.1(d) demonstrates this by showing the delivery ratio that a greedy forwarding strategy can achieve if it has perfect location information at its disposal . This also means that a recovery scheme like the perimeter mode for GPSR, presented in [171], would certainly improve the rates for GPSR/RLS.

When compared to GPSR/GLS, we see that GLS behaves similarly to RLS (or to OLS, which can be seen as a benchmark), with the exception of a slight decrease for 100 nodes at 10 $\frac{m}{s}$, and a decline in performance for dense high-speed scenarios. The first is due to greedy connectivity; only in GLS is the effect magnified because location queries are also sent as unicast packets by means of greedy forwarding. The reason for its weakness in dense high-speed scenarios is the fact that in GLS, each node needs to send so-called position update packets to all nodes that can be queried for its position (its location servers). If nodes move at high speed, the frequency of these updates rises, and in dense scenarios many nodes try to send them at the same time, thus congesting the network. For a closer look at GLS, refer to [194⋆].

We also included a comparison of RLS with binary flooding to RLS with exponential flooding (Figures 3.1(a) and 3.1(e)), because it shows that trying to reduce the overall network load by keeping query floods in as small a subset of the network as possible can be quite harmful if the communication patterns have no preference for localized traffic. The multiplication in query packets creates an up to 20 times

[9] Total connectivity is the transitive hull of the network graph.

[10] A hole is an area of at least the radio range in diameter that is devoid of nodes and thus two bordering nodes at opposite sides of this hole are unable o communicate.

higher load that soon exceeds the capacity of the wireless medium (as can be seen in the comparison of Figures 3.2(a) and 3.2(e)).[11]

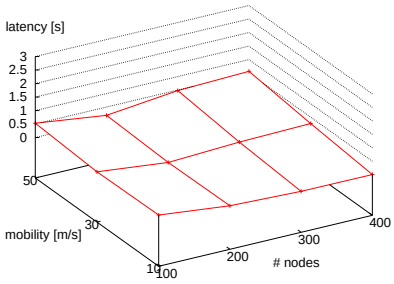
Figure 3.2(d) depicts the amount of traffic generated by GPSR beaconing, which is not related to the speed at which nodes move at but only to the number of nodes that periodically generate them. Since OLS does not generate any overhead, GPSR beaconing is the main determining factor.

All other graphs in Figure 3.2 show the same basic behavior, i.e., an increase in packet overhead for denser and very mobile scenarios, on different scales. Correlated to the delivery ratios in Figure 3.1, we see that a routing scheme achieves good ratios only as long as its protocol overhead does not congest the network. Achieving scalability is therefore a matter of keeping the overhead low in the targeted application area. All GPSR-based routing schemes have an overhead of 90-450 $kByte$ per simulation run in the scenarios we chose, which is well below the network capacity. Thus, the location service has to be the attention focus of the design. RLS with binary flooding never exceeds 1600 $kByte$ per simulation run and outperforms GLS, as well as DSR. But RLS with exponential flooding shows that RLS is only as effective as its flooding strategy. GLS, though unoptimized, does not seem to be suited for fast-moving scenarios, and DSR seems best suited for small and slow scenarios.

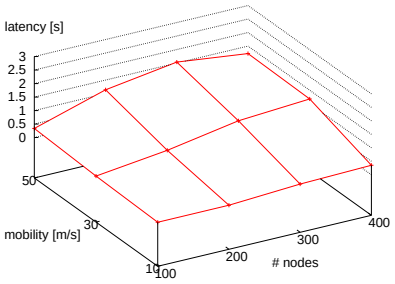
The graphs depicted in Figure 3.3 show the average single-hop latency, i.e., the average amount of time a ping packet needs to get from one node to the next on its route to the destination. Only pings that arrive at the destination were used for these graphs, thus, the values may misrepresent the average for (node number, movement speed)-pairs in those cases in which the delivery ratio is quite low. For example, DSR delivers only 14% of the ping packets in the 400 nodes scenario when nodes move at 50 $\frac{m}{s}$. This influences the single-hop latency, because those pings that are delivered also tend to be very close to the source (according to Figure 3.4(b), delivered ping packets have only taken about three hops for the DSR case we just mentioned).

RLS achieves single-hop latencies of 0.01-0.5 seconds, with the best (i.e., smallest) delays in networks that are moderately populated and moving at moderate speeds. These delays are only two to three times higher than those of pure greedy routing, which is due to the fact that the small query packets need at least one shortest round trip time to find the destination and get the information back to the source. However, Figure 3.3(e) shows that delays can be kept small only as long as the network is not congested, in which case single-hop latency gets unacceptably high. The same is true for GLS and DSR. Please note that Figure 3.3(b) shows a slightly decreasing latency for the high density / high mobility case. This is due to the fact

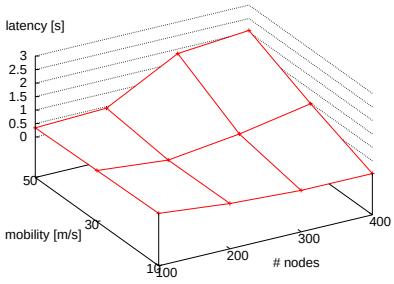
[11] It remains for future work to evaluate how RLS with exponential flooding behaves if used in extremely large scenarios with localized traffic patterns. These scenarios emphasize the definition of localization, because more disjoint small areas may exist in which nodes have to communicate in a multi-hop fashion. The impact of broadcast suppression also remains to be studied.



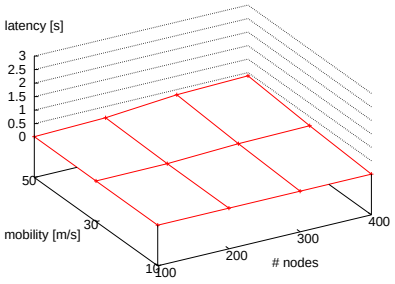
(a) RLS



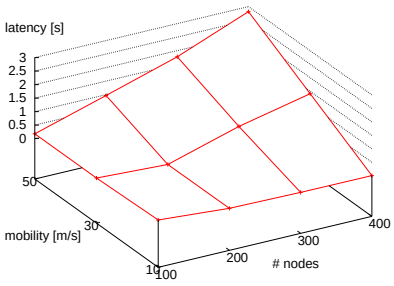
(b) DSR



(c) GLS



(d) OLS



(e) RLS (Exponential Flooding)

Figure 3.3: Single hop latency for DSR and GPSR with different location services

that in this case, the communication that takes place is increasingly local, causing lower distance and thus lower latency.

One thing that is noteworthy and can be observed in all location services that employ greedy forwarding, except for OLS, is the fact that single hop latency is lower for scenarios with 200 nodes as opposed to scenarios with 100 nodes. For DSR, this effect is less noticeable. While researching this phenomenon we observed that it is based on a lack of connectivity. All location services, except for OLS, which delivers location information whether or not the queried node is reachable, need to reach the destination or a location server with a query and use timeouts to retry if they do not receive an answer. In scarce scenarios, nodes may not temporarily be reachable because of network partitioning. This introduces a delay that cannot be avoided by any routing scheme or location service. In scarce scenarios, this temporary partitioning occurs more frequently and lasts longer, thus producing higher delays for packets. Since DSR makes use of total connectivity, it suffers less from this phenomenon than GPSR, which considers the network partitioned more often due to usage of greedy connectivity.

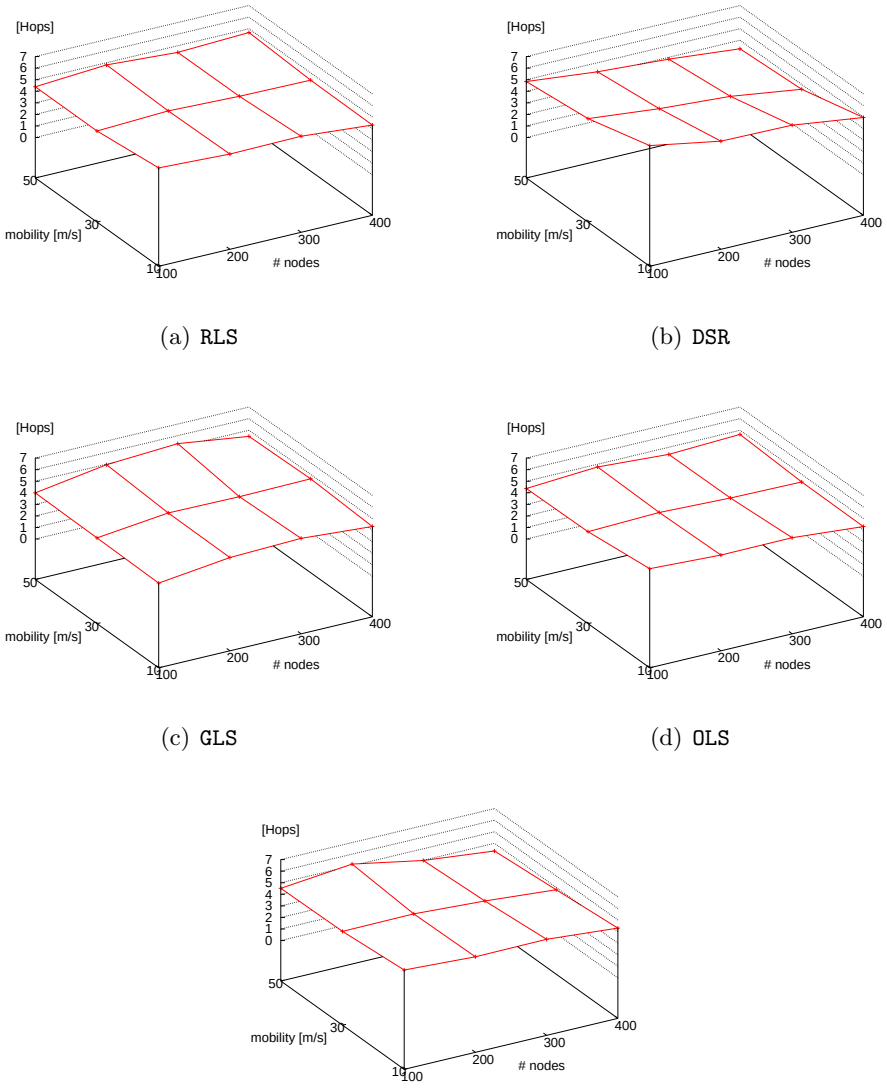
To illustrate this, we provided the single-hop latency spectra for the 100 nodes scenario in Figure 3.5 for RLS and DSR. It is easy to see that while the majority of packets have very low latencies, some packets have latencies that are orders of magnitude higher and influence the average. We also see that greedy worst-case latency is roughly twice the size of the worst-case latency for DSR. We should also note that the default packet retention time of 30 seconds in DSR, which we adapted in GPSR for comparability, is quite unrealistic for real world scenarios where packets this old would probably be dropped. With 200 nodes in a 4 km^2 square, density has improved enough for greedy connectivity to be very close to total connectivity, which causes the drop in average single-hop latency. The following increase is then due to network load and represents the delay acquired in the MAC 802-11 layer during the network traversal.

To better understand the single-hop latency, one should also take a look at Figure 3.4, which depicts the average route length in hops that ping packets have taken to the destination. Three things are worth mentioning:

1. RLS has a nearly constant route length of 4-5 hops and thus stays close to the greedy route length determined by OLS.

2. Route length decreases with the delivery ratio for all schemes because the number of reached, distant destinations decreases since longer routes are harder to maintain.

3. Conversely, we see that DSR has route lengths of 5-7 hops in scarce, slow-moving scenarios, which represent connectivity that is not greedy connectivity



(a) RLS

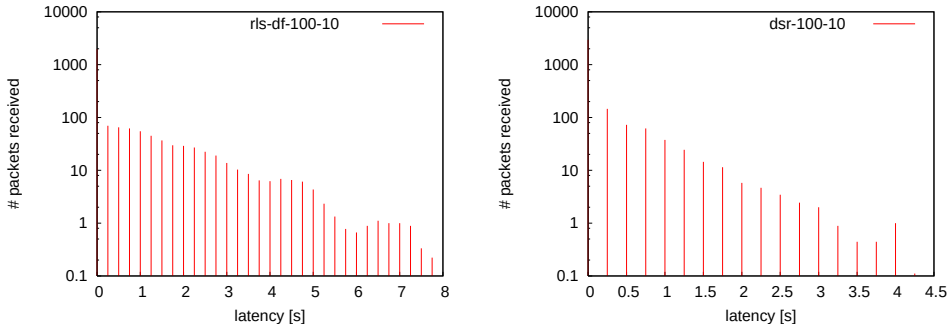
(b) DSR

(c) GLS

(d) OLS

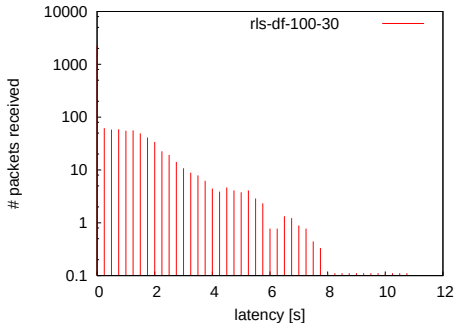
(e) RLS (Exponential Flooding)

Figure 3.4: Average route length for DSR and GPSR with different location services

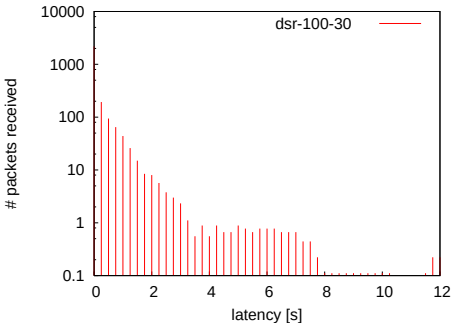


(a) RLS 100 Nodes 10 $\frac{m}{s}$

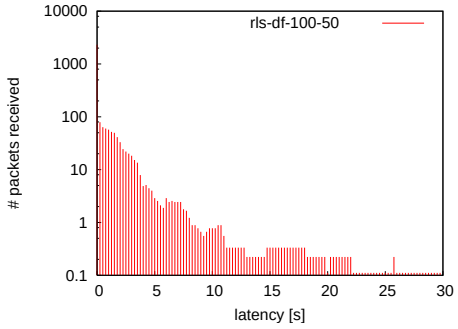
(b) DSR 100 Nodes 10 $\frac{m}{s}$



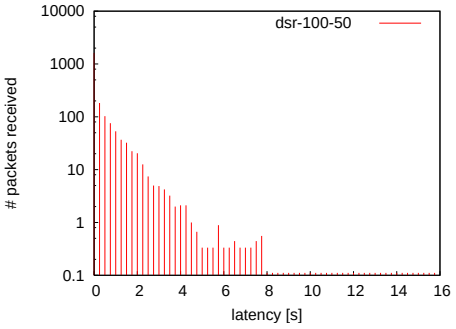
(c) RLS 100 Nodes 30 $\frac{m}{s}$



(d) DSR 100 Nodes 30 $\frac{m}{s}$



(e) RLS 100 Nodes 50 $\frac{m}{s}$

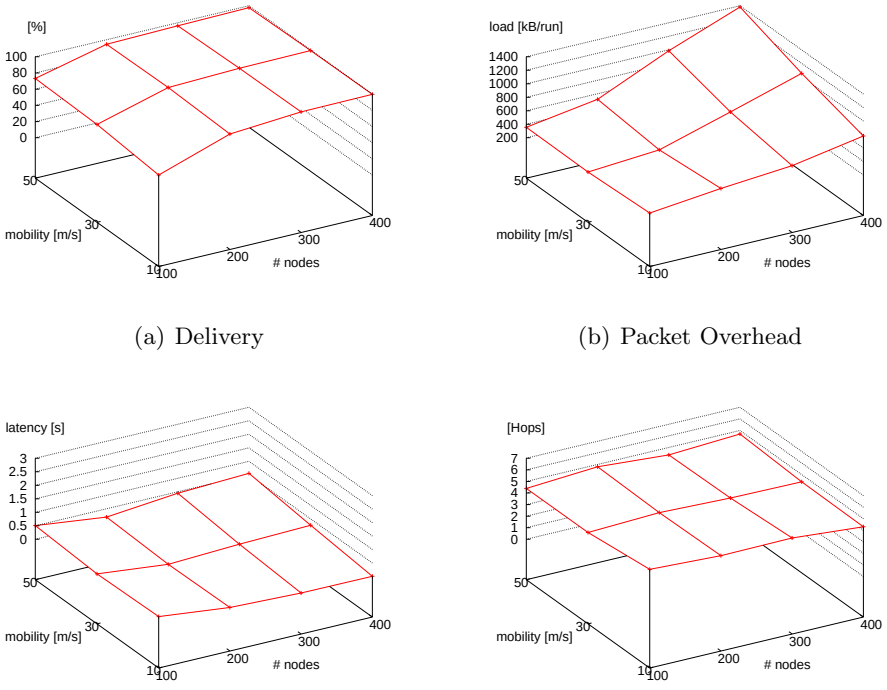


(f) DSR 100 Nodes 50 $\frac{m}{s}$

Figure 3.5: Single hop latency spectra for DSR and GPSR/RLS in a 100 nodes 10 $\frac{m}{s}$ scenario (9 run average)

(e.g., complicated routes) and cannot be found by the simple greedy routing we use.

Finally we evaluated `RLS` with activated broadcast suppression and discovered that the values were nearly identical to those of `RLS` without suppression; the deviations could not be told apart from the variations of multiple simulation runs. This leads us to the conclusion that broadcast suppression, while not needed in connection patterns that do not lead to the broadcast storm problem (like the ones we used), does not show any negative effect in scenarios with light traffic. However, its use in heavy load scenarios remains to be tested. For detailed results on broadcast suppression, see Figure 3.6.



(a) Delivery

(b) Packet Overhead

(c) Single Hop Latency

(d) Average Route Length

Figure 3.6: Evaluation graphs for `GPSR`/`RLS` with rebroadcast suppression.

3.1.3 Conclusions

This detailed simulative analysis shows that MANET algorithms tend to be rather fragile to varying network conditions. However, there is no single protocol variant superior in all tasks. Moreover, a special setting is useful for a special subset of parameters. Also, rather small 'mistakes' in protocol setup can cause the algorithm to malfunction, even if principally correct.

Nevertheless, a simple location service based on flooding has been shown to be feasible, especially for higher mobility. In this sense, we have shown that Brad Karp's original GPSR [171] works, even without the unfair advantage of knowing the destination node's position. Even then, it still outperforms DSR for high-mobility scenarios.

3.2 A Hierarchical Location Service for Mobile Ad-Hoc Networks

The last section's RLS is designed for simplicity and mobility resilience rather than for scalability. The flooding part of the algorithm has problems with big networks and with non-local communication, causes high momentary packet loads. Designing a location for spatial / node-count scalability, our group has proposed the *Hierarchical Location Service* or HLS. The similarity in name is intentional, since it is very close to the Grid Location Service GLS [204]. In fact, it is combining GLS core ideas with the concept of the Virtual Home Region [138].

The basic operation of HLS is as follows: The area occupied by the network is divided into a hierarchy of regions called cells. On every level of the hierarchy, a hash function determines for every node a responsible cell. The node informs this responsible cell about its whereabouts, and when another node needs to know this location, it uses the same hash function to calculate the destination node's responsible cells, which overlap at the region with the smallest hierarchy level containing both nodes. In detail, HLS works as follows:

3.2.1 The Algorithm

Area partitioning

The area partitioning of the Hierarchical Location Service is more general than that of previous location services. HLS partitions the area in *cells*, the partitioning must be known to all participating nodes. The shape and size of the cells can be chosen arbitrarily according to the needs of the network. The only prerequisite is that a node in a certain cell must be able to send packets to all other nodes in this same cell. This can be either achieved by choosing an appropriate cell size,

i.e. the distance between any two points in the cell must be smaller than the radio range, or by implementing a cell-wide broadcasting mechanism. It is not required that the area in which the MANET is deployed be fully covered by cells. Thus HLS is applicable to areas containing obstacles like buildings.

The cells are grouped hierarchically into *regions* of different levels. A number of cells forms a region of level one, a number of level-one regions forms a level-two region, and so on. Regions of the same level must not intersect, i.e. each region of level n is member in exactly one region of level n+1. An example for the area partitioning with cells of similar size and form is shown in Figure 3.7.

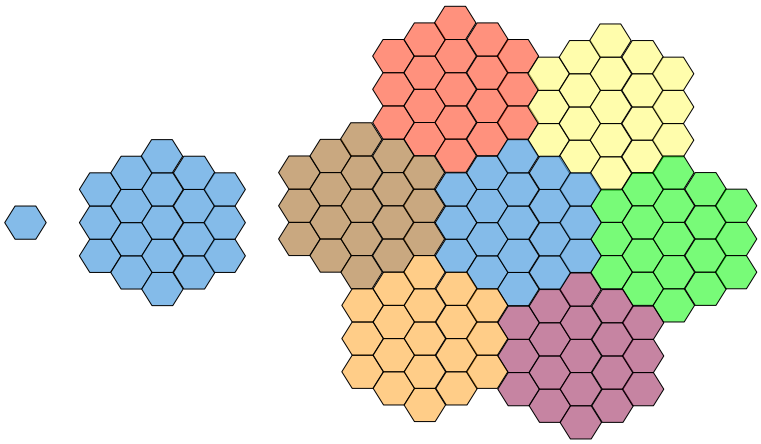


Figure 3.7: HLS cells and regions

Responsible cells

HLS places location information for a node N in a set of cells. We call these cells the *responsible cells* (RC) of N. As a first rough definition, we say "N updates its responsible cell x" when N sends an update packet to an arbitrary node L in or close to x. This node L becomes *location server* for N. It is possible that subsequent updates arrive at different nodes within that cell e.g. because nodes have moved. A cell may therefore contain multiple location servers for a node. Moreover, we assume that all necessary routing for HLS is done with a position-based routing protocol like GPSR (Section 2.5.8).

A node N selects one responsible cell for each level in the hierarchy. For a given level p, the RC is selected according to the following algorithm:

1. Compute the set $S(p, N)$ of cells belonging to the region on level p which contains N.

2. Select one of these cells with a hash function[12] which takes characteristics of S and the ID of N as input.

A possible hash function is the simple, modulo-based function:

$$RC(N, p) = Id(N) \mod |S(p, N)|$$

As a result of the above selection, N has exactly one responsible cell on each level and it is guaranteed per definition of the hash function that the RC of level p and node N share the same level-p region. An example of the selection of RCs is shown in Figure 3.8 for a three-level hierarchy. The large circles mark the regions, the cells with the numbers are the responsible cells. The node and its RC on level one share the same level-one region, the RC of level two lies within the same level-two region as the node and so on. (Please note that the ID of the responsible cell is relative to the region where the RC lies in.)

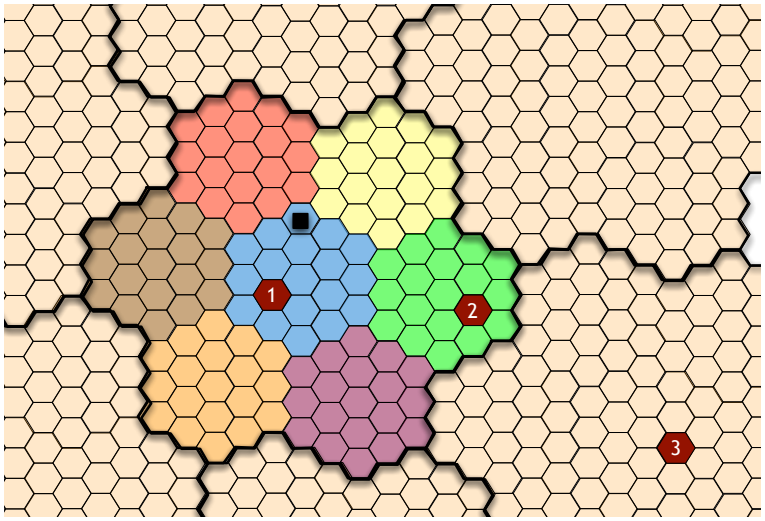


Figure 3.8: Example for responsible cells of a node

An interesting observation is the following: If we compute and mark all cells which may become responsible cells as a node moves through the network, we get a structure similar to the one shown in Figure 3.9. All cells marked here are candidates for responsible cells. These *candidate cells* are connected with arrows to visualize their hierarchical, treelike structure which we call *candidate tree*. The root of the tree is the single RC candidate on the highest level which in this example

[12] The performance of the Hierarchical Location Service depends on the hash function used to calculate the responsible cells. It should therefore be adapted to the environment in which the MANET is used.

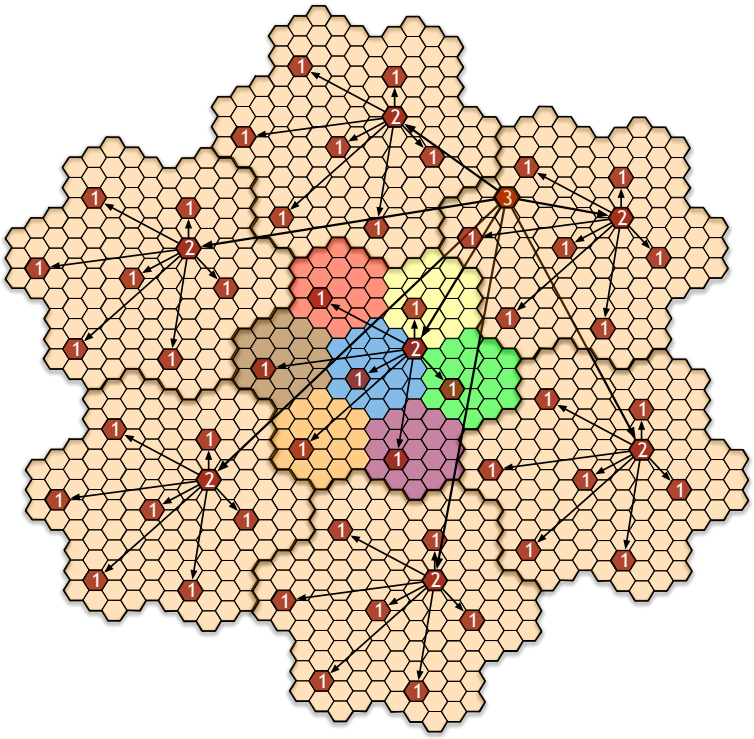


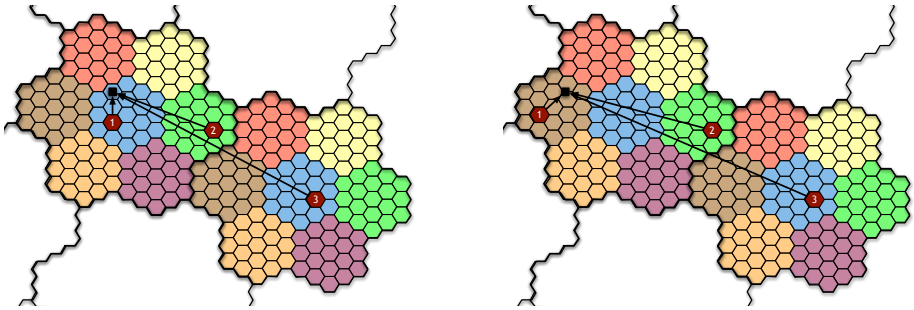
Figure 3.9: Candidate tree in a three-level hierarchy

is located near the center of the area, the leaves are the cells which are candidates for responsible cells on the lowest level. The candidate tree may be different for each node and can be computed with the hash function and the node `ID`. Selecting the responsible cells for a node N can be seen as selecting the branch in this tree which ends in the current level-one region containing N as shown in Figure 3.8.

Location update

There are two different methods for `HLS` to update location servers, the *direct location scheme* and the *indirect location scheme*.

To update its location servers according to the direct location scheme, a node computes its responsible cells as explained in Section 3.2.1. Position updates are then sent to all `RC`s at the same rate. This update scheme is called "direct" because a location server directly knows the position of the node. In Figures 3.10(a) and 3.10(b), the location information in the `RC`s is represented as a pointer to the position of the node. Figure 3.10(a) shows the responsible cells after receiving an update. The `RC`s on all levels contain exact location information about that node.



(a) before movement (b) after movement

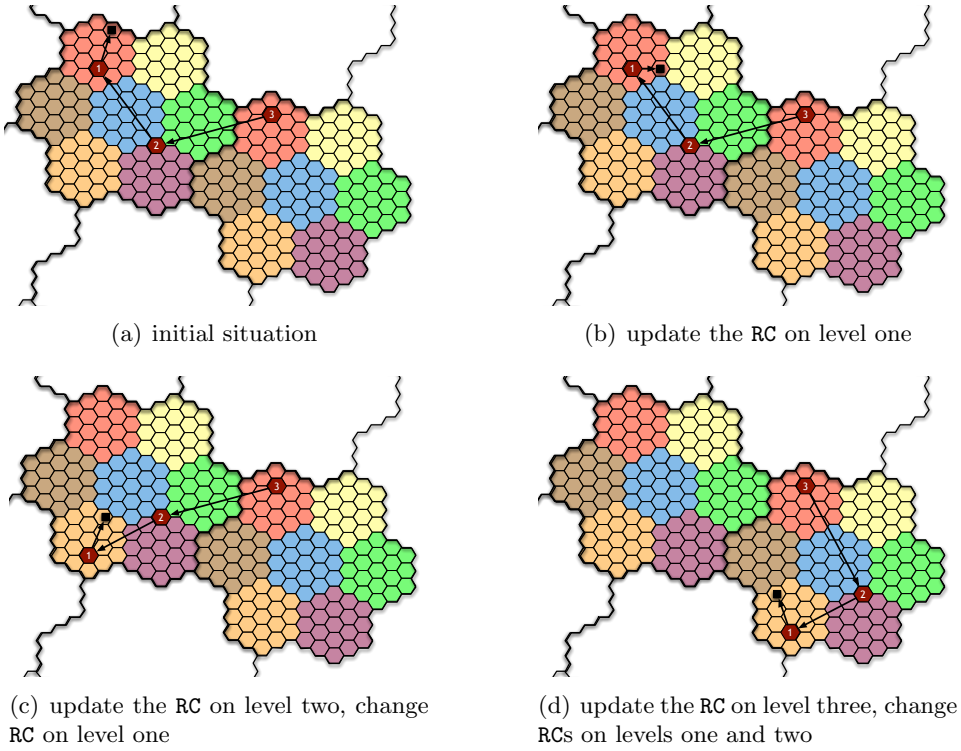
Figure 3.10: Direct location scheme

All responsible cells must be updated whenever the node has moved a distance large enough to render the position information in the location servers useless (Figure 3.10(b)). While this can be easily done for close location servers, updating the RCs on a higher level which tend to be farther away may cause a lot of traffic.

This traffic can be reduced with the indirect location scheme, where the location servers on higher hierarchy levels only know the region of the next lower level a node is located in. More precise location information is not necessary on higher levels. As shown in Figure 3.11(a), the pointers which represent the location information no longer point to the last known position. They point to the responsible cell on the next lower level. Thus, this update scheme creates "indirect" location knowledge in the location servers. In contrast to GLS, the chains established here do not consist of moving nodes but of cells with fixed positions. If the location information available in the RC on level p says "node A is in the level-$(p-1)$ region y", the next responsible cell which must be contained in y can be computed with the hash function and does not depend on the potentially outdated position of moving nodes.

Given an ideal environment with no packet loss, a location server on level n needs to be updated only when the node moves to another level-$(n-1)$ region. Thus, the responsible cell on level one will be the only cell which is updated if the node moves within the boundaries of the level-1 region (Figure 3.11(b)). Cells on higher levels need to be updated only if the RC on the next lower level changes (Figures 3.11(c), 3.11(d)). Hence, update traffic generated by a node is mostly local. The majority of the update packets have to travel only a few hops, whereas long-distance updates are rarely sent.

To overcome location server failures which can occur for both update schemes, we have chosen a soft state approach. As this leads to possibly unnecessary updates especially for the indirect update scheme, other mechanisms to overcome these losses, like the duplication of location servers, could reduce the necessary bandwidth.



(a) initial situation

(b) update the RC on level one

(c) update the RC on level two, change RC on level one

(d) update the RC on level three, change RCs on levels one and two

Figure 3.11: Indirect location scheme

Handovers

Since the identification of a location server depends only on its position, a node leaving a responsible cell can no longer be location server for information belonging to this responsible cell. In this case, the information belonging to the cell just left is handed over to this cell and treated like an update: the handover packet is forwarded to a node in or close to the cell which becomes the new location server.

Position Requests

To successfully query the current location of a target node T, the request of a source node S needs to find a location server of T. When querying the position of T, S knows the Id of T and therefore the structure of the candidate tree defined via the hash function and T's Id. It furthermore knows that T has selected a RC for each region it resides in. Thus, the request only needs to visit each candidate cell of the regions containing S. The candidate cell of the region with the lowest level

containing both S and T is by definition a responsible cell, and so are the candidate cells of all higher levels.

Therefore, S computes the cell which T would choose as responsible cell if it were in the same level-one region as S and sends its request to this cell. When the request packet arrives at the first node A within the boundaries of the candidate cell, it is processed as follows:

1. Node A broadcasts the request to all nodes within the candidate cell. This is called *cellcast request*.

2. Any node which receives this cellcast request and has location information in its location database sends an answer to A. This can also be A itself.

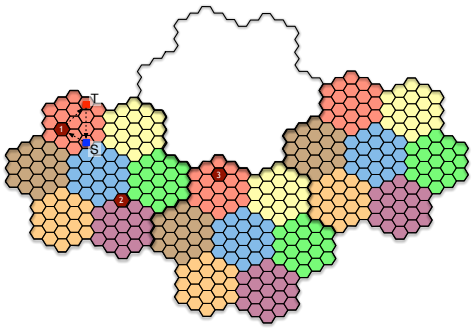
3. If A receives an answer for its cellcast request, the request is forwarded to the target node T.

4. Otherwise, it forwards it to the corresponding candidate cell on the next hierarchy level.

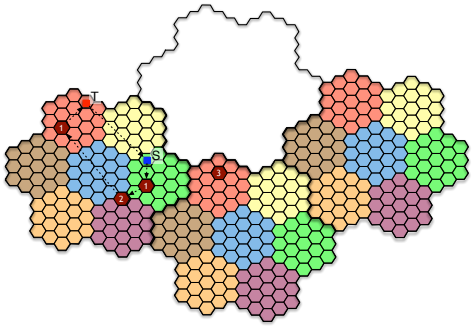
With this mechanism, the request is forwarded from candidate cell to candidate cell until it finds a location server for T or no more candidate cells are left. In the latter case, which can occur e.g. if the network is partitioned, the request has failed. Otherwise, the request can be answered either by the location server of by T itself.

The algorithm guarantees that the request is forwarded to at least one candidate cell which is also a responsible cell, the top-level RC. In more advantageous cases, the request is already forwarded to a responsible cell on a lower level. The level of the first candidate cell which is also a responsible cell for T depends on the distance between S and T. The closer the two nodes are, the earlier the branch of the candidate tree selected by T for its updates and the one calculated by S for its request will match. If r is the region with the smallest level i which contains S and T, the branches match on level i. All candidate cells computed for this request with a level greater or equal to i are also responsible cells.

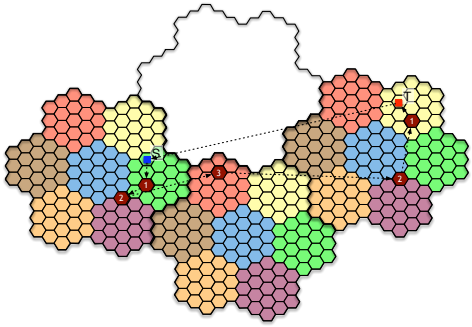
Examples on requests are given in Figure 3.12 for nodes S and T. Here, the location servers are updated according to the indirect location scheme. If the two nodes are located in the same level-1 region as shown in Figure 3.12(a), the candidate cell on level one also is a responsible cell and should contain a location server. The request can be delivered and answered directly. In Figure 3.12(b), S is located in the same level-2 region as T. The request is forwarded via the candidate cell on level one to the responsible cell on level two, which should contain a location server. In the third example presented in Figure 3.12(c), S and T are located in different level-2 regions. The request is forwarded to the candidate cells on levels one and



(a) A request from a node in the same level-1 region



(b) A request from a node in the same level-2 region



(c) A request from a node in the same level-3 region

Figure 3.12: Example requests

two, then it reaches the RC on level three and eventually finds a location server for T.

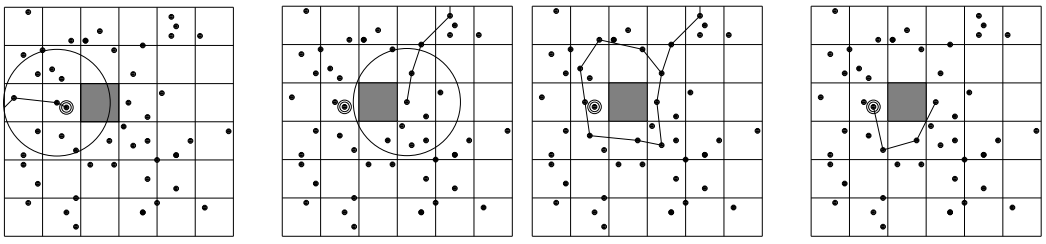
As shown in the examples, a request packet is forwarded only within the boundaries of the lowest level region where both nodes reside in. Therefore, the communication complexity of a request depends on the distance between sender and target of the request. A node needs a location server on each hierarchy level. With the number of hierarchy levels being $O(\log n)$, so is the number of location servers.

Empty cells

A problem which has not been addressed so far are empty or unreachable cells, i.e. a location update or request packet is sent to a cell which does not contain a node or which is unreachable due to a partitioned network. In HLS this problem is solved as follows: if an update cannot be forwarded to to the target cell, the node detecting this becomes temporary location server. Thereafter, the information is treated by the handover mechanism explained above like any other location information outside its target RC: the temporary location server regularly tries to hand the information over to the target cell.

If a request looks for a location server in an empty cell, it cannot be determined if the cell is a responsible cell or a candidate cell. There are two ways to proceed if a request cannot reach the cell it is sent to: either search the neighborhood for a temporary location server or forward the request to the next higher level. For HLS we have chosen a combination: for all but the highest level, requests are forwarded to the next level, while on the highest level the neighborhood of the cell is searched for a temporary location server. This does minimize the likeliness that a search has to be performed while still providing a high chance of finding a location server.

An example of the search strategy is shown in Figure 3.13 with quadratic cells[13].



(a) update stored outside of RC

(b) request does not find location server

(c) request sent to the neighbors of the RC

(d) answer packet

Figure 3.13: The area-extension mechanism

Whenever an update packet cannot reach any node within the target cell (marked dark gray), it is stored by a temporary location server (the node marked with two circles). If a top-level request is sent to the respective cell, the location server is not in the correct cell and thus may be unreachable (Figure 3.13(b)). In this case, the request is successively routed to the neighbors of the RC (Figure 3.13(c)). On overhearing the request (by using promiscuous mode), the temporary location server forwards the request to the node being searched for.

3.2.2 Conclusions

Summarizing, HLS is a flexible location server with the two extreme configurations being Homezone [138] and RLS. In Homezone, the node ID is mapped to a virtual home region to which the location updates are sent. This is equivalent to HLS with just one hierarchy level. The other extreme is RLS, represented by HLS with just one cell covering the whole network extent and without proactive location updates.

Compared to RLS, GLS and Homezone, a well-configured HLS has some very desirable properties:

- It does not use network-wide flooding
- The number of location servers scales only logarithmically with the size of the network
- It is well-suited for high node mobility
- It specifically supports non-uniform communication patterns
- It is robust to node failures

Since HLS is not the core of this thesis, we will not go further into details nor show our quantitative results. Please consider looking at [173*, 172$'$], where we present the details that have been omitted here.

Overall, GLS and now HLS proves that scalable location services are feasible, even in highly mobile networks.

3.3 Contention-Based Forwarding

Previously described position-based unicast routing algorithms which forward packets in the geographic direction of the destination require that the forwarding node knows the positions of all neighbors in its transmission range. This information on direct neighbors is gained by observing beacon messages sent out periodically

[13] Remember that shape and size of cells can be chosen arbitrarily

by each node. Due to mobility, the information that a node receives about its neighbors becomes outdated, leading either to a significant decrease in the packet delivery rate, or to a steep increase in the load on the wireless channel as node mobility increases.

In this section, we describe a mechanism to perform position-based unicast forwarding without the help of beacons. In our *Contention-Based Forwarding scheme*, or `CBF`, the next hop is selected through a distributed contention process based on the actual positions of all current neighbors. For the contention process, `CBF` makes use of biased timers. To avoid packet duplication, the first node that is selected suppresses the selection of further nodes. Since the basic scheme can lead to packet duplication, we describe methods of suppressing those. In addition to that, we compare the `CBF` schemes to standard greedy forwarding by means of simulation with ns-2.

3.3.1 Introduction

As introduced in Section 2.4.2, the general idea of position-based routing is to select the next hop based on position information such that the packet is forwarded in the geographical direction of the destination.

The most important characteristic of position-based routing is that forwarding decisions are based on local knowledge. It is not necessary to create and maintain a global route from the sender to the destination. Therefore, position-based routing is commonly regarded as being highly scalable and very robust against frequent topological changes. It is particularly well suited in environments where the nodes have access to their geographical position, such as in inter-vehicle communication [225, 148].

Position-based routing can be divided into two main functional elements: a *location service*, and a *position-based forwarding strategy*. The location service maps the unique identifier (such as an `IP` address) of a node to its current geographical position. It can be seen as analogous to the route discovery process of reactive topological routing algorithms such as `DSR` [165] or `AODV` [241]. For the remainder of this section, we assume the presence of an appropriate location service that supplies the sender of a packet with the geographical position of the packet's destination. Candidates for location services have been outlined in Sections 2.5.6, 3.1, and 3.2 above.

Position-based forwarding is performed by a node by selecting one of its neighbors in transmission range as the next hop to which the packet should be forwarded. Usually, the forwarding decision is based on the node's own geographical position,

the position of all neighbors within transmission range and the geographical position of the destination. The sender requests the position of the destination from the location service and then includes it in the header of the packet. Given this information, the node forwards the packet to one of its neighbors such that the packet progresses toward the destination. This process is called *greedy forwarding*. It might occur that there is no neighbor with positive progress toward the destination while a valid route to the destination exists. The packet is then said to have reached a local optimum. In this case, a *recovery strategy* is used to escape the local optimum and to find a path toward the destination. (see Section 2.5.8).

In all previously existing strategies for greedy unicast forwarding, the position of a node is made available to its direct neighbors (i.e., nodes within single-hop transmission range) in the form of periodically transmitted beacons. Each node stores the information it receives about its neighbors in a table,and thus maintains position information about all direct neighbors. While the beaconing frequency can be adapted to the degree of mobility, the fundamental problem of inaccurate position information is always present: A neighbor selected as a next hop may no longer be in transmission range. As will be outlined later (see Section 3.3.3, Figure 3.25), this leads to a significant decrease in the packet delivery rate with increasing node mobility, and to a heavy load on the wireless channel due to several `MAC`-layer retransmissions.

To reduce the inaccuracy of position information, it is possible to increase the beaconing frequency. However, this also increases the load on the network up to a point where the available capacity is almost exclusively used for the transmission of beacons. Alternatively, it has been proposed to hand packets back to the routing layer if the next hop is no longer available [171]. At the routing layer, the packets are then rerouted to a different neighbor. While this eliminates the problem of packet drops, the trial-and-error approach can cause even more bandwidth-consuming `MAC`-layer retransmissions. Our experiments (see Figure 3.27) indicate that under high mobility, the beacon-based forwarding approach requires on average more than three `MAC` transmissions for one single-hop packet forwarding, increasing the load on the network caused by data packets by more than a factor of three. Existing work (e.g., [171]) does not take this effect into account since there, the load is measured at the routing level instead of at the `MAC` layer. Thus, for a given packet delivery rate, the load at the `MAC` layer increases dramatically with beacon-based greedy unicast forwarding (either through an increased beaconing frequency or through trial-and-error) with increasing node mobility. In addition, a node which is forwarding a packet can select a neighbor as next hop only if the target node is contained in its neighbor table. Nodes that have just moved into transmission range and that have not yet sent a beacon are therefore not considered as next-hop nodes. This may lead to the failure of greedy forwarding, even though an appropriate neighbor is present.

In this section, we propose a novel greedy forwarding strategy for position-based routing algorithms. We call the approach *Contention-Based Forwarding* (CBF). CBF performs greedy forwarding without beacons and without the maintenance of information about the direct neighbors of a node. Instead, all suitable neighbors of the forwarding node participate in the process of the next-hop selection, and the forwarding decision is based on the actual position of the nodes at the time a packet is forwarded. In order to escape from local optima, existing recovery strategies, as mentioned in the section on related work, can either be used directly or may be adapted to be used with CBF.

CBF consists of two parts: The *selection* of the next hop is performed by means of contention, while *suppression* is used to reduce the chance of accidentally selecting more than one node as the next hop. We present three suppression strategies, each with different suppression characteristics. The results of our study show that suppression of duplicate packets works well, that CBF has packet delivery ratios similar to those of beacon-based greedy routing, and that it dramatically reduces the load on the wireless medium for a given delivery rate if node mobility is high. CBF, therefore, represents a viable alternative to traditional beacon-based greedy forwarding.

The contention process of CBF used for next hop selection represents a change in the paradigm for forwarding of packets. In traditional protocols, the forwarder actively selects the desired next hop by unicasting the packet to the corresponding MAC address. In contrast, with CBF, the responsibility for next hop selection lies with the set of possible next hops. Furthermore, if no other interaction between forwarder and next hop is required, which is the case in two of the three presented strategies, MAC layer addresses become obsolete, because the nodes are implicitly addressed by their suitability to forward.

3.3.2 CBF Algorithm

The general idea of CBF is to base the forwarding decision on the current neighborhood as it exists in reality and not as it is perceived by the forwarding node. This requires that all suitable neighbors of the forwarding node be involved in the selection of the next hop.

CBF works in three steps: first, the forwarding node transmits the packet as a single-hop broadcast to all neighbors.[14] Second, the neighbors compete with each other for the "right" to forward the packet. During this *contention period*, a node

[14] In general, this should require resources similar to those required for a single-hop unicast transmission, except that packets for other nodes cannot be discarded at the network interface but have to be passed up the protocol stack. Depending on the physical and MAC layers, there may be further differences between unicast and broadcast (e.g., in IEEE 802.11 the sleep mode may not be applicable).

determines how well it is suited as a next hop for the packet. Third, the node that wins the contention *suppresses* the other nodes, and thus establishes itself as the next forwarding node.

In the following, we describe in detail how contention can be realized on the basis of biased timers. Furthermore, we present three different suppression strategies.

Timer-Based Contention

The decentralized selection of one node out of a set of nodes is a common problem encountered in many areas of computer networks. It is known as feedback control in group communication [232, 122], or as medium access control in (wireless and wired) local area networks such as IEEE 802.11 [61].

A standard approach to this selection is by means of timers. In its most simple form, timer-based contention requires that each node sets a timer at a random value. Once the first timer expires, the corresponding node responds. All other nodes receive the message, their timers are canceled, and their responses are suppressed.

It is important to realize that with this contention algorithm more than one node may respond, even if a 'good' suppression mechanism is used. This will happen if the difference between the timeout value of the earliest timer and some other timer is less than the time required for suppression. Therefore, the interval from which the timeout values are selected should increase in duration with the number of competing nodes. It was shown in [232] that, compared to uniformly distributed timers, exponentially distributed random timers can further decrease the number of responses.

To use such a simple timer-based mechanism for the forwarding decision, all nodes that receive the packet check if they are closer to the destination than the forwarding node. If this is the case, a random (exponentially distributed) timer is set to start the contention, and the node to respond first is selected as the next hop.

The problem of the simple timer-based contention is that all nodes which are located closer to the destination than the forwarding node are treated equally. Thus, a node providing minimal progress would have the same chance of being selected as the next hop as a node providing a greater progress. We therefore propose that rather than selecting random timer values, these should be determined based on how much progress toward the destination a node provides, instead of setting them randomly.

To greedily minimize the remaining distance to the destination, the progress P is defined as[15]

$$P(f,z,n) = \max\left\{0, \frac{dist(f,z) - dist(n,z)}{r_{\text{radio}}}\right\}$$

given f as the position of the forwarder, z the position of the destination and n the position of the considered neighbor. *dist* is defined as the Euclidean distance between two positions, and r_{radio} is the nominal radio range.

Figure 3.14 illustrates how well suited a node is as the next hop, depending on its location. A progress value (P) of 0 indicates that a node is unsuitable while a value of 1 is optimal and is reached if the node is located at the intersection of the circle delineating the transmission range of the forwarding node and the line from the forwarding node to the destination. Thus, P increases linearly from 0 to 1 with the progress that a node at this position would provide for the packet.

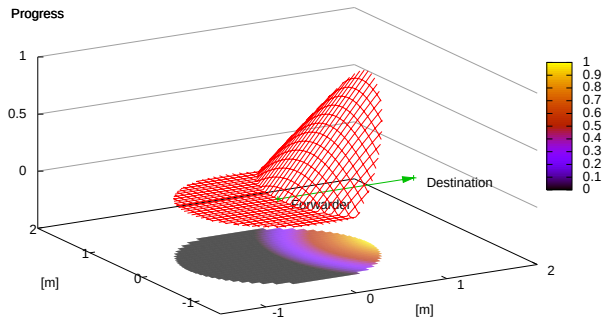


Figure 3.14: Packet progress (transmission range 1)

For the contention in `CBF` we select the timer run-time as

$$t(P) = T(1 - P)$$

where T is the maximum forwarding delay. This ensures that the node offering the greatest progress is selected as the next hop. Since the run-time of the timer depends only on the remaining distance to the destination, it is identical for all nodes that are located on the same circle around the destination.

A packet duplication may occur in the following situation: If the best-suited node has a progress of P_1 and there exists at least one node with a progress of P such that $t(P) - t(P_1) < \delta$, where δ is the minimum time interval needed for suppression,

[15] Note that the original definition of progress in [282] is different from ours since in [282] an additional projection onto the line crossing f and z is used.

then at least one packet duplication occurs. All nodes with progress P and

$$P_1 \geq P \geq 1 - \frac{\delta + T(1 - P_1)}{T} = P_1 - \frac{\delta}{T}$$

are within this so-called *duplication area* and cannot be suppressed, as shown in Figure 3.15.

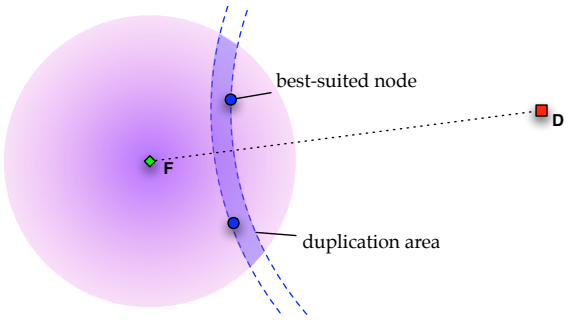


Figure 3.15: Duplication Area

An interesting property of the duplication area is that it shrinks the closer the best-suited node is located to the destination. As long as the positions of the nodes are uniformly distributed, this reduces the chance of packet duplication similar to the way in which exponentially distributed random timers reduce the chance of packet duplication when compared to linearly distributed random timers.

Analytically, this property can be made explicit via the probability density function (PDF) of the progress of a randomly selected point within the forwarding node's transmission range. Let d denote the distance between forwarding node and destination, and let us assume a normalized transmission range of 1. The radius r of a circle around the destination as depicted in Figure 3.15 corresponds to a progress $d - r$ for $r \in [d - 1, d + 1]$. The PDF for progress $d - r$ is given as

$$\frac{2}{\pi} \cdot r \cdot \arccos\left(\frac{r^2 + d^2 - 1}{2dr}\right) \tag{3.2}$$

Graphs of expression (3.2) for $d = 1, 2, 20$ are shown in Figure 3.16[16]. From the shape of these graphs, it can be seen that there are relatively few well-suited nodes (with a great positive progress). Setting the contention timer according to the progress will thus result in few timers with a short run-time, and many timers with a long run-time which decreases the likelihood of packet duplication.

[16]We note that this figure ignores that values below zero are unsuitable for forwarding.

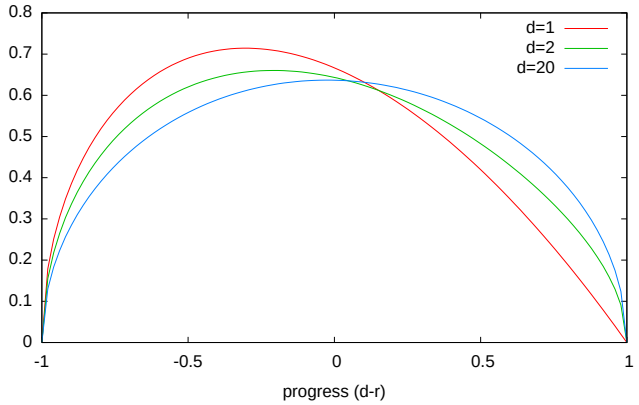


Figure 3.16: Probability Density Function of Packet Progress

Packet duplication is closely coupled with the characteristics of the MAC layer. With many MAC schemes (as, for example, IEEE 802.11), packets will be serialized; thus, packet duplication can be avoided. In wireless networks based on CSMA/CA [178], the serialization is not only performed between packets from nodes which are in transmission range of each other, but it is typically done on the basis of the interference range, which is roughly double the transmission range. As a consequence, the transmission of all neighbors of the forwarding node will be serialized since the distance between any two neighbors does not exceed twice the transmission range. If packets can be removed from the interface queue of the MAC layer, then the forced serialization can be used to eliminate the effect of packet duplication caused by the suppression delay δ, as described in Section 3.3.2. One node will be the first to forward a packet. Other nodes that have queued a duplicate of the packet may drop it once they overhear the forwarding of the packet by another node.

Suppression

Let us now assume that all neighbors of the forwarding node have set their contention timer according to their respective distances from the destination. After the first of those timers expires, a suppression algorithm aims to cancel those in all other nodes so as to prevent multiple next hops and thereby packet duplication.

Basic Suppression Scheme The most basic conceivable suppression mechanism works as follows: If the timer at a node expires, the node assumes that it is the next hop and broadcasts the packet. When another node receives this broadcast

and still has a timer running for the packet, the timer is canceled, and that node will not forward the packet.

Depending on where the initial next hop is located, other nodes may be out of transmission range and thus will not be suppressed. In the worst case, up to three copies of the packets may be forwarded, as shown in Figure 3.17. The larger the number of nodes within transmission range of the source, the higher the probability of one or more packet duplications.

It should be noted that the packet duplications described here are in addition to packet duplications caused by the length of time required for the suppression of other nodes, as described in the previous section. They do occur, even if the suppression requires no time at all.

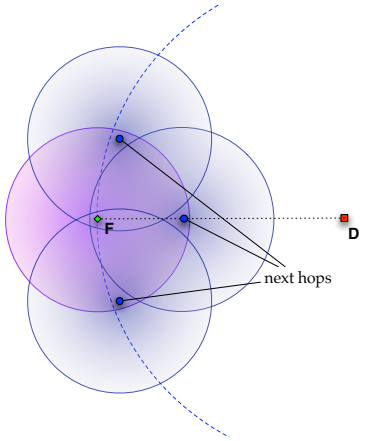


Figure 3.17: Packet duplication in the basic scheme

Area-Based Suppression In order to avoid the extra packet duplications from the basic suppression scheme, we propose to artificially reduce the area from which the next hop is selected. We call this reduced area the *suppression area* and the presented algorithm *area-based suppression*. The key idea is to choose the suppression area such that all nodes within that area are in transmission range of each other, thereby avoiding extra packet duplications as they may appear in the basic suppression scheme.

Area-based suppression requires a decision on how to choose the suppression area. One possible choice is a circle with the diameter of the transmission range located within the forwarding node's transmission range in direction of the destination (e.g., the gray circle in Figure 3.18). A circle is the geometric shape covering the largest area given that any two points within the shape are no farther apart

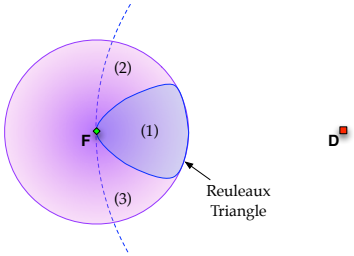


Figure 3.18: Forwarding areas

than the transmission range. If the nodes are uniformly distributed this means that on average the circle will contain the highest number of neighboring nodes when compared to other shapes where the distance between any two points does not exceed the transmission range. However, several parts of the forwarding area which make good forwarding progress are not included in the circle. A different shape where any two points are no further apart than the transmission range, the Reuleaux triangle [140], much better covers the area with good forwarding progress (see Figure 3.18).[17] By using the Reuleaux triangle with a width of the transmission range, we trade off the number of nodes contained in the suppression area against the inclusion of better-suited nodes.

The motivation for using the Reuleaux Triangle is illustrated in Figure 3.19. The curve titled "total" is the probability density function for the progress of nodes with positive progress. The curve "circle" denotes the fraction of the density "total" for a neighbor with progress p to be contained in the circle. The same applies for the "Reuleaux" curve and the Reuleaux triangle. Between 60% and 100% progress, the Reuleaux triangle covers more of the neighbors than the circle and above approximately 80%, the Reuleaux triangle covers all of the neighbors with this progress. Therefore, it is more likely to include a node with good forwarding progress.

Given the Reuleaux triangle as the suppression area, the suppression algorithm works as follows:

- The forwarding node broadcasts the packet.
- Only the nodes contained in the Reuleaux triangle participate in the contention process.
- The node at which the timer runs out first is the next hop and broadcasts the packet.

[17] A Reuleaux triangle with a width of r can be constructed by placing three circles with radius r at the corners of an equilateral triangle with an edge length r. The intersection of the circles is the Reuleaux triangle.

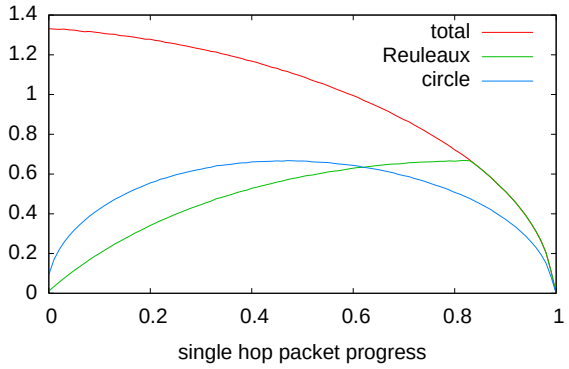


Figure 3.19: Probability density function of nodes with equal forward progress (total) and fractions contained within the circle and Reuleaux areas

- All other nodes are suppressed. Packet duplication may occur only because of the time required for suppression.

It might be possible that the only neighbors of the forwarding node that provide forward progress toward the destination are not contained in the Reuleaux triangle (1). In this case, the forwarding node will not hear another node forwarding the packet. Consequently, the process is repeated with the remaining areas (2) and (3) where nodes with forwarding progress may be located, until the forwarding node hears a rebroadcast of the packet. If no node within areas (1), (2), or (3) responds, then there is no node with positive forward progress, and a recovery strategy has to be used, just as in existing position-based forwarding schemes. The order in which areas (2) and (3) are selected when no node is located in area (1) should be chosen randomly. In this way, a tendency to always route around areas with little or no coverage in the same direction is avoided.

The key advantage of area-based suppression is the reduction of packet duplications. This comes at the cost of requiring up to three broadcasts in order to forward a packet. However, it is important to realize that as the number of nodes increases, the likelihood decreases that more than one broadcast will be required. Furthermore, the Reuleaux triangle covers the largest of the three areas and therefore has the highest probability of containing a potential next hop.

Active Selection

While area-based suppression eliminates the packet duplications caused by nodes not being in transmission range of each other, it does not prevent packet duplications caused by the time required to perform the suppression. Active selection

of the next hop prevents all forms of packet duplication, but at the cost of additional control messages. It is inspired by the Request To Send, Clear To Send (RTS/CTS) MACA-scheme proposed in [168], and is used (as a variant) in IEEE 802.11 (see [1, 61]).

The scheme works as follows: The forwarding node broadcasts a control packet called RTF (Request To Forward) instead of immediately broadcasting the packet. The RTF contains the forwarding node's location and that of the final destination. Every neighbor checks whether or not it provides forward progress for the packet announced by the RTF. If this is the case it sets a reply timer according to the basic suppression scheme. If the timer runs out, a control packet called CTF (Clear To Forward) is transmitted to the forwarding node. The CTF packet contains the position of the node sending the CTF. If a node hears a CTF for the packet, it deletes its own timer and is suppressed.

The forwarding node may receive multiple CTF control-packets. From among all neighbors that have transmitted a CTF packet, it selects the node with the greatest forward progress and transmits the packet to this node using unicast. An additional benefit of active selection as opposed to basic and area-based suppression is that it may be integrated with RTS/CTS schemes in order to avoid the "hidden terminal problem".

Active selection prevents all forms of packet duplication, even though multiple nodes may send a CTF control packet. The forwarding node acts as a central authority, deciding which node is selected as the next hop. This comes at the cost of additional overhead in the form of RTF/CTF control packets.

Summarized, CBF shows advantages over existing greedy forwarding strategies in two important aspects:

1. *Use of accurate position information*: In CBF, each neighbor uses the (very accurate) position information it has about itself to determine if it should become the next hop for a given packet. For a given delivery rate, the required bandwidth for CBF therefore does not increase with node mobility (i.e., neither an increased beaconing frequency, nor trial-and-error is needed). In addition, CBF always bases the selection of the next hop on *all* direct neighbors, including those that have just moved into transmission range, leading to an optimal utilization of available forwarders. This makes it especially attractive for MANETs with high mobility.

2. *Elimination of beacon overhead*: Removing the beacons eliminates a major part of the routing overhead that occurs independently of the actual data

traffic. This includes the bandwidth used for the transmission of beacons[18] and the memory required in the nodes to store neighbor information.

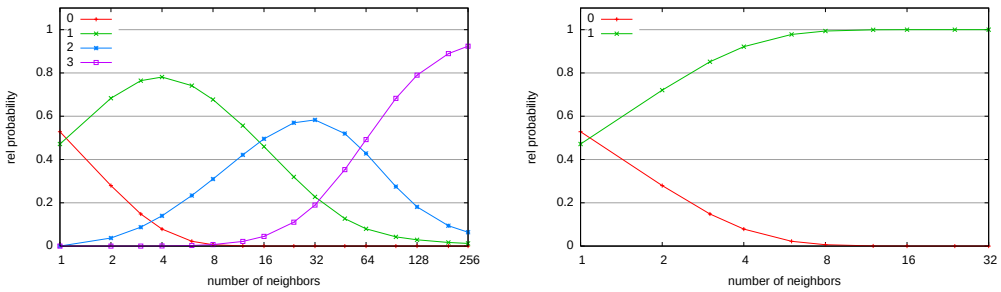
3.3.3 Performance Analysis

The most important characteristic of the different algorithms is the packet duplication probability. Furthermore, it is interesting to see how much message overhead and time are required to forward a packet from hop to hop. In the following, we determine the likelihood of packet duplication, and the forwarding delay for each of the three suppression schemes.

For the analysis, the following model was used. Without loss of generality, the forwarding node is located at position $(0, 0)$, and the transmission range is set to one. The position of the final destination is $(d_x, 0)$, with d_x exponentially distributed with parameter $\beta = 4$ in $(1; \infty)$. Neighbor nodes are sampled similarly with the number of neighbors between 1 and 256. The timer used for contention is calculated by each neighbor n with forward progress as

$$t(n) = T \left(1 - \frac{\sqrt{(d_x - n_x)^2 + (d_y - n_y)^2}}{\sqrt{{d_x}^2 + {d_y}^2}} \right),$$

where T is the maximum response time and $t(n) \in [0; T]$.[19]



(a) Basic suppression scheme

(b) Area-based suppression and active selection

Figure 3.20: Relative probabilities of n next hops ($\delta = 0$)

[18] While some existing `MAC` protocols do require beacon messages (e.g., for synchronization purposes), the overhead incurred by these beacons is very small compared to that required for beacon messages used to build up neighbor tables.

[19] For a reasonably low variance, each simulation was run 10^6 times. As pseudo-random number generator, the "Mersenne Twister" [218] as implemented in the popular C++ library framework *boost* [5], was used.

Average Number of Next Hops

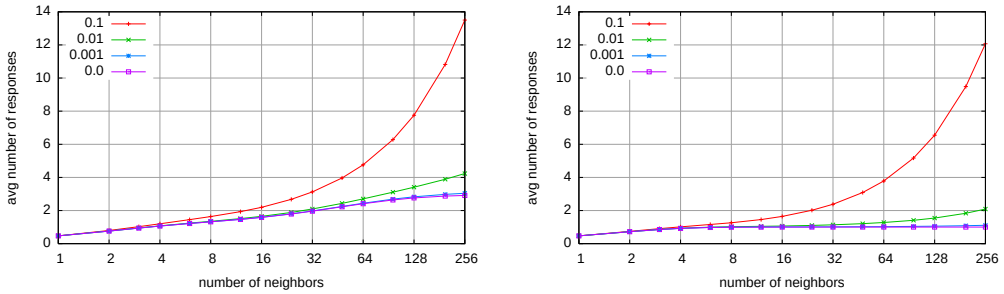
The simulation results regarding the probability of packet duplication for the three algorithms are presented in Figure 3.20. In the simulations, there is no suppression delay ($\delta = 0$) and no node mobility.

For the basic suppression scheme, there are at most three next hops, and packet duplication can occur only because nodes are further apart than the transmission range and thus do not suppress each other. With a growing number of neighbors, the probability of "no next hop" approaches zero, while the likelihood of packet duplication (2 or 3 next hops) increases. With the basic suppression scheme, the probability of a single next hop reaches a maximum for approximately four neighboring nodes. With more than 9 neighbors, packets are duplicated with a probability of more than 0.5.

In area-based suppression, packet duplication can occur only due to suppression delay or to node mobility. This is confirmed by the simulation results presented in Figure 3.20(b). The curve for "no neighbors with forward progress" quickly drops to zero as the number of neighbors increases, and in most cases exactly one node will forward the packet.

For the active selection scheme there can be no packet duplication at all, since the forwarder is the final arbiter for the decision about which neighbor is selected as the next hop. This comes at the cost of additional overhead. The overhead consists of one `RTF` control packet transmitted by the forwarder and of one or more `CTF` control packets transmitted by the neighbors. The number of `CTF` control packets generated is the same as the number of unsuppressed nodes in the basic suppression scheme and can thus be seen in Figure 3.20(a).

Impact of the Suppression Delay δ



(a) Basic suppression scheme

(b) Area-based suppression

Figure 3.21: Average number of next hops for increasing suppression delay δ

For the basic and for the area-based scheme, packet duplication can occur even if neighbors are within each others transmission range, as long as they are contained in the duplication area. The size of the duplication area depends on the time required for the suppression, causing an increase in packet duplication probability with increasing suppression delay. In Figure 3.21(a), the average number of next hops for different suppression delays is shown for the basic scheme. While a suppression delay of $0.001T$ and $0.01T$ affects the duplication of packets only marginally, a suppression delay of $0.1T$ causes significant packet duplication, even for low numbers of neighbors. Hence, given a certain (MAC-dependent) suppression delay, T should be chosen as a large multiple of δ if the basic suppression scheme is used.

The number of duplicates is much lower when area-based suppression is used. Also, there is no significant increase in the number of next hops as long as δ is a small fraction of T. Only for $\delta = 0.1T$ is there a noticeable increase in duplicate packets as shown in Figure 3.21(b).

As discussed before, due to the suppression delay, active selection will not cause packet duplication.

Forwarding Delay

With respect to delay, the basic suppression scheme is faster than the other two alternatives. The only delay introduced is caused by waiting for the first neighbor to forward the packet, as depicted in Figure 3.22.

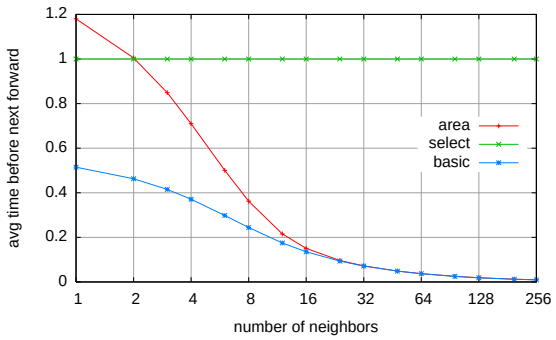


Figure 3.22: Average time before next forward

With the area-based suppression scheme, it is possible that no node with forward progress is contained in the Reuleaux triangle oriented toward the destination, even though a neighbor with forward progress exists outside of this area. Up to three broadcast transmissions of the same packet may be necessary to guarantee that a suitable neighbor is found if one exists. Figure 3.23 shows the probability

distribution for the number of broadcasts required to find a neighbor with forward progress. Again, it is possible that no neighbor with forward progress exists. From Figure 3.23 we observe that for any significant number of neighbors, it is highly likely that a node is located within the Reuleaux triangle. This corresponds to the conclusions made in Section 3.3.2 concerning Figure 3.19. In particular, the best nodes are likely to be located within the Reuleaux triangle.

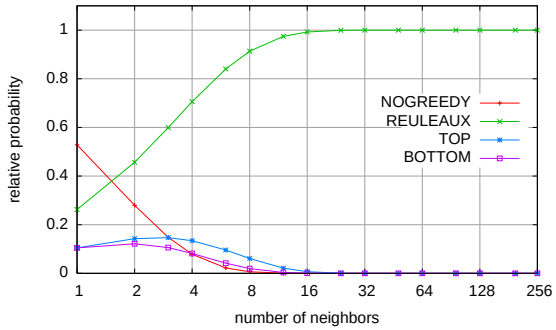


Figure 3.23: Relative probabilities of "First Next Hop is in Region"

The area-based suppression scheme has the same characteristics as the basic suppression scheme when a forwarding node can be found in the first Reuleaux triangle. Otherwise, the forwarding node has to wait for T, and then has to rebroadcast the packet in the second, and possibly even in the third area. The probability of no next hop in the Reuleaux triangle is very small for a reasonable number of neighbors (six or more). Hence, the difference in forwarding delay between the basic and the area-based suppression schemes is significant only for a small number of neighbors within transmission range.

The forwarding delay introduced by active selection depends not only on the time required to transmit a data packet, but also on the time needed to transmit the `RTF` and `CTF`. Both packets are likely to be small, and the time to transmit them should be significantly shorter than that for data packet transmission. If the forwarder waits for the feedback delay T (i.e., until all possible `CTF`s have arrived) and then forwards the packet to the best suitable node, we have a constant forwarding delay of T, as shown in Figure 3.22. With an integration of the `MAC` layer and `CBF`, the forwarding delay can be improved by giving a higher priority to data packets which suppress subsequent `CTF` packets once the first `CTF` has been received by the forwarder.

General Remarks

To conclude, even though the basic suppression scheme is the fastest and does not incur any additional overhead in terms of additional messages or retries until a next hop is found, its applicability is limited. Even under favorable conditions, packet duplication occurs with a likelihood of more than 50% at each hop. Therefore, more sophisticated suppression schemes are desirable.

The area-based suppression scheme is well suited if the density of nodes is sufficiently high. Only for very small numbers of neighbors are the good suppression characteristics offset by a greater forwarding delay.

Active selection can be used with all node densities and suppression delay values. There will be no uncontrolled duplication of packets. Its main drawback is that it transmits at least two additional packets (`RTF`/`CTF`) for each forwarding of the data packet. In scenarios where the density of nodes is high and the suppression delay is comparatively low, the area-based suppression scheme may be preferable.

3.3.4 Protocol Simulations

Simulation Setup

The proposed mechanisms were implemented for the ns-2 network simulator [15] version 2.1b8a (using the `MAC` layer of the version 2.1b9 with additional bug fixes). The size of the simulated area is 2 km × 2 km. We simulate different node densities and different levels of mobility using the *Random Waypoint Model* [165].[20] The different levels of mobility are achieved by modifying the maximum node speed, with a movement pause time of zero. For every combination of protocol variant, node density and maximum speed, we generate 50 independent sets of movement scenarios. For each of these scenarios, we randomly pick one sender-receiver pair. The sender transmits 100 packets with a payload of 128 bytes at a constant rate of four packets per second. Each simulation lasts for 40 seconds of simulation time. Data traffic starts at 5-10 seconds (randomized) after the start of the simulation, giving the beacon-based protocols time to exchange neighbor information and leaving enough time at the end to deliver remaining packets before the simulation terminates.

The simulated protocols are the three `CBF` schemes as described in Section 3.3.2 and a basic greedy forwarding mechanism based on `GPSR` [171]. The protocols are simulated without the perimeter mode (i.e., without the repair strategy if greedy forwarding fails to find a route to the destination; in that case, packets are simply dropped). Greedy forwarding using beacons is simulated with and without the

[20] Note that with the *Random Waypoint Model*, the node density is not uniform [72]. The higher the node mobility, the earlier will the originally uniformly distributed nodes accumulate in the middle of the simulation area, decreasing the average communication distance. Nevertheless, we choose the model to allow comparison of our simulation results with other simulation studies.

ability to re-route packets if a selected next hop is not reachable by means of the link layer, the so-called MAC callback option. The two alternatives are called 'optimized greedy' and 'basic greedy' in the discussion of the simulations. The simulated beacon intervals are 0.5, 1.0, and 2.0 seconds, and both greedy schemes used implicit beaconing, whereby beacons are also piggybacked on data packets.

An 'optimization' used for CBF is the introduction of an duplication suppression scheme that works as follows: Every packet is marked with a packet ID by the original sender. If a node has already forwarded a packet with this ID or was suppressed during the contention, it will not attempt to forward the packet again.

The underlying MAC protocol is IEEE 802.11 with a capacity of 1 $\frac{MBit}{s}$ to ensure that the broadcasts for CBF (as well as the beacons) and the unicast packets for greedy routing are transmitted at the same bitrate.[21] For the simulations, an unmodified MAC 802.11 was used, but we note that by integrating CBF and MAC, the performance of CBF can be improved considerably.

In the following sections we investigate the performance of the different routing algorithms with particular focus on the impact of node mobility. A more extensive simulation study of CBF can be found in [192′].

Node Density

As a first sanity check, we simulated CBF and greedy forwarding without node mobility for different node densities. Without mobility, the beacon interval has no impact on the performance of greedy routing, and location information is always accurate.

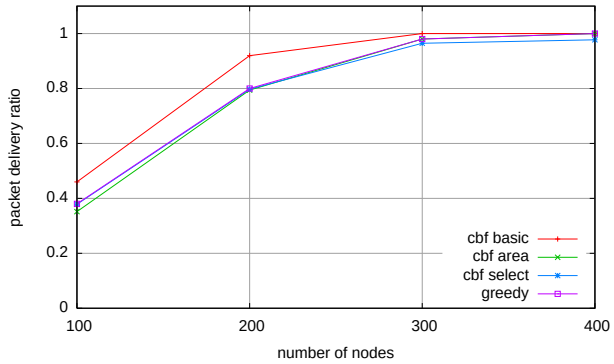


Figure 3.24: Packet delivery ratio for different node densities

[21] Earlier versions of the ns-2 MAC had a bug, using a higher rate for broadcasts than the standard allows. This bug is fixed in the code we used.

An immediate result of this simulation is that the runs with 100 and 200 nodes result in high packet loss rates for all approaches. This is caused by the fact that they frequently reach a local optimum and thus fail with low node densities.

In more detail, it can be observed that the basic `CBF` scheme achieves a higher packet delivery ratio than all other schemes as shown in Figure 3.24. Due to packet duplication, packets may be forwarded along a non-greedy path and find a route to the destination, even if no greedy route exists. As expected, the other `CBF` schemes, as well as greedy forwarding, have very similar packet delivery ratios, which depend mostly on the probability that a greedy route exists, given the current node density. The area-based scheme has a slightly lower packet delivery ratio for very low node densities, as the sequence of probing areas may result in the choice of a forwarding node that makes less progress than the best node of all forwarding areas. For higher node densities where the forwarder is almost always in the first forwarding area, this discrepancy vanishes. Active selection performs slightly worse than the other schemes for higher node densities since the request-response procedure increases the likelihood that a packet collision will occur during the forwarding process. Its performance could easily be improved by allowing packet retransmissions.

The analysis of other performance measures (e.g., routing overhead and forwarding delay) is of little value if only a small fraction of the sent packets arrive at the destination. For this reason, we limit the remainder of our analysis to simulations with 300 nodes.

Packet Delivery Ratio

Figure 3.25 shows the packet delivery ratio of the three `CBF` schemes: the basic greedy scheme for all three simulated beacon intervals, and the optimized greedy scheme for a beacon interval of one second. The values for optimized greedy with other beacon intervals were omitted because their performance in the chosen scenarios is similar to that in the run with a beacon interval of one second. The node density is 300 nodes in the simulated area of 4 km^2. The x-axis shows the four different groups of movement scenarios with their respective maximum node speed.

As can be seen from the graph, all `CBF` schemes and the optimized greedy scheme reach very high packet delivery ratios. Since the node density is fairly high, greedy routes exist most of the time. Only the packet delivery ratio of the active selection scheme suffers slightly when mobility is high. In such scenarios, it is possible that a node will have moved out of transmission range before sending the `CTF` (which nevertheless may suppress the `CTF`s of other nodes) or before receiving the actual data packet. Currently, the active selection scheme uses no recovery strategy that attempts to retransmit a packet if no `CTF` is heard after the timeout interval T, and the packet is lost.

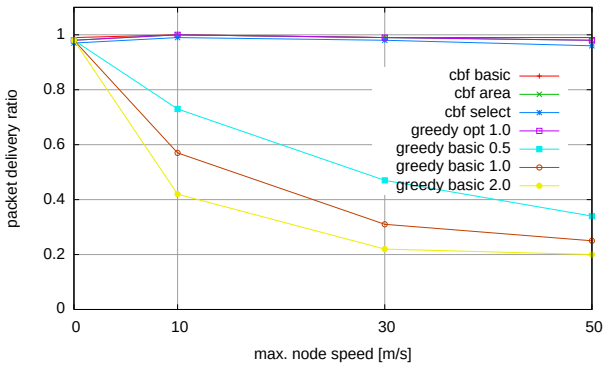


Figure 3.25: Packet delivery ratio for scenarios with 300 nodes

In contrast to the CBF schemes and to the optimized greedy approach, the basic greedy scheme performs significantly worse under mobility. At a maximum node speed of 50 $\frac{m}{s}$, the packet delivery ratio drops to 0.2, with a beacon interval of 2 seconds. 'Basic greedy' selects a greedy forwarder out of the list of neighbors and tries to transmit the packet to it. If a neighbor moves out of transmission range, its entry expires, and it is removed from the neighbor table after a timeout period during which no packets are received.[22] During this period, all packets handed down to the link layer with this node as the next hop are lost. The optimized greedy scheme detects these failures and reroutes all packets in the MAC queue destined for this next hop. Consequently, no packets are lost when the best suitable neighbor leaves the radio range if there is another suitable next hop in the neighbor table. The higher the node mobility, the more the packets exist that cannot by delivered by the basic greedy scheme and are therefore re-queued by the optimized scheme. Hence, the good performance of the optimized scheme comes at the expense of a trial-and-error strategy to detect a suitable forwarder that is still in transmission range, which may significantly increase the per-hop delay (see also Section 3.3.4) and the network load. The CBF schemes achieve similar packet delivery ratios without any link-layer packet loss recovery for the packet transmissions.

The same scenarios have also been simulated for densities of 100, 200, and 400 nodes within the 4 km^2 simulation area (not shown here). Generally, low node densities with only 100 or 200 nodes reduce the likelihood of greedy routes to the destination, and all schemes achieve lower packet delivery ratios. With 400 nodes, the optimized greedy scheme, the basic CBF scheme, and the area-based CBF scheme deliver 100% of the packets. Active selection achieves a delivery ratio slightly below

[22] This beacon expiry timeout is usually a multiple of the beacon interval. We chose it as 3.5 times the beacon interval as in the simulations in [171].

100% in high-mobility scenarios for the reasons explained above. The performance of the basic greedy schemes improves only marginally.

Transmission Costs

In Figure 3.26, we show the transmission costs for the optimized greedy schemes and the CBF mechanisms in terms of the average number of bytes transmitted at the MAC layer over the course of the simulation. The basic greedy schemes were omitted for lack of comparability; at high mobility, the packet delivery ratio is too low to allow a meaningful interpretation of the total overhead.

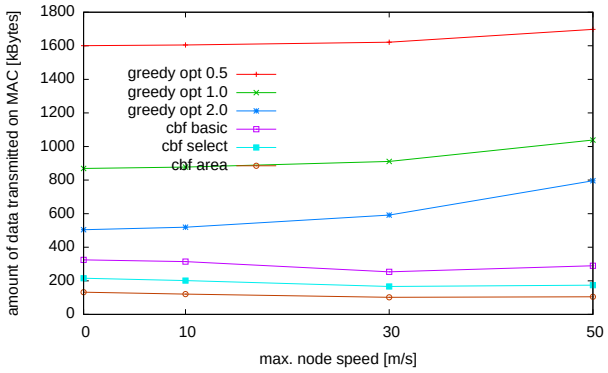


Figure 3.26: Transmission Costs on the MAC layer for 300 nodes

As expected, all CBF methods use less bandwidth than the greedy schemes that include the overhead caused by the beacon messages.[23] The area-based scheme consumes the least bandwidth, as no packet duplication occurs and – given a node density of approximately 15 nodes within transmission range – the forwarding node is almost always located within the Reuleaux triangle. Active selection causes a slightly higher overhead through the additional RTF and CTF messages, and the basic CBF schemes causes the highest transmission costs of all the CBF schemes, due to packet duplication. The bandwidth consumption by all CBF schemes is relatively independent of mobility. The slight decrease in overhead can be attributed mainly to the decrease in the average path length caused by the *Random Waypoint Model*.

The overhead caused by optimized greedy routing depends on a number of factors. The amount of data transmitted for beacon messages scales proportionally to the number of nodes, the beacon interval and the simulation time. The value decreases somewhat with an increase in traffic since implicit beaconing causes beacons to be

[23] Results are significantly worse for the greedy schemes when we investigate the number of packets instead of the number of bytes, since beacon messages are generally much smaller than data packets.

piggybacked onto the data packets. Furthermore, the transmission costs for the greedy scheme increases significantly with an increase in mobility. The better the available neighbor information is due to a high beacon rate, the lower the increase in the MAC overhead caused by increasing mobility. When mobility is high, a large fraction of the packets have to be sent multiple times because of the MAC callback. This ratio decreases when more accurate neighbor information is available, at the expense of an increase in the overhead caused by the beacons.

To analyze the transmission costs caused by the optimized greedy scheme in more detail, Figure 3.27 shows the specific components of MAC traffic for a beacon interval of 2 seconds and the scenarios with 300 nodes.

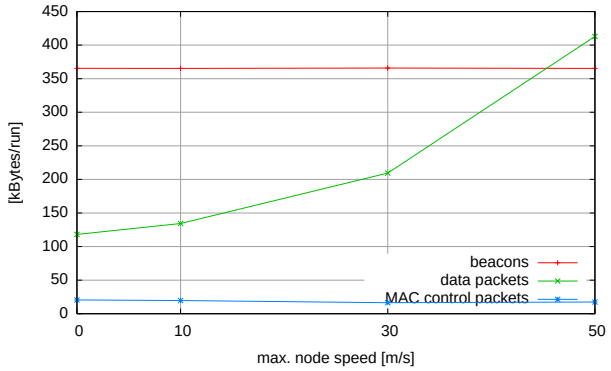


Figure 3.27: Cost composition of greedy opt 2.0

The bandwidth consumed by beacon messages and MAC control packets (i.e., unicast acknowledgments of the data packets) is independent of the mobility rate. In contrast, the overhead caused by the transmission of data packets increases significantly with higher mobility. Without mobility, optimized greedy consumes about as much bandwidth as does area-based CBF. For a maximum node speed of 30 $\frac{m}{s}$, optimized greedy already consumes the same bandwidth as the active selection scheme (where the additional RTF/CTF messages in the active selection scheme also provide protection against the hidden terminal problem). For node speeds of 50 $\frac{m}{s}$ and above, the greedy scheme even significantly exceeds the bandwidth usage of the basic CBF scheme with its unsuppressed duplicates. At this node mobility, the forwarding overhead is higher than the overhead caused by the beacon messages of all 300 nodes and exceeds the forwarding overhead with no mobility almost by a factor of four.

With only one sender and receiver and a data rate of 4 $\frac{kBit}{s}$, the amount of data traffic is extremely low given the total number of nodes. At such low rates, the additional traffic caused by the optimized greedy scheme can be handled by the MAC layer without any problems. However, for reasonable combinations of beacon

traffic and actual data traffic, we expect the overhead ratio to become much worse. When the additional traffic caused by repeated MAC callback results in congestion, data packets as well as beacon messages may be lost. The former have to be retransmitted at the cost of additional bandwidth consumption, while loss of the latter decreases the accuracy of the neighbor tables, further aggravating the MAC callback problem.

Forwarding Delay

For all CBF simulations, the maximum response time T was set to 45 ms. Although this parameter has a large impact on the average latency, it was not subject to optimization because (a), it would have multiplied our simulation time and (b) it is very scenario dependent. The optimal setting of T depends to a large degree on the MAC protocol and can be significantly reduced by integrating MAC and CBF. The parameter should further be dynamically adjusted to the node density and to the network load. An optimized maximum response time adjustment strategy is left for future work.

Nevertheless, an analysis of packet forwarding latencies confirms the observations regarding the protocol overhead. Figure 3.28 shows the average per-hop latency (i.e., the time required by a packet to travel from source to destination divided by the average number of hops along the route). Comparing the CBF schemes, the basic scheme has the lowest latency. There is no RTF/CTF handshake as there is with active selection and no sequential querying of regions, as in the area-based scheme. This also explains the delay characteristics of the other two CBF schemes: with zero mobility, the select scheme performs slightly better, because sometimes better routes may be found than by the ordered querying of areas. In a static scenario, this affects all packets, causing a perceptible difference in latencies. Mobility alleviates this effect, and area-based CBF achieves slightly lower latency values.

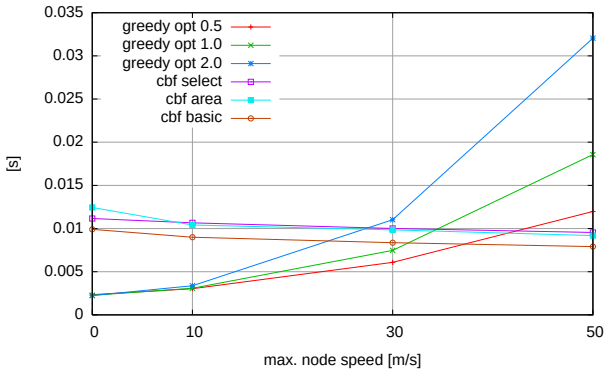


Figure 3.28: Average hop latency for 300 nodes

When comparing CBF with the optimized greedy strategy, one can observe behavior similar to that for the transmission overhead. While mobility even *reduces* the forwarding delay of the CBF schemes, the delay of optimized greedy increases drastically with higher mobility. For maximum mobility rates of more than 30 $\frac{m}{s}$, the forwarding delay is longer than that of the unoptimized CBF schemes. The responsibility for this effect lies again in the increasing number of link layer retransmissions.

3.4 Contention-Based Distance-Vector Routing

Position-based routing protocols forward packets in a greedy manner from source to destination without having to maintain routes through the network. Contention-based routing strategies improve upon position-based routing in that they do not even require the maintenance of neighbor tables at the nodes. This makes them very robust in highly mobile networks. However, neighbor tables are essential to the recovery mechanisms that are used when greedy routing fails. In this section we outline "Contention-Based Distance-Vector Routing", a recovery strategy for contention-based routing protocols that works when no neighbor tables are present. We describe the basic idea and give a simulative analysis of its performance.

3.4.1 Introduction

The basic forwarding mode of both the beacon- and the contention-based routing mechanisms minimizes the remaining distance to the destination in a greedy fashion. While this heuristic allows one to base the forwarding decision on local information only, there are cases where it will not reach the destination even though a valid route exists. [85, 171] propose recovery strategies based on the distributed planarization of the neighborhood graph that achieve theoretical completeness (i.e., they always find a route if one exists). The planarization requires neighbor table information, which is not present in CBF-like protocols. Therefore, such recovery mechanisms are not applicable in the context of CBF.

In this section, we propose a recovery scheme for CBF-like protocols. It is based on distance-vector routing (see Section 2.4.1) that is specifically adapted to contention-based operation. It allows recovery from local optima of the greedy forwarding while still maintaining the desirable property of relatively low resource consumption.

The remainder of this section is organized as follows: The next section gives a short overview of the related work. In Section 3.4.3, the protocol will be described,

and in Section 3.4.4 we will present some results from our simulations. Concluding thoughts and a summary are given in Section 3.6 at the end of this Chapter.

3.4.2 Recapitulation of some Basics

The approaches for routing in MANETs can be divided in table-driven (topology-based) and position-based algorithms. Position-based algorithms require that each node knows its own position and the position of the destination. The latter information is obtained by using a *location service*. In the following we will assume that such a location service is available (see Section 2.5.6). Most of the position-based algorithms are based on the idea of *greedy routing*.

Greedy Routing

The main idea behind the greedy routing approach is to find a global maximum through a sequence of locally optimal decisions. Applied to routing algorithms this means that each node selects its best suitable neighbor to whom to forward a packet to [117]. In most cases, this will be the neighbor located closest to the destination is considered the most suitable. While in protocols known before CBF, the neighbors were selected explicitly out of a beacon-generated neighbor table, the contention-based approach does without those.

Especially for dense networks with high mobility, this approach was shown to dominate the beacon-based protocols. It uses timer-based contention to determine the next forwarder. Each data packet is broadcast rather than being sent to a pre-selected neighbor. Every neighbor overhearing this transmission sets a timer according to its distance from the destination. The timer of the node the least distance away expires first. This node broadcasts the packet again and suppresses at the same time all further transmissions of the same packet in its neighborhood. With this suppression scheme (called *basic suppression*), packet duplication is possible, since there may be nodes whose timer is set for the current packet but which are not in the transmission range of the first forwarder. To avoid this, two further suppression schemes are proposed: *area-based suppression* and *active selection*. In the area-based suppression scheme, only those nodes that are in the transmission range of each other are allowed to take part in the contention. If active selection is used, control packets are broadcast prior to the transmission of the actual data packet to determine the best suitable node.

Voids

Like all greedy algorithms, greedy routing fails once a *local maximum* is reached. This is the case, when a node has no neighbor closer to the destination than itself, although a path to the destination exists. The region in which no suitable node is

located is called *void*. For a complete algorithm, it is therefore necessary to design a *recovery strategy* for escaping from those voids. The first node at which greedy forwarding fails is denoted as *void node* (see Figure 3.29).

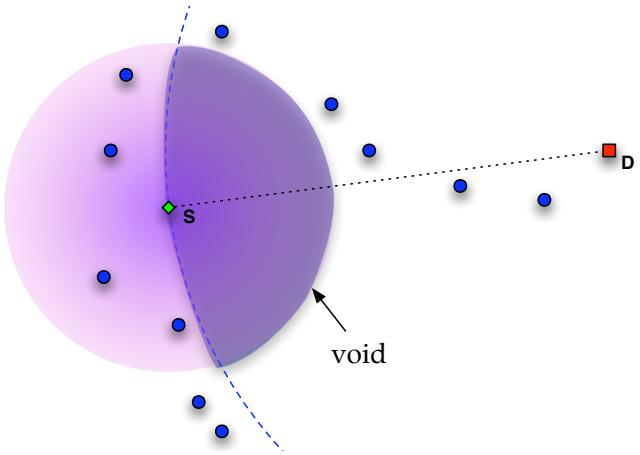


Figure 3.29: Greedy void: S is the void node, D the destination (just as in Figure 2.20)

In [171, 85, 189] it is proposed to escape from the void by using the right-hand rule on a planar graph which is created in a distributed way from the neighborhood information. Apart from the problem that this approach may lead to network disconnection and loops [210⋆, 170], it is not applicable to beaconless-algorithms such as `CBF`.

Another approach proposed in [206] is to reduce the probability of voids by allowing nodes to increase the distance to the destination to forward the packet if no better-suited node is available. Although the delivery rate can be increased considerably in this way, the algorithm remains incomplete, so that a recovery strategy is nevertheless necessary. The investigations in [200′] indicate, that the combination of this approach with a complete recovery strategy leads to a higher resource usage.

DeCouto et al. propose in [103] to use intermediate nodes as anchors along the route of the packet to reduce the problem of voids. However, this approach also is a heuristic and cannot guarantee delivery, not even in the case of a static network.

Table-Driven Algorithms

Apart from the greedy-based algorithms, there are many proposals for table-driven protocols. They have in common that they perform a route discovery by flooding the network with route request packets. These are answered by the destination, or

by nodes that know a route to the destination. In DSR [165], each node maintains a route cache, whereas AODV [241] stores *distance vectors* at intermediate nodes. Route failures due to node mobility are propagated using route failure packets.

3.4.3 Contention-Based Distance-Vector Routing

Overview

In this section, we present a recovery strategy to escape from local maxima based on contention forwarding that works without beacons. It is called Contention-Based Distance-Vector Routing (CBDV). In CBDV, a void node sends a request packet to find a node that can perform greedy forwarding again. We call such a node (i.e. a node that is closer to the destination than the void node) the *next greedy node*. It sends a reply packet back to the void node (*route discovery*). The actual data packet is then forwarded roughly along the established path (*recovery forwarding*). To reduce the number of control packets, the paths are stored for a specified time in the routing tables of the intermediate nodes.

Concepts

Contention Based Forwarding CBDV is an extension of CBF. Whenever it is possible, CBF is used to forward the packet. Only if greedy forwarding fails, is CBDV performed until a node is found that can forward the packet in a greedy manner again. Even while the packet is forwarded along an established path, it is always checked whether or not the current node is closer to the destination than the last void. Therefore, the positive properties of greedy routing are kept whenever the network topology allows it.

Adaptive Flooding CBDV uses adaptive flooding to find an escape route from a void. At first, the request packet is flooded with a range of n hops. Only if no route out of the void is found within n hops is flooding performed with a scope of $2n$ hops. Each request packet carries a Time-To-Live counter (TTL) and is dropped if the TTL is 0.

If the sender and the destination are not connected (due to the topology of the network), the packet cannot be delivered. To detect such a situation, a maximal number of hops (MAX_TTL) must be chosen for the flooding phase. If MAX_TTL is reached, the two nodes are considered to be unconnected, and the void node drops the data packet.

Contention-Based Distance-Vector Routing Like in AODV, distance vectors are used to store the established paths. However, in contrast to AODV, we resort to contention to choose the next hop in the recovery mechanism. There are two

Field	Size
Inter_Node.Address	4
Inter_Node.Location	6
Last_Void.Address	4
Last_Void.Location	6
Seqno	4
Hop Count	1
Mode	1
GREEDY/RECOVERY	

Table 3.4: Additional header fields for `CBDV` data packets

reasons for this: First, we do not require a working unicast connection between two neighbors if the next hop is selected by means of contention. Second, the established paths are more stable because the movement or break-down of one node along the path does not necessarily disrupt the path.

Sequence Numbers If the mobility rate is high it is possible that nodes with outdated routing information stored in their routing tables erroneously take part in the contention. To prevent this, each request packet is associated with a unique sequence number that identifies one route discovery process. A node may only take part in the contention during the recovery phase if its table entry has a sequence number higher or equal to that of the packet to be forwarded.

Algorithm Details

The complete algorithm can be divided into three parts: *greedy forwarding*, in which `CBF` is performed, *route discovery*, and *recovery forwarding*. These parts will now be explained in more detail. In the greedy part of the algorithm there are only small differences from the original `CBF` protocol, the most important ones being the notification packets that have to be broadcast, if a node drops a packet due to packet duplication (see Section 3.4.3). Apart from this, there are small differences in the data packet header (see Table 3.4). Every node maintains two tables: a *routing table*, in which routing information learned in the last route discoveries is stored, and a *next greedy table*, in which a void node stores the information about the previously found next greedy nodes. The routing table contains the following fields:

- *destination address*
- *destination location*

- *number of hops*
- *sequence number*
- *expiration timer.*

The next greedy table has the following entries:

- *destination address*
- *next greedy node address*
- *next greedy location*
- *hop count*
- *expiration timer.*

When a node becomes a void node, it first checks whether or not it has in its *next greedy table* an entry for the destination. If this is the case, the packet is switched to RECOVERY MODE, and the address and location of the next greedy node found in the table are included in the data packet. From here on, the packet will be forwarded according to the recovery forwarding procedure (see Section 3.4.3). If no route to the destination is found in the next greedy table, the route discovery phase begins. This phase is also divided into two parts: a *route request* and a *route reply.*

Route Discovery Phase

Route Request In the request phase, the void node broadcasts a request packet with the initial `TTL` of $n = 2$. For the proper performance of the algorithm, it is very important to find a suitable value for n. Therefore we ran simulations with a static network model to find out the probability distribution of the void sizes (see Figure 3.30.) We observed that the probability for large voids is highest in networks with a critical node density. In our simulations with a network size of 2km $\times$2km this node density is reached at 100 nodes. In networks significantly above or below this critical density, the voids are smaller since greedy forwarding is more likely to succeed between nodes that are transitively connected. At lower densities this is due to shorter routes and at higher densities it es due to the fact that voids are more likely to filled with nodes, enabling greedy forwarding. But even in networks with the critical node density voids are mostly only of size 2. We therefore decided to choose a `TTL` of 2 for the first request packet sent during one recovery procedure. In doing so it is possible to overcome most of the voids, transmitting only a small number of control packets.

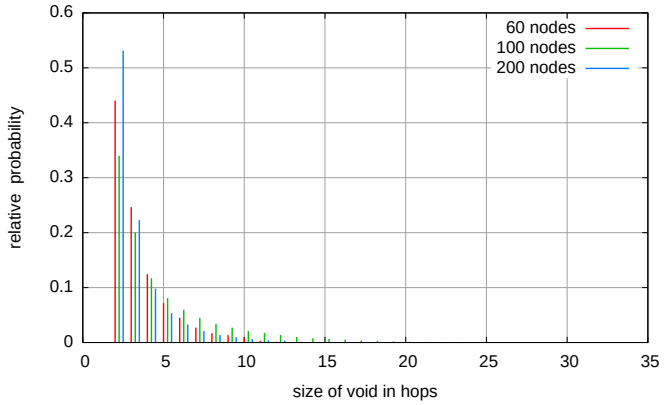


Figure 3.30: Probability distribution of the void sizes for different node densities. Network size: 2km ×2km

The fields of the request packet are described in Table 3.5. The request packet is broadcast by every node that has never forwarded this packet before.[24] At every hop, the `TTL` is decreased by 1. Each node receiving the request computes whether its distance from the destination is less than the distance of the void node (whose location is included in the packet's header) to the destination. If this is not the case and if the `TTL` is not 0, the packet will be re-broadcast. To set up the reverse path for the reply packet, the source address of the request packet is stored together with its hop count in each intermediate node's routing table. Older table entries are updated. If a node is nearer to the destination than the last void, it creates a reply packet and sends it back to the void node (see 3.4.3).

If the void node does not receive a reply within a certain period of time, it doubles the `TTL` and starts a new request phase with a longer `TTL`. If the `TTL` is greater than the chosen *maximal TTL*, the packet is considered to be undeliverable and dropped.

Route Reply The reply packet is forwarded along the reverse path established by the request packet, also using contention. For the fields of a reply packet, please refer to Table 3.6. In contrast to the greedy part of the algorithm where the contention is based on the distance from the destination, we use the hop count of the request packet as the first selection criterion for the contention in the recovery phase. This leads to the following contention process: Every node intercepting the transmission of a reply packet checks whether or not there is an entry for the desired destination (i.e., the last void node). A node may take part in the contention if

[24] For this reason, a small table is maintained in which the `ID` of each request packet is stored for a short period.

Field	Size
Type	1
Broadcast_Id	4
Source.Address	4
Source.Location	6
Destination.Address	4
Destination.Location	6
Hop_Count	1
TTL	1
Seqno	4

Table 3.5: Request packets

Field	Size (Bytes)
Source.Address	4
Source.Location	6
Destination.Address	4
Destination.Location	6
Orig_Dest.Address	4
Hop_Count	1
Seqno	4
Last.Hop_Count	1

Table 3.6: Reply and route failure packets

three conditions are met: first, such an entry must exist, second, the sequence number of the routing table entry must be higher than or equal to the sequence number of the reply packet, and third, the hop count of the entry must be lower than the last hop count. The final condition precludes the creation of loops during the contention.

Every participating node sets its timer according to the hop count stored in the table entry and its distance from the destination of the reply packet. The timers with the lowest hop count will expire first. If two nodes have the same hop count, the node the least distance away wins the contention. The successful node broadcasts the reply packet, and a new round of contention starts. All three suppression schemes described for CBF are applicable for the contention in the recovery phase, but we will show in the simulation section that the basic suppression scheme is sufficient (see Section 3.4.4).

Note that it is not necessary to forward a reply packet exactly on the same path as that of the request packets. If one node along the path is shut down or has moved away, the packet may use alternative paths. Since the third condition guarantees that the hop count decreases at each hop, loops cannot be created.

To establish the forwarding route for the data packet, each node that has intercepted the transmission of a reply packet stores the corresponding routing information in its routing table, regardless of winning or losing the contention. Therefore, it is also possible for the data packet to use alternative paths.

Once the reply packet has arrived at the void node, an entry for the final destination is created in the void node's next-greedy table. The void node enters the address and the location of the chosen next greedy node into the data packet's *Inter_Node* field and sets the mode of the data packet to RECOVERY. After that, the data packet is forwarded according to the recovery forwarding. If a void node receives more than one reply packet, it chooses the path with the shortest length.

Recovery Forwarding A data packet in the RECOVERY mode is forwarded using the same contention as described for the reply packet. Each node winning the contention checks whether or not it is closer to the destination than the last void node before it broadcasts the packet again. If the successful node is closest the packet is switched to GREEDY MODE again, and the recovery procedure has found a way out of the void. After that, greedy forwarding is performed as described for the `CBF` protocol. Note that the node that changes the mode does not necessarily have to be the node declared as the *Inter_Node* in the packet header.

If node movement has disrupted the established path, the last forwarding node will not overhear any further transmission since no neighbor can take part in the contention. It therefore sends a *route failure notification* back to the last void. This packet has the same fields as the reply packet (see Table 3.6) and is forwarded in the same manner. To mark a packet as route failure, the hop count is set to INFINITE. Since the reverse path is not the stored by the intermediate nodes, this field is not necessary for the route failure packet. A void node receiving a route failure notification for a previously sent packet will perform a route discovery procedure again.

It is possible that the route failure notification cannot be delivered to the last void. This is the only situation in which the algorithm may fail to deliver a packet, even if the maximal `TTL` is chosen high enough. We decided not to introduce another control mechanism to inform the sender of the route failure notification about this loss because a 100% delivery can never be guaranteed in a mobile network anyway. All table-driven protocols suffer from this problem, but in our simulations, the loss of notification packets occurs very seldom.

Packet Duplication Like in CBF, packet duplication may also occur in the recovery part of CBDV. If the basic suppression scheme is used packets are duplicated if neighbors exist that take part in the contention but are out of the transmission range of the node winning the contention. These neighbors cannot be suppressed and will therefore forward the packet themselves. Area-based suppression avoids this situation, but duplicates are nevertheless possible if two nodes are located within the duplication area (see Section 3.3.2). To reduce the additional traffic due to duplicates, the duplicated packets are dropped if they reach a node that has forwarded the same packet before. Also, to prevent other nodes from considering this packet drop to be a route failure (which would prompt them to initiate a new route discovery round), a duplicate notification packet is broadcast.

3.4.4 Simulative Evaluation

Implementation Model

To get an understanding of the algorithm and the task it is facing, we have implemented a simple packet-based network simulator. Also, we could have done this on packet level with ns-2. However, this would have created the problem that the void situation could not have been analyzed with statistical significance. Thus, we have decided to study it with a simpler model, enabling us to repeat the experiments more often and leave packet-level simulations to protocol engineers. The following model was used:

Connectivity Model All nodes are located within a rectangle in $\mathbb{R}^2$. For each simulation run, a graph is created by choosing n random points within this rectangle which represent the nodes of the graph. The connectivity between two nodes is modeled according to the *Unit Disk Graph Model* [115].[25]

Network Model We assume that a bidirectional connection between two nodes that are in transmission range of each other is guaranteed. We did not model the MAC layer or any other layer of the network. Therefore, collisions on the MAC layer are not taken into account.

Time Model Assuming the node movement to be much slower than the delivery time of a packet, the network is considered temporarily static between two deliveries. Therefore, a discrete time model with two time steps was used. In the first time-step, a packet is sent between each node pair (n_i, n_j) according to the described algorithm. After that, a new location for each node is chosen randomly from a

[25] A unit disk graph is a graph, where an edge between two nodes n_1 and n_2 exists, if the Euclidean distance $|\overline{n_1 n_2}| \leq r$, where r is a predefined transmission range that is equal for all nodes.

circle around the node, with a radius defined as the parameter *speed*. A second packet is created with the same source and destination that is routed on the altered graph, using the routing information gained in the first time step.

Algorithms To compare the different suppression schemes in the greedy phase as well as in the recovery phase, we simulated different combinations of the algorithm variants. For the greedy phase we simulated - aside from basic suppression (`BAS`) - only the active selection (`ACT`) and area-based suppression (`AR`) schemes since the basic suppression scheme seemed to be not very promising in previous studies (see Section 3.3). However, in the recovery phase, we simulated all three suppression schemes. In addition, we implemented a so-called *unicast* (`UNI`) variant for the recovery procedure, in which no contention is performed. Instead, the next hop is chosen according to a node `ID` stored in the routing table, similar to the procedure of `AODV`. In the following graph legends, a pair of strategies always means a combination of strategies for the greedy and for the recovery part. I.e., `AR BAS` means area-based `CBF` together with basic-suppressed `CBDV`.

Also, for the sake of comparison, we simulated the original `CBF` algorithm with active selection and area-based suppression.

Simulation Setup

All simulations have been run with a network size of 2000 $m \times 2000$ m and a fixed radio range of 250 m For each combination of the parameters *number of nodes*, *maximal TTL* and *speed*, 100 independent graphs were chosen randomly. In each time step, one data packet was sent between each node pair in the graph. The data payload was set to 128 bytes. The values for the different parameters can be found in Table 3.7.

Number of Nodes	50, 100, 200, 300 nodes
Maximal `TTL`	2, 4, 8, 16, 32 hops
Speed	0, 10,30, 50 meters per time step

Table 3.7: Simulation parameters

Choosing a higher number of nodes than 300 is not useful for our purposes; in such dense networks, `CBDV` behaves very similarly to the original `CBF` algorithm. Preliminary simulations with a static simulator have shown that a maximal `TTL` of 32 is high enough to achieve a delivery ratio of nearly 100% in static networks of the chosen size; therefore higher values are not taken into account.

Metrics

The following metrics were used to evaluate our algorithm[26]:

We denote the ratio of connected node pairs among all node pairs in the graph as *connectivity ratio*. This value is a characteristic of a graph and not of a certain algorithm. It is calculated as

$$\text{connectivity ratio} = \frac{fw}{n(n-1)},$$

where n is the number of nodes in the graph and fw is the number of node pairs with an existing path between them. fw is computed by using the Floyd-Warshall algorithm [96].

One of the most important characteristics of an algorithm is its *delivery ratio*. This value means the ratio of sent data packets to successfully delivered data packets. Packets that could not be delivered because the sender and the destination are not connected should not influence the delivery ratio because such delivery failures do not represent a failure of the routing algorithm. Since we always investigate *all* node pairs, we define the delivery ratio as

$$\text{delivery ratio} = p/(fw_i + fw_{i+1}),$$

where p is the number of delivered data packets and fw_i the number of all connected node pairs in time step i.

To quantify the costs of an algorithm, different metrics can be used. In [133⋆], the costs are measured in terms of *overhead per payload byte*. This value is defined as the total amount of transmitted bytes per each byte of data sent from a source node. Although this values provides a good comparison between algorithms with similar delivery ratios, it does not allow the comparison of algorithms that differ much in the delivery ratio. The reason for that is that routes on which the delivery fails are shorter than successful routes in the average case. Hence, the overhead is lower for algorithms with a low delivery ratio. We therefore use a different metric to measure the protocol costs: the *bytes per delivered packets*. It is defined as:

$$\text{bytes per delivered packet} = \frac{b_{all}}{p},$$

where b_{all} is the total amount of bytes transmitted during one simulation and p the number of successfully delivered data packets. This value is a measure for the efficiency of the algorithm.

[26] At this point, we stick to these freshly-defined metrics because they (a) reflect the special nature of this simulation and (b) are — in our opinion — best suited to describe the properties.

It is very difficult to find a meaningful metric with which to measure the actual 'overhead' of an algorithm. In [97] it is proposed to define the overhead as the number of control bits per delivered data bit. The drawback of this definition is that a protocol may cause 'redundant' data traffic that has to be counted as overhead. Such data traffic occurs, for example, if duplicates are generated or if the path length is longer than the shortest path. We therefore define the *optimal cost* as the number of bytes that have to be transmitted if a data packet is delivered on the shortest path, considering an 'ideal' algorithm that has no need for packet headers and control packets. The optimal cost is thus defined as

$$\text{optimal cost} = l_{min} \cdot \text{payload},$$

where l_{min} is the average path length as calculated by the Floyd-Warshall algorithm.

To evaluate the stability of the routes under mobility, we define the *route failure* as the ratio of data packets dropped in the second time step due compared to the number of routes discovered in the first time step.

As an indicator for the quality of the routes, we also investigated the *average path length*. Note that only successful deliveries are taken into account for this metric.

Results

Delivery Ratio The most important factors for the delivery ratio are the chosen maximal `TTL` and the node density. Figure 3.31 shows the delivery ratio as a function of node density for `CBDV` with maximal `TTL` 32 and 4, and for the sake of comparison, the delivery ratio of the original `CBF` algorithm. As the choice of the suppression scheme is of minor importance for the delivery ratio, we only show the variant with area-based suppression in the greedy phase and basic suppression in the recovery phase. The figure shows that there is a critical node density at which the delivery ratio has a minimum. This is even true if the maximal `TTL` is set to 32, which means that the delivery ratio is about 99.9%. This behavior is similar to the behavior of the original `CBF` algorithm. In very sparse networks, the connectivity is low (see purple line in Figure 3.31), therefore the existing paths are short and easily found. In dense networks, each node has so many neighbors that greedy routing is almost always possible. Very large voids (where `CBDV` may fail if the chosen `TTL` is too short) are only possible in the critical node density. This density is therefore the most interesting one for our studies.

Comparing `CBDV` with the original `CBF` algorithm, we can state that `CBDV` increases the delivery ratio considerably, even if the maximal `TTL` is short. With a longer `TTL`, we can achieve a delivery ratio of nearly 100%.

Figure 3.32 shows the delivery ratio with respect to speed. It can be observed that the delivery ratio is only slightly reduced in networks with high mobility. The

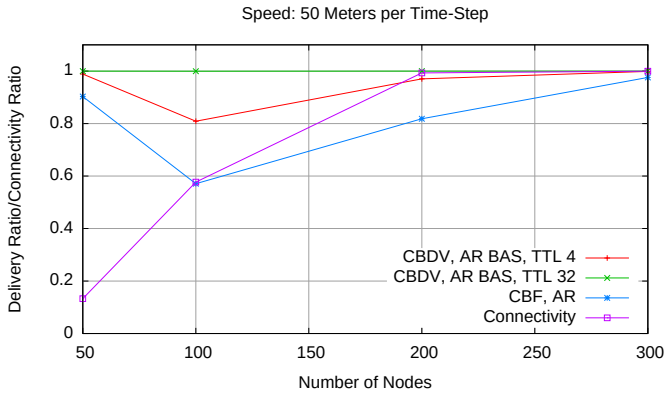


Figure 3.31: Absolute connectivity and packet delivery ratio of connected node pairs

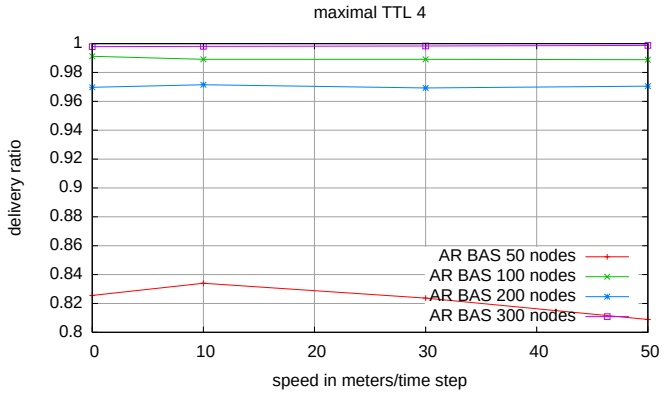


Figure 3.32: Delivery ratio with respect to speed

reason for that is that routes that are disrupted due to mobility are re-established by the last void. The packet is dropped only if the route failure notification packet gets lost.

Another interesting observation from this figure is that the influence of the mobility on the delivery ratio decreases in dense networks. In those networks, most paths are found by using pure greedy forwarding. Since greedy forwarding works completely in a stateless fashion, it is independent of the mobility rate.

Route Failure The investigation of the route failure allows us to compare the different suppression schemes in the recovery phase. Figure 3.33 shows this metric for all implemented suppression scheme combinations as a function of speed. It is

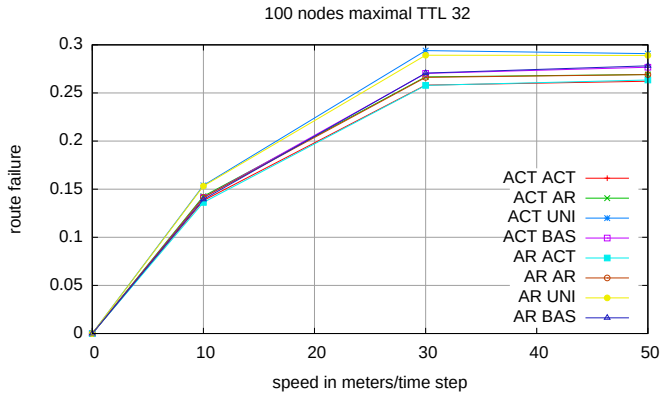


Figure 3.33: Route Failure with respect to Speed

obvious that the number of route failures increases at higher mobility rates, but this increase stagnates at a certain speed. This occurs because the probability of using a pre-established route decreases in networks with high mobility. In those networks, a data packet may never reach the recovery path discovered by the first data packet since the greedy forwarding has lead to a completely different route in the greedy phase. Most established routes are therefore never used. Consequently, the usage of routing tables can only add benefit to the algorithm if the mobility rate is moderate. This is a consequence of the tendency of greedy forwarding to react very fast to changes of the network topology. However, for lower mobility rates, using previously learned routes seems to be promising since most of the used routes are intact.

The comparison of the different suppression schemes shows that the unicast variant causes the highest number of route failures. The differences between the other suppression schemes are less distinct. This indicates that the introduction of contention in the recovery phase improves the stability of the routes.

Cost composition `CBDV` uses five types of control packets: request packets and reply packets for route discovery, duplicate notification packets, and (if active selection is used) `CTF` packets and `RTF` packets. Figure 3.34 depicts the cost composition for the variant using area-based suppression in the greedy phase and active selection in the recovery phase, as a function of node density. It can be observed that the `CTF` packets and `RTF` packets for the recovery phase, as well as the notification packets, and even the reply packets, account for only a small portion of the total costs. Moreover, the most important factors are request packets and data packets. Note that we always counted *all* packets, including those used for non-successful deliveries. This is the main reason for the higher amount of traffic per delivered

packet in sparse networks. At higher network densities, the request packet traffic decreases since the greedy success rate improves and the recovery procedure thus has to be performed more rarely.

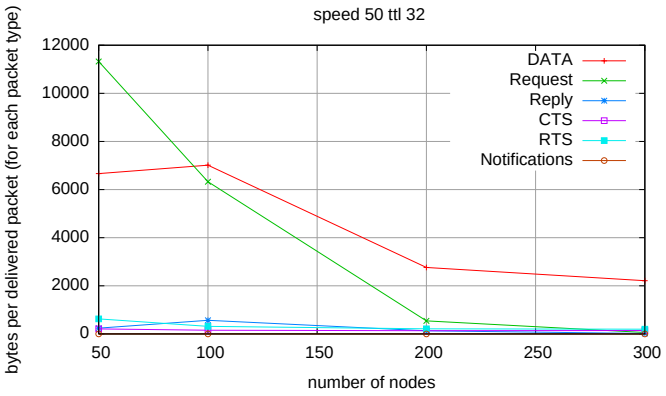


Figure 3.34: Cost composition with respect to node density

Request Packets We have seen in the last paragraph that the request packets constitute a considerable portion of the total amount of bytes. We therefore show the total number of request packets transmitted during one simulation in Figure 3.35 with respect to speed. It is obvious that the number of request packets increases

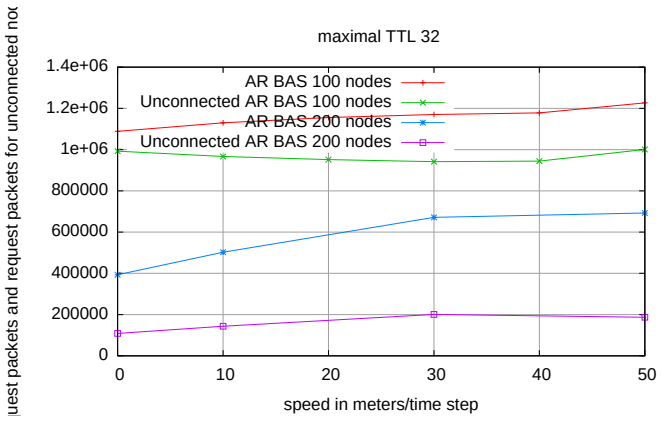


Figure 3.35: Total number of request packets, and number of request packets transmitted before network disruption is detected

at higher speed since the higher mobility causes more route disruptions, so that

the recovery procedure has to be performed again. However, we observed that this increase is very limited, especially for networks within the critical node density. The reason for this lies in the request packets that are necessary to detect that two nodes are unconnected. The number of those packets depends on the connectivity of the network, which is independent of the mobility rate. As we can see from Figure 3.35 (the solid line), the number of those packets is very high. Due to their independence from mobility, the overall number of request packets is influenced very little by the mobility rate. Note that we depicted the results of the simulations with a maximal `TTL` of 32, where this effect is highest. With shorter `TTL`s, we have a greater dependence on the mobility, but the number of request packets is also smaller (since greedy forwarding succeeds more often), so the affect on the overall overhead is thus rather small.

Number of Bytes per Delivered Packet To evaluate the overall overhead of the algorithm, we examined the number of bytes per delivered packet and compared it to the optimal costs. Figure 3.36 shows this metric as a function of speed. Again,

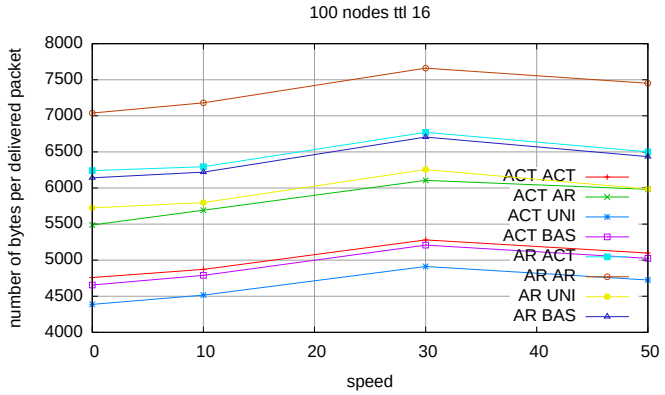


Figure 3.36: Bytes per delivered packets with respect to speed

we can state a rather low influence of the mobility on the number of bytes. This is especially true for dense networks. We have already shown that the number of request packets is affected very little by the node speed. The same is true for the number of data packets since data packets are usually transmitted during the greedy phase in which the mobility has no influence on the performance. This also explains why the overhead for denser networks is lower (compare to Figure 3.37). The low impact of mobility on the overhead indicates the good suitability of the algorithm for highly dynamic networks.

Moreover, Figure 3.36 shows us the different suppression schemes in the recovery phase. It can be seen that the area-based suppression scheme generates the most

traffic. This results from the fact that we have to send up to three data packets in the greedy phase, and even up to six data packets to detect that a node has no suitable neighbor to whom to forward the packet. Note that we depicted the results for a network with the critical density of 100 nodes. In denser networks, the probability of having a forwarding neighbor becomes higher. The area-based suppression scheme is therefore more suitable for dense networks. On the contrary, the active selection scheme works best in sparse networks, since a greater number of neighbors also generates a greater number of RTF and CTF packets, whether or not a forwarder exists. However, these characteristics are of minor importance for the recovery part of the algorithm because the basic suppression scheme is sufficient for this part. We discovered that the probability of generating duplicates in the recovery phase is very low, even in dense networks. Therefore, the basic scheme has no drawbacks in comparison to the other schemes. In addition, it has the advantage of a lower delay. We thus consider this scheme the best suitable for CBDV.

We also depicted the results for the unicast variant which produced the lowest overhead. However, as we already mentioned, the model of this method is not very realistic. Furthermore, the main reason for the lower overhead is the smaller packet header.

The most important characteristic of CBDV can be seen in Figure 3.37. The

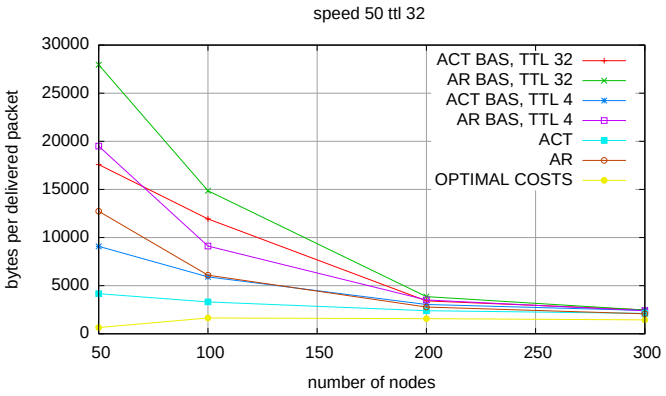


Figure 3.37: Bytes per delivered packets with respect to node density

overhead of CBDV approximates the overhead of the original CBF algorithm and consequently, apart from a constant factor, the optimal cost. This is the consequence of the characteristic of CBDV to use CBF whenever it is possible. Thus, the good performance of CBF is only reduced in networks with the critical node density.

Path Length At last, we show the average path length of the paths found by CBDV. This metric can be considered as an indicator of the quality of the paths. As

can be seen from Figure 3.38, the path length is nearly optimal in very dense or in very sparse networks.

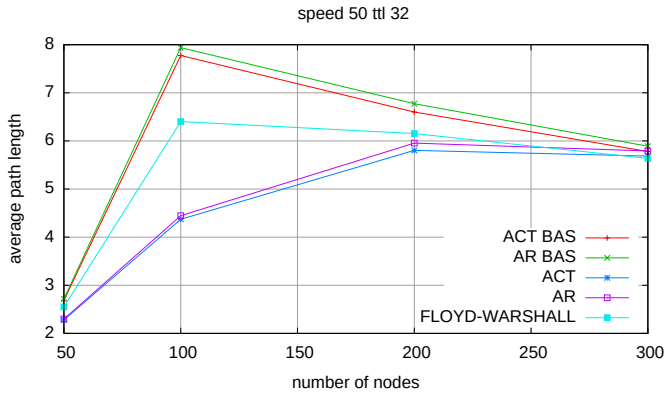


Figure 3.38: Average path length

The reason for that is the property of greedy forwarding to generate paths close to the shortest path in case of success. However, at the critical node density, the average path length for CBDV is about one hop longer than the optimal path length. Since the recovery procedure theoretically creates an optimal path between the void node and the next greedy node, this increase is mainly caused by the location of the void node. In general, a packet that has reached a void has taken a few hops in the 'wrong' direction. These additional hops affect the total path length. Note that only successful routes are taken into account for the average path length. Therefore, the path length is always shorter for the pure greedy algorithms, which are more likely to fail on longer routes.

3.5 Earlier/Parallel Work and Evolution of the Concept

Contention[27] as a fundamental concept has created a significant impact on the wireless networking community. While [134⋆], which was first submitted for publication in 2002, has been among the top ten of bought-via-download articles for Elsevier's *Ad-Hoc Networks* journal in 2003, a substantial amount of work published later makes use of contention, unfortunately often without making proper reference to our work.

However, to claim that the whole concept was invented with CBF would also be not true. Our own roots with feedback timers originate from wired multicast with Jörg

[27] Other researchers also refer to CBF-like concepts as being opportunistic because they use the best available forwarder for every transmission.

Widmer's Equation-Based Congestion Control [300, 298] and the corresponding work on the distribution of feedback timers [299, 122], again, founding on [232], [86] proposes a progress-jittered flooding scheme, and [79] sketches Geo-MAC, which is a merging of the medium access problem with geographic forwarding, as a research perspective.

Later published Proposals

GeRaF or Geographic Random Forwarding [309, 308] describes an algorithmic scheme similar to CBF Basic, but without packet-level protocol simulation.

BLR or Beacon-Less Routing [152]: An approach similar to area-based suppression, one of the three suppression schemes presented here, also giving some mathematical analysis, but also lacking packet-level simulations.

ExOr [77, 76, 78] describes Opportunistic Routing in the Context of M.I.T.'s RoofNet rooftop network [16]. This work is the first to introduce opportunistic routing to realistic radio conditions.

In addition to packet-level simulation, [196′] describes an implementation of CBF on sensors. Since our target platform has been *Vehicular Ad-Hoc Networks*, the next chapter will show the application of Contention-Based Methods in such VANETs.

3.6 Conclusions and Perspectives

The reactive location service proposed in this chapter is the piece necessary to study working on real-world-applicable position-based routing schemes that had been lacking until now. Previous work on forwarding simply assumed the availability of the destination's position and used simulator knowledge for packet-level simulation. In addition, the available location services were complex and as shown for the example of GLS choked the network at high mobility. RLS, being based on the well-known flooding scheme, is purely reactive and highly mobility-resilient. As a side result we show that location-based routing performs better in high-density / high-mobility conditions than DSR. Hence, we confirm the work in [171] but without the omniscient location knowledge the simulator can provide.

The advantage of position-based routing over other MANET routing protocols is the fact that nodes require only knowledge about the local neighborhood and the destination's location rather than a knowledge of a global route topology. Position-based routing is therefore better suited for networks above a certain degree of mobility. With the contention-based forwarding mechanism proposed here, even this local knowledge and, hence the sending of beacon messages is no longer required: any node with progress toward a destination can participate in the forwarding process

without the need to register this node in a neighbor table. For CBF, data packets are transmitted via single-hop broadcast. All nodes within radio range and with forward progress toward the destination are eligible to continue to forward the packet. Thus, the responsibility for the forwarding decision now lies with the set of possible next hops instead of with the forwarding node, as is the case in conventional forwarding methods. Forwarding takes place after a contention period during which one or more nodes are selected as next hops. Selection of more than one next hop causes unwanted packet duplication. However, we have presented different suppression strategies to avoid this.

For existing position-based forwarding schemes, node mobility results in frequent beacon messages to keep the neighbor tables reasonably up-to-date. Particularly for highly mobile networks, CBF can provide significant bandwidth savings through the elimination of beacon messages and the reduction of MAC-layer retries for packet transmissions caused by inaccurate neighbor tables. Furthermore, the decrease in the total number of packets reduces the probability of packet collisions, and inefficient routing caused by inaccurate neighbor tables is avoided.

The simulation results presented in this paper show that excessive re-transmission of data due to outdated neighbor table entries, as it is the case for traditional position-based routing, can be completely avoided by the proposed contention-based forwarding approach. Since CBF does not require any beaconing, and since CBF together with the area-based suppression strategy does not lead to any noticeable packet duplication, the resulting data volume overhead of the contention-based method is much less than the data volume overhead generated by traditional position-based routing in highly mobile *Ad-Hoc Networks*. Clearly, reducing load on the wireless medium is beneficial for *Mobile Ad-Hoc Networks* in general. In the rare case where a packet duplication occurs due to CBF, a simple strategy exists to improve the proposed suppression schemes: if duplication of packets occurs these packets will be routed to the same destination at roughly the same time. Investing a small state about which packets were recently forwarded, the duplicates can easily be suppressed in later nodes. Thus, packet duplication can be reduced, while the simplicity of the suppression schemes is retained. In addition to the reduced forwarding overhead, the CBF schemes also provide a lower packet-forwarding delay when node mobility is high. In the simulations, we used very conservative timer settings, and we expect the reduction in forwarding delay to be much more pronounced for a well-tuned CBF implementation.

As a companion to CBF, we have proposed Contention-Based Distance-Vector Routing (or CBDV) as a recovery strategy for CBF-class protocols in case position-based greedy forwarding fails. Our protocol can be seamlessly integrated into CBF and approximates the advantages of CBF in dense networks. Furthermore, we showed that the introduction of contention to the recovery phase improves the

stability of the routes. As the main advantage we identified the good suitability of the protocol in highly dynamic networks.

In our experience, research on MANET protocols is never complete in the sense of a full enumeration of all possible scenarios. Thus, for every protocol we focus on items that are interesting from the scientific point of view. Every protocol needs to be thoroughly evaluated in context before finding its way into products. In the following, we will list some possible future research steps one might undertake with the presented protocols.

RLS perspectives

Possible future work could deepen the effects of RLS parameters on scenario-specific performance. E.g., it would be especially interesting to see how optimizations like re-broadcast suppression and cached replies could be used and combined to increase the application area of RLS. For scenarios in which the broadcast storm problem occurs, it would also be interesting to test the impact of different suppression mechanisms. Furthermore, a comparison of binary flooding to exponential flooding could reveal whether binary flooding is always better, or if scenarios exist in which exponential flooding (maybe combined with cached replies and/or re-broadcast suppression) proves more suitable.

CBF perspectives

One key item in CBF's future will be the further integration of CBF and MAC functionality. Since both serve a somewhat similar purpose, their integration can significantly reduce the overhead incurred by the CBF scheme. In particular, we expect that it is possible to significantly reduce the run-time of the random timers used in the contention process. If a MAC layer with RTS/CTS is used to solve the hidden terminal problem (as is possible with IEEE 802.11), it can be combined with the RTF/CTF messages of active selection, which will significantly increase the efficiency of this suppression strategy. Furthermore, a maximum response time T which adapts to network load and node density can reduce the delay incurred by the contention period. So far, we have only considered greedy forwarding. In position-based routing, greedy forwarding fails in the absence of a neighbor with progress toward the destination. In such a case, a recovery strategy is used to circumnavigate the area with no reception.

The use of directional antennas in *Ad-Hoc Networks* recently gained increased scientific interest [65]. This technology seems to be a promising candidate, particularly in the context of area-based suppression.

Also, research already started in [263*, 262*], evaluating CBF with a probabilistic radio model, has to be extended; by integrating the real nature of radio propagation, one could come up with promising new suppression strategies, e.g., Forward Error Correction (FEC) [37] could be used to equip parts of the packet with a higher range

than others. When the header is encoded in the former, the suppression range of a forwarded packet could be much higher than the forwarding range, which might very well be a very intelligent real-world alternative to area-based suppression.

CBDV perspectives

While our research of CBDV is mature, the most important next step would be to engineer the proposal into a real distributed protocol. This distributed protocol then would have to be evaluated by means of discrete event simulation (e.g, with the network simulator ns-2), which would also allow for a quantitative comparison with existing topology-based approaches, and position-based approaches using neighbor tables. Also, the simulation could narrow down the different protocol variants one could think of. One possible improvement is to adjust the maximal TTL dynamically to the node density, and to limit the number of route requests for the same destination within a certain time period. Since the request packets used to determine that two nodes are unconnected constitute a high portion of the total overhead, it could be advantageous to devise a more efficient way to acquire connectivity information (possibly over the location service). Furthermore, the question of unidirectional links has to be considered, which has proved to be a major drawback of a lot of existing approaches, but an inherent strength of CBF in position-greedy mode.

Chapter 4

Packet-Forwarding in Vehicular Ad-Hoc Networks

> Das Automobil ist so erfolgreich, daß es nur einen wirklichen Feind hat, nämlich sich selbst. Seine massenhafte Verbreitung ist eine Herausforderung an die Zukunft des Straßenverkehrs.
> (The automobile's success leaves none but one real enemy, which is itself. Its massive distribution is a challenge for the future of road traffic.)
>
> *(Eberhard von Kuenheim (former CEO of BMW))*

Chapter Outline

While it is good scientific practice to solve problems as abstractly as possible, the quest for generality in *Ad-Hoc Networks* also creates drawbacks (see Chapter 2), the most important one being that random node movement is not at all a generalization of reality. Moreover, random node movement is not even a special case similar to any realistic node behavior. Having discussed forwarding algorithms not applied to special scenarios in the last chapter, we will now move on to a special, more realistic scenario: the packet forwarding in a *Vehicular Ad-Hoc Network*, i.e., a network formed by vehicles traveling on roads.

In the course of this chapter, we will first introduce to the topic, and then separate the problem into forwarding on highways, and forwarding in cities, coherently arguing why this separation makes sense . Finally, we will give conclusions and perspectives.

As in the last chapter, we have already published most of the scientific content contained here. Readers interested in single concepts, are referred to read the papers, i.e., for forwarding on highways (Section 4.3) [125*, 126*, 222*, 124*, 175*, 127*, 185*, 128*, 176*, 263*], and for forwarding in cities (Section 4.4) [208$'$, 210*, 209*, 128*].

4.1 Introduction

Analyzing the requirements of a VANET communication system, it soon becomes clear that application requirements are spread significantly further than in traditional wired network systems as the Internet. E.g., time-critical communication to groups of nodes is very natural in these networks, as is addressing not based on IDs but on vehicle properties (see also Section 2.2.3). While these kinds of applications, often used in a vehicular safety context, are now believed to be the most interesting from an industrial point of view, our research mainly derived from using a vehicular ad-hoc communication system for the purpose of providing IP-style unicast datagram connectivity between vehicles that are not necessarily direct radio neighbors. Consequently, since link layer technology to enable communication between neighbors is available, the challenge lies on the next higher layer which is the network layer, essentially charged with solving the routing question.

Most of the research work during the preparation of this thesis has been dedicated to this special problem. In fact, the previous chapters are actually generalizations of results sought for VANET routing, which principally differs from general routing in the following major issues:

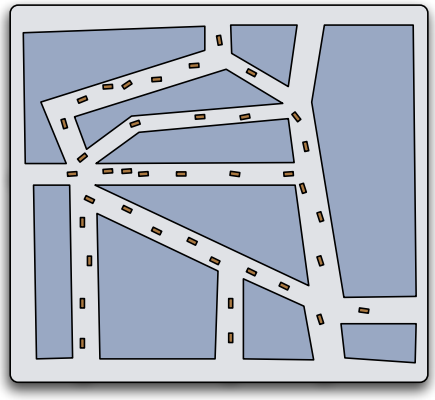
a) While the quest for energy preservation is an important matter in general MANETs, the energy consumed by the communication system is assumed to be insignificantly small for en-route vehicles. Thus, low energy consumption is not a major design goal.

b) In contrast to general MANET nodes, modern vehicles are equipped with all sorts of sensors busy collecting environmental data, the most important being accurate information about the vehicle's position. This information can be exploited for communication purposes.

c) Mobility significantly differs from purely random motion on a 2D plain, mainly because vehicles tend to move along roads, the speed of their movement being dictated by the road itself, by traffic, or by government regulations. While this at least seems to be exploitable for datagram routing, the mobility is also a problem because the sheer speed per radio range can be significantly higher than that which most general MANET algorithms have been developed for.

The last item presents the starting point of our research, when we were trying to find algorithms that reasonably solve the datagram routing problem for a rather simple vehicular highway scenario. In the process of this research, we found it helpful to divide VANET unicast routing into these highway scenarios, and into the more complex scenarios consisting of interconnected roads as in urban, or city, environments.

(a) Highway



(b) City

Figure 4.1: Vehicular Movement Scenarios

Consequently, the remainder of this chapter is structured as follows: First, we analyze the different mobility scenarios of significance to routing (Section 4.2). Then, we discuss research results we found addressing the packet routing problem in highway scenarios (Section 4.3) and afterwards the more complex city, or 2D scenarios (Section 4.4). Finally, we conclude this chapter by summarizing the contributions, and extending the results to other forwarding modes (Section 4.5).

4.2 Highways and Cities — Problem Separation

Vehicular Ad-Hoc Networks are mainly aiming at well-developed countries to increase safety or convenience. Thus, we set aside concerns pertinent to military or civilian off-road use in favor of road-bound traffic, simply because roads are the main structuring element for traffic in such countries.

While the movement in general `MANET`s is arbitrary, the notion of being bound to a street creates a very strong correlation in movements, which we are going to exploit to increase packet-forwarding performance. Going from simple to complex, we first look into a simple scenario of cars traveling on the same highway.

As depicted in Figure 4.1(a), cars may move in different directions or in different lanes, but the main extent of movement is along the road, not across, which makes

the movement one-dimensional. Intuitively, this makes packet forwarding fairly easy if a node knows whether or not the destination node is in front or behind the sending car. With that simple fact, every radio neighbor being on the same side as the destination (and not further away from the destination than the sender) will reduce the distance from the destination, inevitably leading to delivery success, if the network is connected. With the increasing availability of GPS, detecting the "side" of a node is as simple as comparing positions with respect to the movement vector, and as we will see later in this chapter, this intuitive approach actually does the job where general MANET algorithms fail.

Without diving further into packet forwarding, and sticking to the pure movement patterns, the situation gets more complicated upon removal of the assumption that all cars travel on the same street. As depicted in Figure 4.1(b), the question as to which is the right "side" to which to forward a packet does not suffice any more. Moreover, even the shortest path might not reach the destination since this path could be disconnected where a longer path might do. While these scenarios are not restricted to inner cities, their characteristic map geometry gave them the name city scenarios. Sometimes, they are also called 2D scenarios to emphasize their more-than-one dimensionality.

So far, we can conclude that routing methods will most likely differ between highway and city scenarios, so it is valid to treat them separately. A city scenario can be seen as a web of connected highways (in the case of a curved street, this curve could be approximated by linear segments), creating an interesting possibility: First, solve the routing problem on highways, then use the solution as a building block for an algorithm that creates a sequence of junctions to reach the destination.

While most of these observations can be done using general knowledge, higher precision is required for research purposes. Moreover, movement patterns precisely defining the movements of single vehicles within a certain geographic area were needed. Within the framework of the *FleetNet* project, this was identified an important background information for routing research. Having the highest expertise in vehicle movements, project partner DaimlerChrysler provided know-how and data, both in highway *and* in city scenarios. This data was then processed and analyzed by our group, focusing on the impact on communication. The results of this work are described in the following sections.

4.3 VANET Packet Forwarding on Highways

In the following, we consider a highway to be a rectangular area stretching for thousands of meters in one dimension (called length), and for tens of meters in the other (called width). Also, we assume the vehicle radio range to cover the width well, whereas a multiple of widths fits into the length. For the movement speed

distribution we assumed a German autobahn scenario [18] with an unlimited top speed and a recommended speed of 130 $\frac{km}{h}$ ($\approx$ 36.1 $\frac{m}{s}$). This scenario appears to be among the most challenging scenarios since unrestricted high speed —to our knowledge — does not exist anywhere else in the world.

In the following, we describe first how the realistic movement patterns were generated. Then we outline the advantages of position-based forwarding over topology-based forwarding in this scenario by presenting a simulation study performed with the ns-2 network simulator. Finally, having shown the superiority of position-based methods, we discuss the impact of contention-based forwarding in vehicular highway scenarios.

4.3.1 Creating Realistic Vehicular Traffic Patterns

Vehicular traffic simulations can be classified coarsely into *microscopic* and *macroscopic* approaches [153].

When following a macroscopic approach, one focuses on system parameters like *traffic density* (number of vehicles per kilometer per lane) or *traffic flow* (number of vehicles per hour crossing an intersection) in order to compute a road's capacity or the distribution of traffic on a network of roads. In general, from a macroscopic perspective, vehicular traffic is viewed as a fluid compressible medium and, therefore, is modeled as a special derivation of the Navier-Stokes [44] equations.

In contrast, with a microscopic approach, the movement of *each* individual vehicle is determined. In order to generate vehicle movement patterns for ad hoc routing experiments one clearly has to follow a microscopic approach, since the position of each individual vehicle is needed. Nevertheless, one also has to take care that a microscopic simulation does not result in unrealistic macroscopic effects. As the vehicle movements are generated by a 'pre-process', and complexity is therefore a minor concern, we decided to use a *Driver Behavior Model* [296, 71] for the microscopic traffic simulation. Such a model not only takes into account the characteristics of the cars but it also includes a model of the driver's behavior, like lane changing and passing decisions, traffic regulation and traffic sign considerations, or decreasing speed in curves, to name only a few. Driver Behavior Models are known to be highly accurate and are therefore used by vehicle manufacturers, e.g., to determine the lifetime of certain parts of the car.

We used as a simulator the well validated DaimlerChrysler driver behavior simulation tool called `FARSI`. This simulator is regularly employed to generate traffic simulations in product development and evaluation at DaimlerChrysler. In particular, `FARSI` simulations show realistic speeds, distances, and macroscopic properties like traffic flow and lane usage, creating vehicle movement patterns very close to reality.

Highway Scenario

The simulated area in the highway scenario covers an area of 30 km length, with two lanes per direction and with an average number of six vehicles per kilometer and lane, representing light day traffic. Furthermore, the so-called 50%-desired speed parameter v_f (the parameter v_f splits the the population of vehicles into two halves: those with a desired speed of at most v_f and those with a desired speed greater than v_f) is set to 130 $\frac{km}{h}$. We assume that 15% of all vehicles are trucks. In `FARSI`, the oncoming traffic is generated as a separate simulation for a single direction, i.e., both directions are independent. The positions of the vehicles are recorded every half a second, together with current speed, lane identifier, and acceleration. From this file, we generated our ns-2 movement file by taking a 200 seconds slice of the scenario.

The described scenario corresponds to weak day time traffic on a German highway. In order to get an impression of the topology of a highway scenario at such a traffic density, a snapshot with realistic proportions for a highway segment of 500 m is given in Figure 4.2.

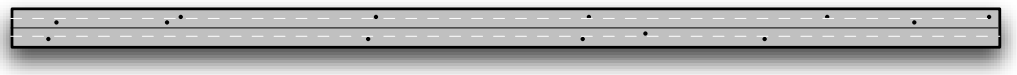


Figure 4.2: An undistorted 500 m highway segment with a traffic density of six vehicles per kilometer and lane taken from our generated movement scenario

Since the topology and the topological changes over time are of utmost importance for our routing experiments, we present in the following some properties of the generated scenario with respect to the distribution of velocities and lane usage.

Figure 4.3 shows the distribution of the speeds as generated by the the simulation (we quantized speeds into 10 $\frac{km}{h}$ bins). This matches very well the cumulative distribution taken from a 'real' measurement from a German highway.

The lane usage measurement for our highway scenario shows 57.2% usage of the left lane and 42.8% usage of the right lane. This is typical for light day traffic on a German highway since vehicles are only allowed to pass in the left lane and drivers tend to stay there longer than required.[1]

General Observations In order to gain an initial understanding of a (highway) *Vehicular Ad-Hoc Network*'s topology and its dynamics, we investigated the high-

[1] One has to note that speeds and lane usage depend on national regulations.

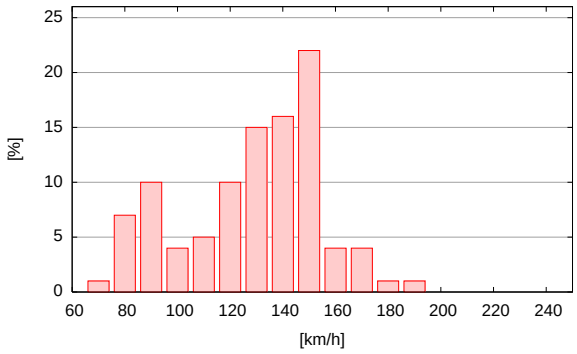


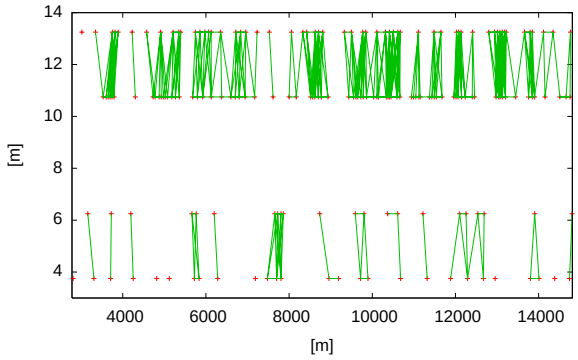
Figure 4.3: Distribution of speeds in a simulated scenario with 6 vehicles per kilometer and lane (graphically from [128⋆, 125⋆])

way scenario in a qualitative manner. Of particular interest was the theoretical connectivity of the *Ad-Hoc Network* formed by the cars. One question we wanted to answer was whether or not it is necessary to route packets over oncoming traffic in order to get acceptable connectivity. This question is important since routing over oncoming traffic implies rapid topological changes and potential problems on the physical level (Doppler effect, etc.). As a simplification, we first assumed that any two nodes can communicate if they are no more than 250 meters apart (approximating the behavior of `IEEE 802.11` without improved antennas). At the given average density of nodes (six per lane per kilometer), network partitioning should then be very rare if the positions of the nodes were equally distributed.

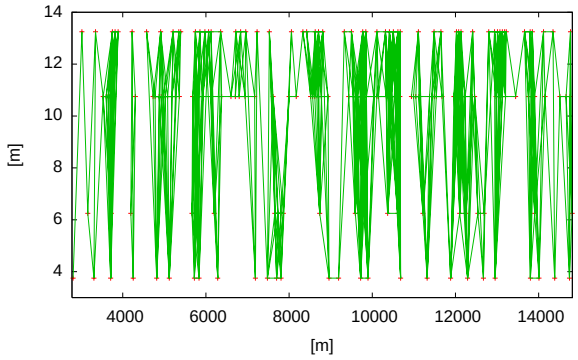
In order to determine the connectivity, we followed a designated node for 200 seconds on a 10 km path. For each node in a 3 km range of that node, we calculated whether it could be reached directly, and visualized it with a line between the nodes. This was done twice: in the first setting, there was no communication allowed between vehicles driving in opposite directions. Then, all vehicles on all four lanes were allowed to communicate with each other.

In contrast, Figure 4.4(b) shows the same situation when nodes in all four lanes are allowed to communicate with each other. It shows that partitioning of the network can be avoided by using oncoming traffic. An investigation of the full 200 seconds shows that most of the network partitions can be alleviated in this fashion.

In addition, we were interested in understanding how the number of network partitions depends on the communication range. Figure 4.5 shows the number of partitions on a 10 km segment with respect to the communication range of each individual node. Two graphs are given in the figure: the dotted one indicates the number of partitions when only vehicles driving in the same direction are consid-



(a) Connectivity when considering only nodes headed in the same direction



(b) Connectivity when considering nodes headed in both directions

Figure 4.4: Connectivity snapshot

ered for forwarding while the other graph describes the situation where all vehicles are taken into account. It can be seen that for the typical radio range of `IEEE 802.11` (250 m) denoted by the vertical blue line, there are seven partitions if only the vehicles driving in the same direction are taken into account. This is reduced to two partitions if all vehicles participate in the *Mobile Ad-Hoc Network*. Furthermore, the graphs show that a communication range of 400 m would be desirable to completely eliminate partitioning in this scenario when all vehicles are used or 1000 m if only vehicles driving in the same direction participate.

Based on these qualitative observations, it is necessary to route data packets over oncoming traffic even if the density of nodes headed in the same direction is quite high. If this is not done, network partitioning can be frequent, and each partitioning persists for a noticeable amount of time. Therefore an adequate technology

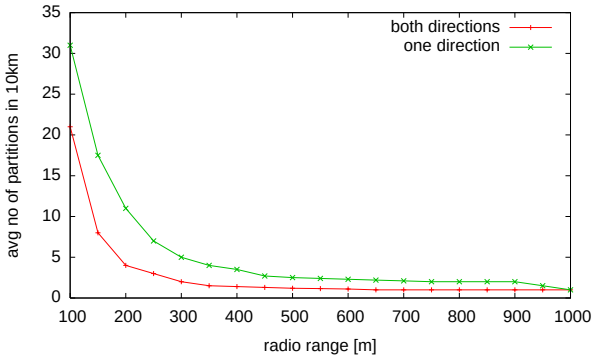


Figure 4.5: Number of partitions with respect to radio range (graphically from [128⋆, 125⋆])

for *Vehicular Ad-Hoc Networks* will have to support the routing of messages over oncoming traffic.

HWGui — A tool to process and analyze vehicular highway movements With `VANET` research emerging all over the globe, and given the necessity to work with the movement patterns, we have created a graphical Java tool to provide the necessary ways to process and analyze the movement data obtained from DaimlerChrysler. The tool and the parameters itself can be downloaded from [11], its description and a complete set of statistics can be found in [131⋆, 185⋆].

4.3.2 Forwarding in Highway Scenarios

In the following, we analyze the quantitative behavior of `DSR` and `GPSR/RLS` when applied to a network of vehicles driving in a highway scenario.

Simulation Setup

The environment used in the simulation is based on the all-in-one distribution of ns-2.1b8a running under Linux. The `GPSR` code of Brad Karp was ported to this platform. The `DSR` code used is the one delivered with the distribution. We took a time slice of 200 seconds of the input data and a reduced kilometer range of 10 km (position from 10 km to 20 km of the original data). This results in about 300 nodes in the scenario.

All experiments were conducted with two different `MAC`s. One was `IEEE 802.11` as provided in ns-2. The other one was an idealized `MAC` we implemented to ab-

stract from MAC-specific effects. This *0-MAC* allows communication between two nodes if they are 250 meters or less apart, and does not impose any upper limit on the amount of transmitted data. Collisions between distinct packets that are simultaneously transmitted do not occur with the 0-MAC.

Communication Pattern To select the communication pattern, we used the following algorithm. At any time there are ten pairs, each consisting of one sender and one receiver. These pairs are randomly selected such that they are no more than a maximum communication distance (in meters) apart from each other. In addition, they are guaranteed to be theoretically able to reach each other during the time they communicate (i.e., they do not reside in different network partitions). The sender then transmits four packets per second over a time of 5 seconds. The starting time is randomized in order to prevent synchronization. When a message is successfully delivered, the receiver sends a reply, modeling bidirectional datagram traffic. All packets carry a payload of 64 bytes. The maximum distance between senders and receivers varies from 500 meters to 4500 meters. Since the selection of partners is random (equally distributed) among the nodes fulfilling the constraints, sender and receiver can travel in the same or in different directions.

A Note about Border Effects When simulating a linear street scenario, one has to consider border effects. For instance, a node leaving the studied area has to be deactivated, for its real position is out of the scope of the simulation. To accomplish that, we used the energy model of ns-2. If a node reaches the border of the simulated area, it is deactivated, and reactivated (again) when it (re)-enters the scenario. In our scenario, since no node is allowed to travel backwards, each node is activated exactly once and deactivated at most once. When a node is deactivated, it stops sending GPSR beacons.

DSR Setup The parameters originally set in the ns-2 implementation of DSR were kept for our simulation. The only modification was done to increase the maximum hop distance that a DSR route can span from 16 to 32 so that it is possible to reach all destinations even in the 4500 meter communication pattern. For a deeper understanding of DSR optimization, please refer to [165]. In our simulation, DSR uses the promiscuous mode of the network interface to investigate all packets receivable, regardless of the destination address.

PBR and RLS Setup The position-based routing algorithm we used is based on the code of GPSR [171], except that the perimeter mode was turned off. The setup is as follows: The beacon information of a node, i.e., its own position, is piggybacked on to every packet (data packets and location service packets) that it forwards.

When piggybacking a beacon, the node resets the timer for the scheduling of its next beacon. We varied the beacon interval between $\{0.25, 0.5, 1, 2\}$ seconds in order to study its influence on the rate of successfully delivered packets and routing overhead. We make use of the MAC callback feature, enabling a node to reroute packets still buffered by the MAC if a MAC link breaks. Although this is a violation of the strict layer separation, the gain it affords is remarkable according to [171].

Our *Reactive Location Service* was used first with a linearly expanding ring search and then with with an exponentially expanding ring search. The timeout value for triggering the flooding with an increased range was set to 100 ms multiplied by the number of hops in the last cycle. The maximum hop count for the flooding was set to 32, the same value used by DSR. This should enable us to reach any node in the simulated area. Each data packet and each reply sent in response to a data packet contains the ID and location of its sender and its receiver. Thus, the location information about a communication partner is updated upon the receipt of a packet from that communication partner.

Simulation Results

0-MAC In order to gain an impression that is unaffected by the properties of the MAC, we started the simulations by using the 0-MAC. The first experiments were conducted for the position-based approach with a linear expanding ring search (increase of 1 hop per cycle). Surprisingly, we had encountered cases where a destination node was not reached by the flooding. A more detailed analysis helped us to understand the reason for this: The problem occurs when two vehicles are driving in different directions want to communicate. For the first flooding, a range of 2 was used, while the vehicles were n hops apart (n was greater than 2). Flooding with range 2, therefore, remained without success. However, during the time required for the first cycle to time out, the cars moved in opposing directions, so that they now were at least $n + 1$ hops apart: the expanding ring search was slower than the vehicles. We concluded that for vehicle communication, linearly expanding ring search is not suitable for a location service. Thus, in the following we consider only the exponentially expanding ring search.

One key performance metric for the suitability of a given approach is the rate of successfully delivered packets. Figure 4.6 shows this metric for DSR and GPSR with increasing maximum communication distances. There is just one plot for GPSR since all tested beaconing frequencies provided the same results in all ranges. This is no surprise since the flooding for the location service allows to piggy back the beacon information of all nodes between sender and receiver at the beginning of the communication. Furthermore, the data packets sent will also be used for piggy backed beacons and to keep the information about neighbors up-to-date. We

tested beaconing frequencies with up to 16 seconds between beacons without a major change in the outcome of the experiment.

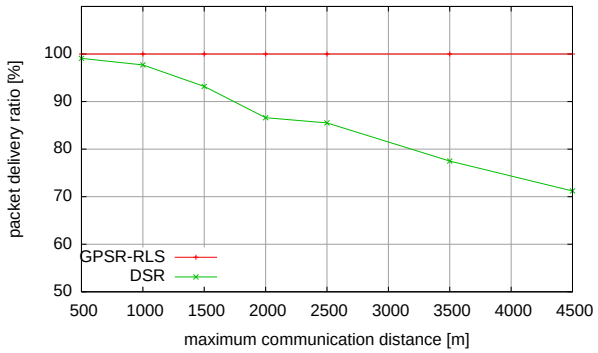
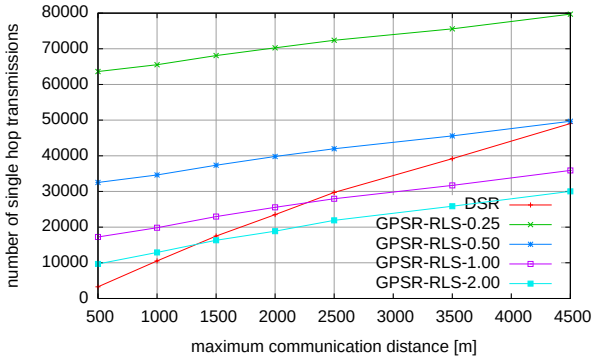


Figure 4.6: PDR with respect to maximum communication distance (0-MAC)

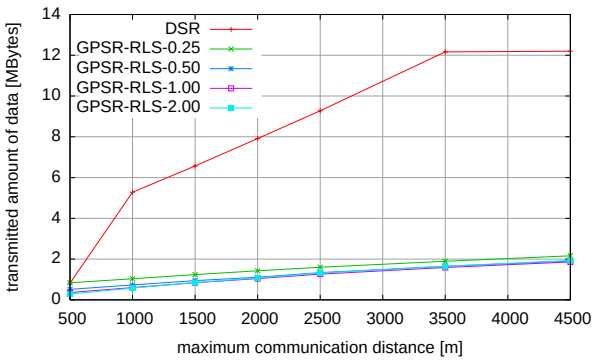
Figure 4.6 can be interpreted as follows: As expected, the rate of successfully delivered packets for DSR diminishes as the maximum communication distance becomes larger. This is caused by the fact that DSR needs to maintain a route from the sender to the receiver, which becomes harder when the length of the route increases. The position-based approach stays at the perfect packet delivery rate of 100% for all distances.[2] This can be explained by the properties of position-based approaches: Packet drops can occur only for one of the following three reasons: (1) if a local maximum is reached; this is very unlikely in our scenario; the reason for this is displayed in Figure 4.8: F is the forwarder and D the packet destination, just outside F's radio range; there are no greedy forwarders, but a route exists using N and O; however, this only can happen in the small area between the red iso-progress line from D and the blue dashed radio range of O; this area gets smaller, the further away D is; (2) if the information about the position of the local neighbors is inaccurate; again, this is very unlikely since the flooding of the location service in combination with piggy-backed beacons will provide nearly perfect information about the neighbors; (3) if the information about the position of the destination is inaccurate. This is also very rare since when using the 0-MAC, the reply containing the position of the destination requires only minimal time to reach the sender; thus, it is very accurate when the data packet is transmitted.

Besides looking at the delivery rate, it is also important to investigate how many packets and how much data are required to transmit a certain amount of payload data. We therefore measured the total number of one-hop transmissions that oc-

[2] It should be emphasized that we did not try to "optimize" the simulation to achieve this figure besides the selection of communication pairs that are able to communicate.



(a) Number of 1-hop packet transmissions



(b) Volume of transmitted data

Figure 4.7: Analysis of communication costs

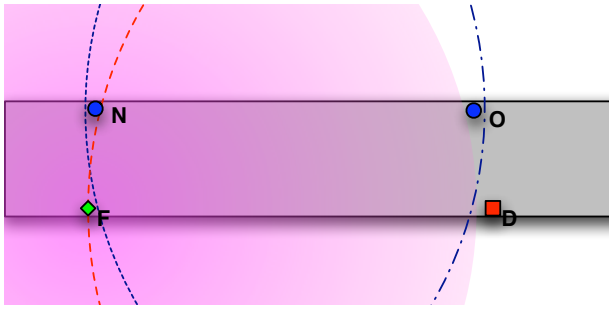


Figure 4.8: Highway void situation

curred over the whole lifetime of the simulation. This is shown in Figure 4.7(a). Both unicast and broadcast messages are included in this figure. For GPSR, we show the communication costs for all beaconing frequencies. It can be seen that the value for DSR starts low if the maximum communication distance is small, and grows quickly with increasing communication distances. This is caused by the increase in overhead for route establishment and maintenance, which are the main sources of packets for DSR (besides the actual data packets). GPSR/RLS, on the other hand, starts at a higher value, and then increases more slowly. Furthermore, it can be noticed that the communication overhead scales almost linearly with the beaconing frequency. The reason for the observed behavior is that beacons are the dominant source of one-hop transmissions in position-based routing. These are independent of the maximum distance between communication partners. Since the packet delivery ratio is almost independent of the beaconing frequency, and since beaconing provides the dominant amount of one-hop packet transmission, it seems appropriate to use a fairly low beaconing frequency when employing GPSR/RLS for vehicular networks.

Figure 4.7(b) displays the total amount of data used in the form of single-hop transmissions. It demonstrates that DSR needs significantly more data than GPSR for all examined maximum communication range values and beaconing frequencies. This is caused by the size of the packets needed to establish and maintain routes in DSR. Since these packets need to carry a source route from the sender to the receiver, they can become quite large. In contrast, the packet size of GPSR/RLS is very small: All that is required is the position information and the ID of the sender (and of the receiver if it is a data packet).

IEEE 802.11 In a second round, we repeated the experiments, using the default implementation of IEEE 802.11 in ns-2 as MAC. Given the results from the previous section, we expected similar, but somewhat less optimal results. In our initial experiments with IEEE 802.11, we were surprised to see that GPSR/RLS actually performed similarly to, and sometimes worse than DSR with respect to the rate of successfully delivered packets. In particular, the exponentially expanding ring search frequently failed to reach the destination node. Investigating this problem, we noticed that the flooded packets tended to synchronize themselves such that they caused collisions at the MAC layer. In IEEE 802.11, broadcast packets that are affected by such a collision are not retransmitted and remain lost. Thus, the synchronized broadcasting of packets can lead to a complete wipe-out of the affected packet. As a consequence we introduced a jitter when sending broadcast packets for the expanding ring search. This solved the problem.

Figure 4.9 shows the packet delivery rate for the simulation with IEEE 802.11. Generally, the outcome is very similar to that of the 0-MAC case. However, there

is one minor detail that is worth mentioning: For GPSR/RLS, we had some runs where data packets got lost, even though the vast majority of runs were completed without a single packet loss. The main reason for those losses was that beacons and broadcast packets from the location service would sometimes still collide. Thus, the information about the position and availability of neighbors is less accurate in the simulation runs with IEEE 802.11. This sometimes causes a forwarding node to be ignorant of the only neighbor with forward progress in the direction of the destination. The cost of the communication remained very similar to that of the 0-MAC case, and is therefore not shown here.

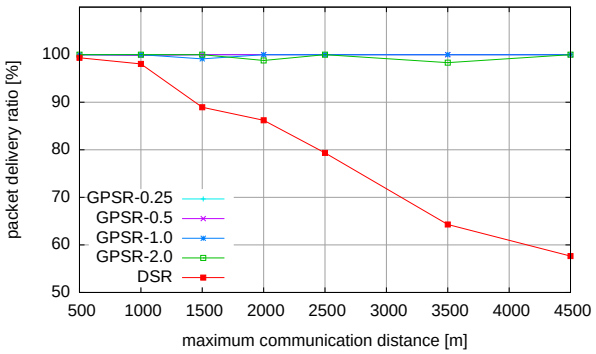


Figure 4.9: PDR with respect to maximum communication Distance (IEEE 802.11)

4.3.3 Improving Position-Based Forwarding on Highways

The last section clearly shows that beyond a certain scope of hops, topology-based forwarding ceases to work while position-based forwarding still performs quite well. While involved in the *FleetNet*, and later in the *NoW* project, it became clear that position-based forwarding with beacons like GPSR will probably be the desired protocol base for a future VANET communication system. The reasons for this can be summarized as follows:

a) VANET safety applications will most likely announce their position anyway.

b) To implement *Contention-Based Forwarding* efficiently, the MAC timers would have to be integrated, while beacon-based methods are more indifferent to the underlying timers.

c) Some *FleetNet* partners were somewhat afraid of CBF and the ongoing patenting process.

Thus, we did not drop the idea of beacons, although CBF already promised to outperform beacon-based routing on highways. However, further investigations led to the following results.

Beacons and Dead-Reckoning

As in general MANETs, the basic remaining reason for packet loss with beacon-based routing is the inaccuracy of neighbor tables, i.e., a neighbor that is still believed to be in range might already be gone while another neighbor that could be used for forwarding was not entered into the neighbor table because its beacon has not yet been received. While the second situation is not very harmful at reasonable node densities, the first one is critical because (a) it requires multiple transmission attempts to realize a neighbor is gone (among other things, seriously increasing the delay jitter), and (b) the greedy selection process is likely to select neighbors that are close to the border of a node's radio range increasing the probability of disappearing neighbors further.

Apart from making the next hop selection process forwarder-centric as in CBF, the standard approach to mobility management[3] is the use of so-called dead reckoning [31, 60]. Known for a long time in ship navigation, this basically means the geometrical extrapolation of a known position by adding movement vectors. The resulting position estimates are then used if no real position fix is available. This strategy should alleviate the inaccuracy in the time between two consecutive beacons.

As a direct consequence of this thinking, we derive the following strategies to improve position-based forwarding:

1. Use dead-reckoning position estimates for greedy neighbor selection
   a) by deducing a movement vector from two consecutive beacons
   b) by including a movement vector in the beacon message.
2. Worst-case estimate if a neighbor might have gone out of range since the last beacon, and use it only if it could not have.

These strategies result in a different performance gain, depending on scenario parameters like node density, speed etc. As shown below, e have studied them by means of simulation. For more information about the simulation, please have a look at [175*, 176*].

[3] Mobility management usually stands for the prediction of node movement in cellular networks, which is a huge area of research.

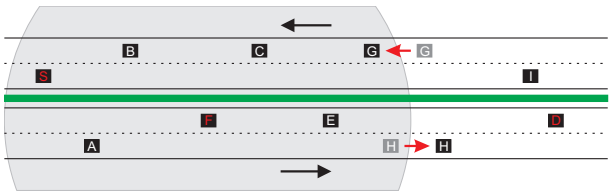


Figure 4.10: Example neighborhood situation on a highway

Simulation Study Setup With PBF as a name for beacon-based greedy forwarding, position information is only precise at the time of sending. The longer ago this happened, the less accurate this information gets. Especially at high beacon intervals, it is thus likely to select a next-hop that is no longer reachable. This is especially true for nodes traveling at high relative speeds such as nodes moving in opposite directions. For instance, Figure 4.10 depicts a scenario where node S communicates with node D. Here, node F carries a packet from S to D and is searching for a next hop. The neighbor positions as perceived by node F are colored in gray, while the real positions are colored in black. As can be seen in the picture, the most critical consequence of position inaccuracy occurs when nodes like node H are selected. In that case, the link-layer has to perform costly retransmissions only to find out the next hop is not reachable any more[4]. Here, H is very likely to be chosen by an MFR heuristic since it provides excellent progress. On the other hand, node G could have been used for forwarding if we had already received a beacon from it. This error could be packet-fatal when there are no other potential next-hops in the neighbor table.

The first error can be alleviated if F tries to avoid selecting H as a next hop, trading in longer routes. To study these effects, we have implemented two next hop selection algorithms based on dead-reckoning and one simple defensive neighbor selection approach. The main goal of these algorithms is to reduce the probability that a selected next hop has already left the communication area of the sending node. The *Active* and *Passive Anticipation* algorithms are dead-reckoning approaches and estimate the position of a neighbor between two consecutive beacons. The *Defensive Neighbor Selection* heuristic modifies MFR by eliminating neighbors that could have left the communication range under utmost pessimistic conditions. We will describe the algorithms in detail in the following paragraphs.

Active and Passive Anticipation The fundamental assumption behind the dead-reckoning strategy is the continuity of a car's movements, i.e., when observing a car

[4] In reality, the boundary between receiving and not-receiving is weaker than in this simplistic radio model. However, even then, position inaccuracy would show similar effects, but be much harder to quantify.

moving on a highway at a certain speed and losing the perception of this car, it can be assumed that the car will continue driving in the same direction and at the same speed. In other words, the last known movement vector is used to extrapolate the present position of the vehicle. Plain `MFR`, in contrast, would use the last known position instead.

The *Anticipation* algorithms use this dead-reckoning approach and work as follows: Assume that node A wants to estimate the position of its neighbor B. Then it first calculates the time t_B that has passed since the last beacon of node B was received. It multiplies t_B with the speed and driving direction of node B and adds the resulting vector to the position p_B stored in the neighbor table. The result is the estimated position $\hat{p}_B$. This is done each time a node forwards a packet.

The two anticipation algorithms differ in the way they obtain speed and driving direction. The *Active Anticipation* algorithm requires speed and driving direction to be contained in the beacons. Thus, as soon as a node receives a beacon of a neighbor, it also knows its speed and driving direction. The speed is measured in meters per second, and the driving direction is measured in degrees. Clearly, adding theses values to the beacons results in a larger beacon packet size.

In contrast, the *Passive Anticipation* algorithm computes the driving direction and speed of a node out of the values from two consecutive beacons. This comes at no additional per-beacon cost but does delay the estimation for one beacon interval since two beacons are required to construct a movement vector.

Defensive Neighbor Selection The *Defensive Neighbor Selection* algorithm makes an assumption about the maximum driving speed v_{max} of vehicles in a certain scenario (e.g., with a speed limit) in order to define the area where possible forwarding nodes have to be located. The basic idea is that it is not the communication range that defines where possible forwarding nodes have to be located. Instead, a smaller area is used to make sure that a selected node is definitely reachable. This concept avoids the occurrence of falsely selected next hops at the expense of possibly longer routes.

The algorithm works as follows: When a node A wants to determine whether its neighboring node B is a next hop candidate, it looks up the time t_B since the last beacon of node B was received. Then, it calculates the distance d_{AB} of node A and B based on the values stored in the neighbor table. After multiplying t_B with $2 \times v_{max}$ and adding the result to the distance d_{AB}, it obtains the maximum distance d_{max} between itself and node B. If d_{max} is smaller than the maximum communication range, it will be guaranteed that both nodes are in communication range of each other and node B is finally marked as next hop candidate. We have chosen $2 \times v_{max}$ because we do not differentiate whether the neighboring node moves in the same

or opposite direction as node A does. However, our intention is to guarantee that a next hop candidate is still in communication range of the sending node.

After all next hop candidates have been determined, `MFR` can be used to select the best next hop.

The environment used for the simulations is based on a modified all-in-one distribution of ns-2.27. We applied all bug-fixes of the ns-2.28 and ns-2.29 and implemented the `PBF` routing protocol including plain `MFR` and the optimizations proposed above. The *Reactive Location Service* [193⋆] was used to obtain the positions of the destinations. `IEEE 802.11` with a (unicast) transmission rate of 2 $\frac{MBit}{s}$ was used as `MAC`. We set the transmission range (Two-Ray-Ground Model) to 500 meters by adjusting the transmission power to reflect results from real-world experiments conducted on German highways [149⋆].

The highway traffic scenarios were selected from the FleetNet Highway Movement Patterns [125⋆, 131⋆], originally based on movement patterns issued by DaimlerChrysler. Each movement scenario is a two-way highway with two lanes per direction. The length of the highway in each scenario is at least ten kilometers. As node movements we selected *sparse* (two nodes per lane and kilometer in one direction and six in the other), *medium* (six nodes per lane), and *dense* (six nodes per lane in one direction and eleven in the other) node density scenarios.

To ensure theoretical connectivity, the communication patterns were set up as follows: A node A is randomly chosen. If a node B in distance of 6750-7250 meters can be found which is theoretically reachable[5] for at least 60 seconds, node A and B will be temporarily selected as sender and receiver. If it is further possible to find nodes between A and B that are in distances of 250-750, 750-1250, ..., 6250-6750 meters and that are theoretically reachable for at least 60 seconds, one node for each of these ranges will also be selected as a receiver. Otherwise a new iteration with another node A will be started.

We have used a simple ping application to generate traffic between senders and receivers. Every 100 milliseconds a 128 byte ping packet is transmitted by the senders. After receiving a ping packet, the receiver will echo back a pong packet. The number of ping packets for each communication pair is limited to 40. Communication starts at $t = 5$ seconds and lasts until $t = 60$ seconds. Every 5 seconds another randomly selected sender-receiver pair starts communicating. As input for the graphs in this paper, every simulation setup has been calculated ten times using different seeds for the random number generator.

For all simulations we have calculated — among other metrics — the packet delivery ratio and the `MAC` layer cost (number of transmitted bytes). Additionally,

[5] Theoretical reachability means that a (multi-hop) route between the sender and the receiver exists under the assumptions that every node has perfect information about the position of neighboring nodes and that a packet transmission is instant.

we have looked at the average positioning error of selected next hops and the route length.

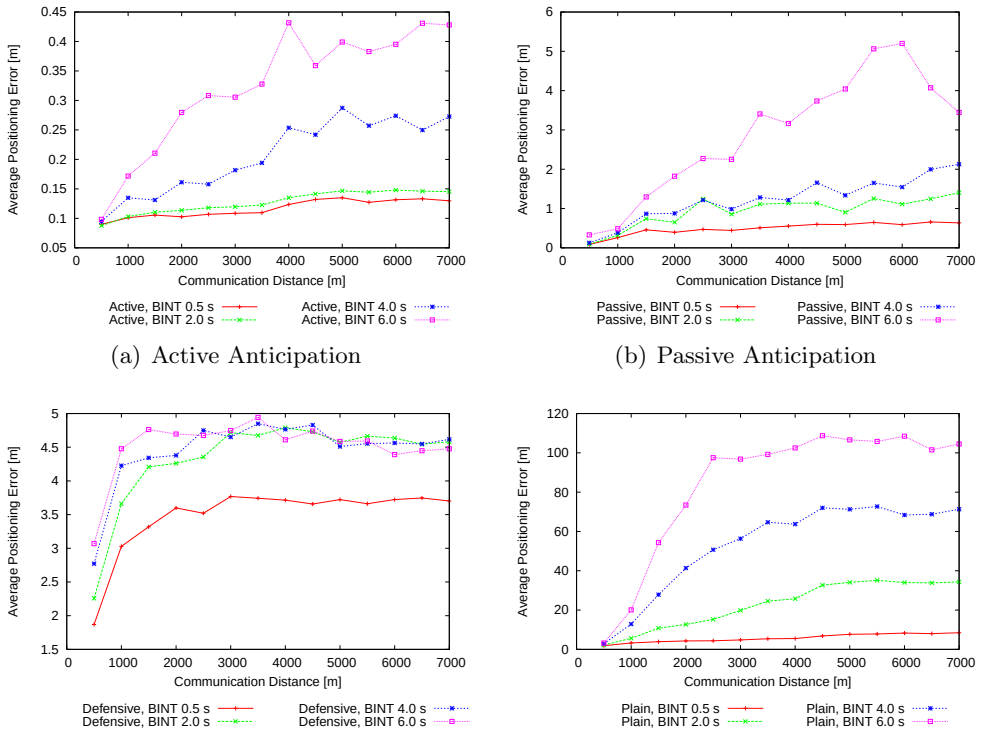
Simulation Study Results First, we investigate the average positioning error of the next hops selected to forward a packet. We expect that the error is proportional to the BINT multiplied by the average node speed and independent of the communication distances. Figure 4.11 shows the average positioning error of the selected next hops in a medium density highway scenario. In order to obtain the graphs, each next hop selection strategy was simulated with different BINTs of 0.5, 2, 4, and 6 seconds. The simulations show that the error grows with an increasing communication distance. Which is explained by the fact that each transmission creates a sample for the positioning error, and a higher communication distance results in a higher number of transmissions and, thus, in more samples. The more samples a value is composed of, the higher is its significance and the better it approximates the theoretical value.

The main purpose of the graphs in Figure 4.11 is the quantitative comparison of the different forwarding/estimation strategies. Figure 4.11(d) shows that plain MFR with a BINT of 0.5 seconds produces an average positioning error of almost 10 meters. The error grows up to 100 meters for a BINT of 6 seconds. In contrast, the Defensive Neighbor Selection algorithm achieves an average positioning error of approximately 3.5 meters for a BINT of 0.5 seconds and between 4.5 and 5 meters in the 6-second BINT case.

Passive Anticipation performs slightly better. It reduces the average positioning error to less than 1 meter for a low BINT, and 5 meters for a high one. The results for the Active Anticipation algorithm are even better. It notably outperforms all other strategies. The average positioning error is lower than 0.5 meters even in the 6 seconds BINT case. These results show that even a simple optimization like dead-reckoning can reduce the average positioning error to less than 1 meter for reasonable BINTs.

The results for sparse and dense node density scenarios were in principle very similar and thus we do not present them in this short paper.

Figure 4.12 presents graphs that show how often a selected next hop was not reachable. The graphs for the Defensive Neighbor Selection and the Active Anticipation are omitted here since these algorithms did not select any unreachable next hops. As outlined above, the selection of unreachable hops is PBF's biggest problem due to the extra load inflicted by unnecessary link-layer retransmissions. Figure 4.12(b) shows that the percentage of unreachable next hops for plain MFR depends on the BINT; the larger the interval, the larger is the percentage of unreachable next hops. E.g., even for a beacon interval of 500 milliseconds, 5 percent of the selected next hops were out of range at a 4 km communication range. In



(a) Active Anticipation

(b) Passive Anticipation

(c) Defensive Neighbor Selection

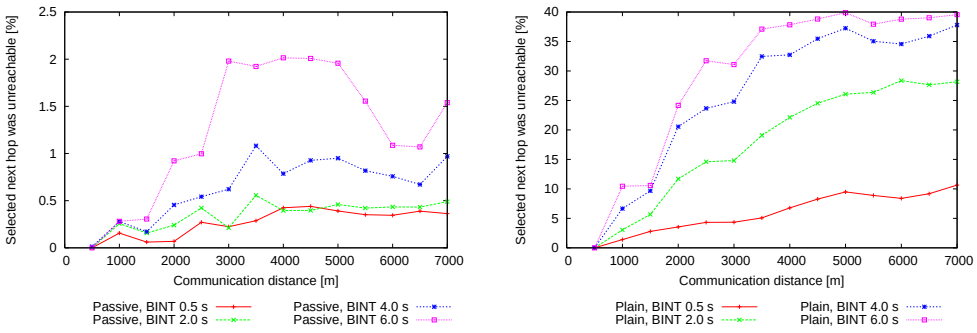
(d) Plain `MFR`

Figure 4.11: Average positioning error of selected next hops in a medium highway scenario

contrast, even with a `BINT` of 6 seconds, Passive Anticipation selects next hops that are reachable at 98 percent.

Examining the consequences of unreachable next hops for the wireless channel, we computed the amount of data that was produced at the `MAC` layer during a simulation. We define "MAC layer cost" as the number of packets sent during a simulation multiplied by their respective packet size. With the radio model used, every unreachable neighbor causes the maximum number of retransmissions to occur, adding to the cost factor. Hence, in a scenario where the sender and receiver were 7000 meters apart from each other, `MAC` layer costs vary from nearly 4.5 megabytes in the Active Anticipation case to nearly 20 megabytes in the plain `MFR` case. The reason for the difference of 15.5 megabytes are the retransmissions of the `MAC`; by default, packets are resent seven times before the `MAC` quits.

While selecting an unreachable next hop is unwanted, being too defensive in picking forwarders creates unnecessarily long routes. Figure 4.13 depicts the impact of



(a) Passive Anticipation (b) Plain MFR

Figure 4.12: Percentage of unreachable next hops (medium car density)

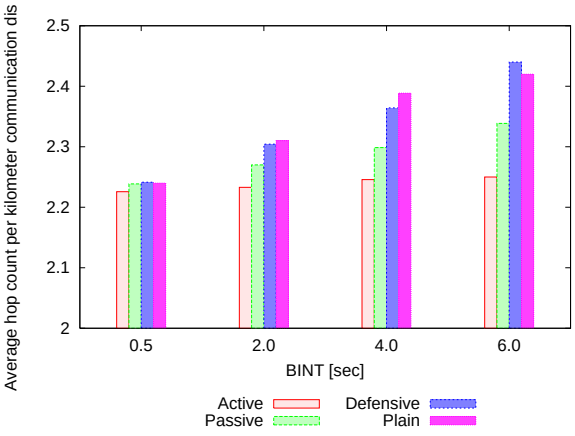


Figure 4.13: Average number of hops per kilometer

different next hop selection strategies on the route length. The graph shows the average number of hops per kilometer communication distance with respect to the beacon interval. Each next hop selection strategy is represented by a differently shaded box. Active Anticipation shows an almost constant hop count independent of the beacon rate. Furthermore, Active Anticipation outperforms the other selection strategies by requiring on average only 0.2 hops more than the theoretical minimum, which is two hops for a communication distance of 500 meters. In contrast, plain MFR requires 2.25 hops in the 0.5-second BINT case and more than 2.4 hops in the 6 second BINT case. With a large BINT, the Defensive Neighbor Selection approach performs even worse and requires more than 2.425 hops.

Figure 4.14 shows the average single hop delay of the different protocols on a logarithmic scale. Also, lain PBF's inherent trial-and-error strategy is also very

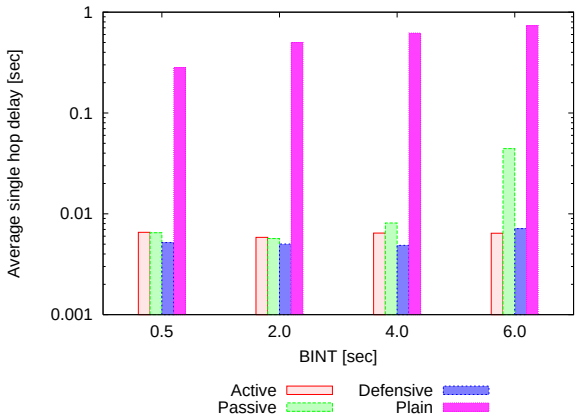


Figure 4.14: Average single hop delay

costly in terms of packet delay, even in these low-load communication scenarios. Delay and cost could both be reduced by a lower-than-default number of MAC retries since next hops that are out of range would be realized much sooner.

The Packet Delivery Ratio (PDR) performance is left out on purpose since it showed the same tendencies already published in [125⋆]. All methods show almost-perfect packet delivery for the higher beaconing rates with decreasing PDR for lower rates and increasing communication distance. However, the optimized methods sustain perfect delivery for all studied beacon intervals.

Contention-Based Highway Forwarding

As stated earlier, the suitability of position-based forwarding — both with and without beacons — derives from the geometrical nature of the street with respect to the radio range. Ultimately, this evades void situations, making greedy forwarding perfect to route a packet to a destination. Hence, it also enables plain CBF (see Chapter 3), i.e., without a recovery strategy, to find any destination on a highway. Moreover, the same geometry theoretically makes for almost-perfect suppression of packet duplicates. Looking at Figure 4.15, we assume node F to have just completed the transmission of a CBF packet destined for D. The blue node N resides at the very border of F's nominal radio range, making it perfect for forwarding. When it sends its packet to all nodes within the blue dashed circle (its own radio range), the only suitable forwarders that might possibly not hear the transmission would be in the tiny area between the blue dashed circle, and the red dashed circle representing the iso-progress line to the final destination. For every forwarder N significantly less suitable, this area would disappear.

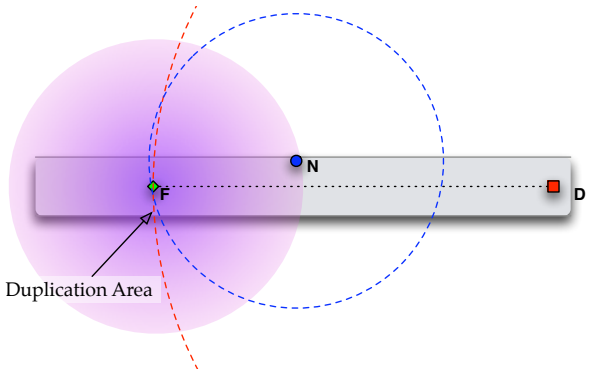


Figure 4.15: CBF packet duplication on a highway

Having deduced that CBF is a very promising candidate for highway forwarding [124⋆], it was still questionable if it is able to outperform the improved beacon-based schemes described above. Practically, the performance of both methods almost converges with the beacon-methods. However, optimized explicit forwarding is slightly more damageable by improper parameter settings, and CBF lacks the additional comfort a neighbor table provides.

However, there is one thing still unaccounted for, which is the simplified modeling of radio propagation. In a study performed together with the University of Karlsruhe [262⋆, 263⋆], we have simulated an optimized beacon scheme competing with CBF, using a slightly more accurate radio model. To be more precise, the improved model now accounts for reception probability with respect to node distance, i.e., the higher the distance between nodes, the lower the probability of reception is. Still, even at short distance, there is a likelihood of not hearing a packet, and at high distances, there is a slight chance of receiving a packet.

Beacon-based forwarding has difficulties exploiting beacons from far away. If the protocol tried to use them, i.e., by adding their senders to the neighbor table, the probability that a subsequent packet would be overheard by the potential forwarder is very low. This leads to the necessity of suppressing neighbors beyond a certain range or below a certain SNR [52]. CBF, however, copes very well with this radio situation, and is fully capable to exploit far-away forwarders, which occur quite often in dense scenarios. The only problem it has is that in this case, a forwarder might not hear the re-transmission and would thus likely try again, resulting in packet duplicates. Since the significance of this obvious problem is not easy to quantify, [262⋆, 263⋆] estimates the performance impact, with astonishing results. Even with the packet duplicates, CBF outperforms beacon-based forwarding significantly. While both methods reach almost-perfect packet delivery, the overall

transmission costs of CBF and the effective path length are significantly lower than when relying on beacons, resulting in very low packet delays.

Selected Simulation Results To back up the statements made above, we present some simulation results. For details about these issues, please refer to [262⋆, 263⋆]. As a specialty to this simulation we have also tried the protocols with probabilistic radio modeling following the Nakagami Model [249].

Simulation Set Up The utilized simulation tool is the network simulator ns-2.28 [15]. However, its MAC/PHY implementation has been adapted to IEEE 802.11p [17], a variant of 802.11a still not standardized, which is the technology that will most likely be used for VANET communication.

Our intention is to analyze how different distances between sender and receiver, and a different radio propagation model (deterministic and probabilistic) affect the performance of the routing algorithms in both directions of a communication. For this purpose, we simulated the highway scenario described in Section 4.3.1, where among all possible nodes we selected two specific vehicles (one communication pair) to exchange 10 ping packets (request/reply). We performed several simulations where we increased the distance between the two nodes forming a communication pair, up to 4500 m. A larger distance results in an increased number of hops since the communication range of all nodes is constant during the whole simulation. We selected a 500m m intended communication range as reasonable 1 hop maximum distance in ideal conditions and absence of interferences (IEEE specifies a range up to 1000m m for this technology).

In each simulation only one communication pair was selected, while all other vehicles on the road would only be potential intermediate nodes. The communication partners were picked such that they were in theoretical multi-hop range, meaning that when applying a unit disk graph model, the resulting graph contained routes between them during the whole communication time. In addition, they remain within the same distance range (500 m wide) during the whole packet exchange. For example, if the studied distance was 3500 m we can be sure that during the simulation time the two nodes are between 3000m m and 3500m m apart and there are always enough vehicles in between to connect them via multi-hop.

In order to have statistical significance, we selected 10 different scenarios (with the same number of lanes and density) from the whole set of traffic patterns. In each scenario we selected 10 different communication pairs (originator/destination) and ran independent simulations with each one of them. Finally, for each configuration setting, we compute the average and the confidence interval (with 95% confidence level) of the studied metrics, see Sec. 4.3.3.

The main configuration parameters are reported in Table 4.1. While we have simulated many other settings, we will stick to these to describe the effects found.

Table 4.1: Highway simulation: CBF configuration parameters

Studied protocols	PBF, CBF, AODV
Radio propagation models	Two-Ray-Ground, Nakagami
Distance between comm. pair	500 m to 4500 m
Intended comm. range	500 m
Ping packets generation rate	4 packets/s
Packet size	64 byte
Number of Ping packets	10
PBF beaconing interval	2 s
CBF max. contention time (T)	20 ms
Vehicle density	12 $\frac{cars}{km}$ / 22 $\frac{cars}{km}$

Results To compare the performance of PBF, CBF and AODV under both types of radio channel conditions we have plotted different figures representing their behavior when increasing the communication distance with respect to the selected metrics.

We can observe in Fig. 4.16 the performance of the different protocols under both types of propagation model, Two-Ray-Ground and Nakagami. Fig. 4.16 reports the packet delivery ratio for different distances between sender and receiver. Note that since the intended communication range of all nodes is fixed to 500 m selecting a destination node 500 m further from the sender is equivalent to adding, at least one hop to the resulting communication path. As expected, AODV achieves the lowest packet delivery ratio, further decreased under non-deterministic radio propagation. In more detail, we observe that communication fails mainly due to two reasons: *i)* mobility, i.e., some chosen nodes drove far from their previous/next hop, significantly decreasing the probability to forward a packet successfully, and *ii)* the random behavior of Nakagami made a too optimistic route choice [212], i.e., some intermediate nodes where 'quite' far from each other so the data flow had a low probability to reach its destination. When not only mobility but also received signal strength fluctuations are considered, the search and use of a fixed route turns to be the worst choice.

Position-based routing protocols are robust against both node mobility and fading. Both schemes show average bidirectional delivery rates higher than 99.7% for all simulated distances and propagation models.

To better understand the effect of using different propagation models we can take a look at the total load sent into the channel resulting from the different routing algorithms, Fig. 4.17. First, we observe a good performance of all strategies when

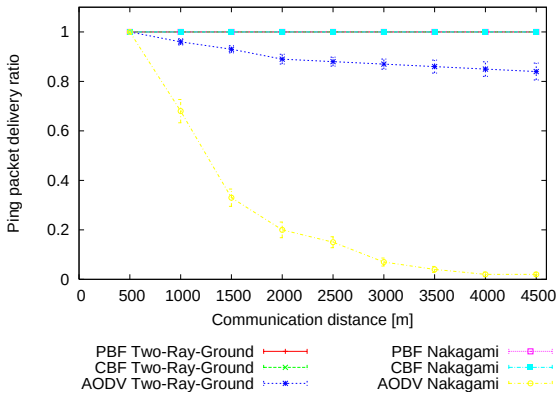


Figure 4.16: PDR of PBF, CBF and AODV when increasing the distance to destination for Two-Ray-Ground and Nakagami. The graphs of PBR and CBF for Two-Ray-Ground and Nakagami all cover each other at a delivery ratio of one.

dealing with Two-Ray-Ground, only AODV results in a significant increase of load with distance; note a constant higher channel load for PBF due to the utilization of beacons. Second, note the high increment of the experienced channel load of PBF and AODV under Nakagami when increasing the number of hops between the sender and the receiver. The difference between PBF and CBF responsible for their different performance is the strategy to select the next forwarding node. A node using CBF broadcasts a message and just expects that one node, which is closer to destination than itself, receives the packet and forwards it. PBF, on the other hand, selects a specific node from the neighbor table and tries to communicate with it. The use of a non-deterministic propagation model notably increases the risk that a successful data exchange between an intermediate node and its next hop needs more than one MAC retry, or more than one neighbor in a worse case. That explains the high increase of transmitted load with respect to the number of hops of PBF with Nakagami. Similarly, AODV increases its resulting load. When AODV routes hold, they tend to need many retries since the neighbors have been chosen poorly. When AODV starts losing routes the higher the distance between communicating nodes gets, the route request process time-outs block packet sending at the original sender. This limits the number of packets that are sent at a total resulting load of 500 $kByte$.

Finally, we plotted the round trip time experienced by the different protocols in Fig. 4.18. We can see how the results are in line with the former figures. The worst performance, i.e., the longest round trip time, corresponds to AODV, especially under the probabilistic propagation model. If we take a look at the zoom (the square

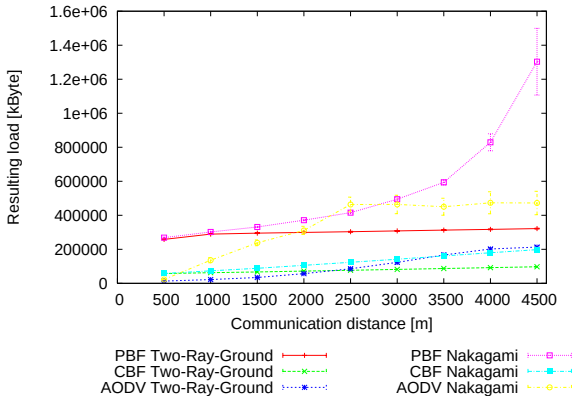


Figure 4.17: Resulting load on the medium from PBF, CBF and AODV when increasing the distance to destination for Two-Ray-Ground and Nakagami.

inside Fig. 4.18), we can see how the performance of PBF under Nakagami is affected when increasing the distance between the communication pairs. Also in the zoom of Fig. 4.18, we can observe an interesting phenomenon, CBF shows shorter round trip times when considering a non-deterministic propagation model, and PBF shows longer ones.

To explain this behavior we also plotted the average number of hops for both protocols for the different communication distances, Fig. 4.19. Again, we see the benefit of not pre-selecting the next forwarding node in the process of routing a packet when considering a non-deterministic radio model. As mentioned before, PBF selects a node inside its communication range and tries to communicate with it. It is reasonable to think that the unreliability of the link results in a longer round trip time, i.e., it will use several MAC layer retries (or even select a new node) before being acknowledged. On the other hand, CBF does not make any assumptions about its communication range to select the next forwarding node. That way, CBF benefits when a node outside of its intended communication range receives the packet, which is a possible situation only when considering a non-deterministic propagation model. That explains that, e.g., the average number of hops could be smaller than 8 when the destination is further than 4000 m and having all nodes an intended communication range of 500 m.

As conclusion, we can state that CBF presented the best performance among the routing protocols. We can observe that the impact of signal strength fluctuations does not have a critical impact on CBF's behavior. Its strategy of not selecting a specific forwarder before the actual message transmission, is the most robust scheme to fight against fading channels.

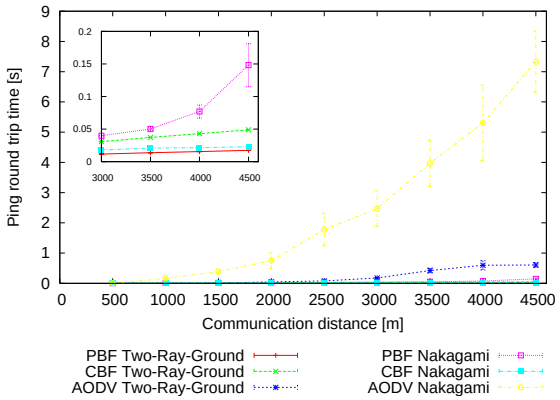


Figure 4.18: Round trip time of PBF, CBF and AODV when increasing the distance to destination for Two-Ray-Ground and Nakagami

Impact of the Reactive Location Service Motivated by the existence of geo-addressed applications in VANETs, we also compared the performance of the position-based schemes assuming that the application relaying on the routing layer already knows the position of the targeted node, or area, at the moment it generates the first Ping packet. The results obtained presented the same overall behavior already seen. In more detail, RLS had a negligible impact on packet delivery ratio and resulted in a slight increase in terms of load (up to 130 $kByte$ for PBF and 80 $kByte$ for CBF) and round trip time (up to 12 ms for PBF and 4 ms for CBF).

4.3.4 From Unicast to 'whatever-cast'

Recent developments show that unicast connections are not really a likely scenario for first-generation VANETs. Thus, we have searched for applications of these results from which even a first-generation VANET could benefit. This, we outline as follows:

Every useful (multi-hop) communication scenario on a highway has to conserve the radio channel to allow for concurrent transmissions. Thus, the ultimate question is to find protocols that are able to spread a piece of information with minimal channel utilization. Again, due to the geometric situation on a highway, we can expect any transmission to cover a slice of the highway in both directions from the sender, with a high probability of being heard by everybody within this slice. Thus, from a purely geometrical point of view, *Contention-Based Forwarding* guarantees a minimum of transmission redundancy and a maximum speed of propagation, since the next forwarder is the one that is the farthest away from the current hop. Consequently, CBF is an efficient protocol element for any kind of addressing

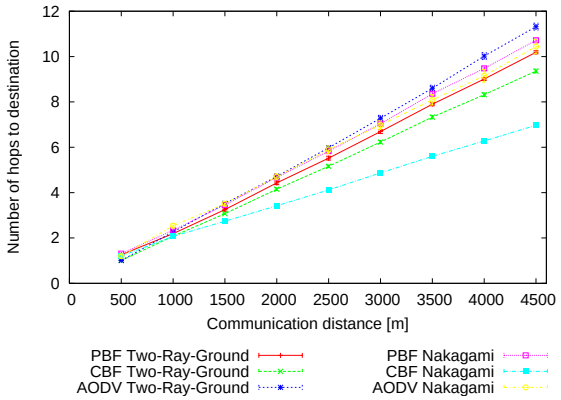


Figure 4.19: Number of hops to reach destination of PBF, CBF and AODV when increasing the distance for Two-Ray-Ground and Nakagami

scheme on a highway,[6] especially those that will most likely be part of even a first-generation VANET platform, because most of these scenarios include the addressing of geographic regions, fitting perfectly into a contention-based geographic forwarding scheme.

4.4 VANET Packet Fowarding in Cities

Having discussed highway packet forwarding in the last section, communication between cars driving in a city environment creates different challenges, largely due to the more complex geometry of the scenario. The challenges can be summarized as follows:

Geometric Two-Dimensionality In a city scenario, vehicles change their movement direction all the time. Moreover, cars can move at any relative angle to each other allowed by the street geometry. In contrast to the highway, this weakens the correlation of the destination position to a suitable next hop.

Obstacles A city is usually characterized by the presence of radio obstacles, considering 2.4 or 5 GHz radio transmissions with power settings similar to WLANs on non-elevated radio antennas. Also, this creates problems with position-based next hop selection [170]. For all of the modeling in these scenarios, usually the simple assumption is made that whenever the line-of-sight between two nodes goes through an obstacle, the nodes are not able to com-

[6] The protocol would have to be slightly modified since it is engineered for unicast, resulting in a situation in which the messages to spread only in one direction.

municate. Note that multi-path propagation and complex obstacle surfaces create a much more complicated situation in reality.

Node Density In cities in industrialized countries, the node density can be expected to be rather high with respect to the radio range, especially at "density hot spots" like junctions. On one hand, node density creates better ad-hoc connectivity, but on the other hand, it poses a challenge to flooding mechanisms that need to be very efficient [231].

Low Mobility Compared to highway scenarios, nodes move at lower speeds, influenced by node density (the more nodes, the slower the movement) and are constrained by speed limits. Also, the mobility is location-dependent, e.g., it is lower at junctions.

When designing a communication system — especially for *Ad-Hoc Networks* — a very important issue is the user communication the system has to support. Application developers ask the opposite question: What will your communication system offer to us? This is a chicken-egg problem since both goals cannot easily be optimized globally.

In the Internet, this problem is solved by transport protocols like `TCP` [275], which try to use the complete available bandwidth and at the same time to be fair to other data streams. With this preposition, a communication system only has to support the capability to send data packets to arbitrary nodes, regardless whether they are direct neighbors or are only reachable via multiple hops.

The decisive question in *Ad-Hoc Networks* is not really the bandwidth, but the hop-distance between communicating nodes. [144, 143] state that the bandwidth in an *Ad-Hoc Network* is, even under optimal circumstances, $\Theta\left(\frac{W}{\sqrt{n}}\right)$, where W is the link bandwidth and n is the number of nodes. While this is only an asymptotic statement, [203] shows by simulation how network performance diminishes with a growing number of hops. In a vehicular context, [149⋆] shows the same for cars running `TCP`.

While the capacity constraint will force ad-hoc communication to be local, `TCP` problems (e.g., [121, 63]) and mobility [215, 216] still add more weight to this necessity. Also, high-bandwidth communication can impair low-bandwidth communication for other `VANET` purposes. From all this, we derive that unicast communication will only be possible for a low number of hops. Internet-style applications will most likely be using infrastructure-based networks.

4.4.1 Creating Realistic City Movements

City traffic simulation itself is a complex challenge because the traffic flow in conurbation deeply depends on rules at its intersections, and on the capacity of the roads

and intersections. The traffic flow simulator Videlio [183], developed by Daimler-Chrysler AG, extends FARSI, and uses time dependent origin-destination matrices. Core elements are the Optimal Velocity Model [64], a special lane changing model [183], and the C-logit model [310], to calculate the vehicular movements. Videlio uses a detailed description of the road network, with information about, e.g., lane numbers, traffic regulations, and time tables of the traffic lights.

To generate the vehicular movement pattern, a small part (6.25 km × 3.45 km) of the city of Berlin was modeled as a graph of streets, with 28 vertexes and 67 edges as depicted in 4.20. In total, the movement of 955 vehicles was simulated.

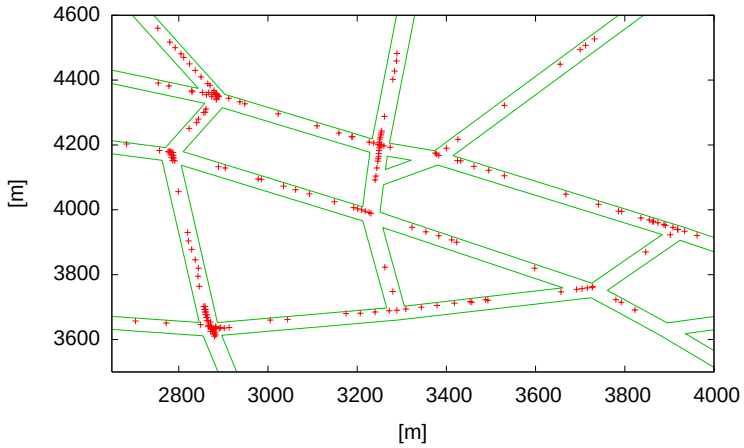


Figure 4.20: Snapshot of city movement data

4.4.2 Forwarding in City Scenarios

Candidate Protocols

In the following, we will roughly describe candidate protocols for VANET routing in a city scenario. Since the utilization of position information is easily possible in VANETs, and beneficial for routing, all protocols considered (except AODV in 4.4.2) make extensive use of the positions of the communicating nodes. While the sender's position is obtained from the local GPS/navigation system, a so-called location service like RLS [193*] provides the position of the destination. Table 4.2 shows an overview of the protocols described in the this section. The table consist of the following rows:

Map Usage indicates if the protocol needs a map or can use one optionally.

	STBR	GSR	GPSR	GPCR	AODV	PBR-DV
Map Usage:	mandatory	mandatory	—	optional	—	—
Forwarding based on:	Position	Position	Position	Position	Topology	Position
Greedy Recovery:	—	—	Planar Graph Routing	Planar Graph Routing	—	Distance Vector
Recovery requires Flooding:	—-	No	No	No	—-	Yes
Variable Header Size:	—-	junction source route	—	—	—	—
Proactive Overhead:	Junction-to-Junction Beacons	Neighborhood Beacons	Neighborhood Beacons	Neighborhood Beacons	—	Neighborhood Beacons
CBF compatibility:	Yes	Yes	No	No	Yes (`CBDV`)	Yes (`CBDV`)

Table 4.2: Protocol comparison

Forwarding Based on There are two basic forwarding strategies: One is based on the position of the destination, and the other one is based on the network topology.

Greedy Recovery Remember: Position-based routing approaches use a so-called greedy routing mode. The neighbor nearest to the position of the destination is selected as the next hop. If there is no suitable neighbor, a recovery mode is started to find a non-greedy route. In this row, you can see the recovery strategy used.

Recovery Requires Flooding This row indicates whether or not the recovery mode of a protocol requires a flooding process.

Variable Header Size Routing protocols add routing information to the header of every packet. This row shows if this additional information has a variable length.

Proactive Overhead Some routing protocols continuously send control packets to build neighbor tables. This row indicates if a protocol sends such control packets.

CBF Compatibility This row shows if a protocol can be extended with `CBF` (see section 4.4.2).

PBR and Greedy Perimeter Stateless Routing The name *Position-Based Routing* (PBR) summarizes a class of routing algorithms that forward on the basis of node positions by means of beacon-generated neighbor tables.

If no neighbor is closer to the position of the destination than the receiving node, the packet has reached a local optimum. Various algorithms have been proposed to overcome such situations. These algorithms principally planarize the neighborhood graph and use graph traversal strategies to find the destination or at least a node, where greedy forwarding becomes possible again (see Section 2.5.8).

Street Topology-Based Routing The idea of *Street Topology-Based Routing* (STBR) [123′] is to interpret a given street map as a planar graph. Every junction is interpreted as a graph vertex, and every street between two junctions as an edge. On these edges or links, small junction-to-junction beacons are sent to check the usability of a street to transport data packets. Additionally, there is link-state information kept at the nodes (junctions) about the connectivity to the neighboring junctions.

In principal, the protocol works as follows: on a junction, one node is selected as the master. All other nodes at the junction operate as slaves, and nodes on streets between junctions are used as forwarders. So there are three valid states for a node: master, slave, and street (forwarder). The job of the master node is to check if the links to the next junctions are up or down. Therefore, every master broadcasts beacon packets. These beacons are forwarded by the street nodes to the masters of the neighboring junctions (which are one hop away on the planar graph). The beacons are forwarded using CBF (see section 4.4.2). The status of the links is stored in a so-called junction neighbor table. The first level, i.e., the list of directly neighboring junctions, is sent out by the master nodes with every beacon. This information is used by the receiving masters of the neighboring junctions to build a two-hop neighbor table, and by all other nodes on the streets to build a one-hop neighbor table. In Figure 4.21, we see an example of a two-hop neighbor table.

If the position of the destination node is unknown, the source node uses a location service to acquire the position. After this, a complete routing process from source to destination consists of three parts:

1. Routing from the source node to the first junction,
2. Routing from junction to junction,
3. Routing from the last junction to the destination node.

If the sender is a street node and the destination is not on the same street, the packet is forwarded to the junction closest to the position of the destination. The

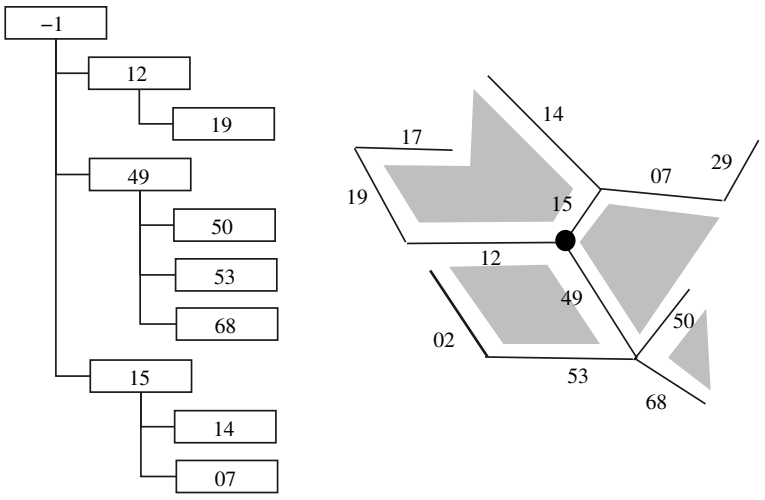


Figure 4.21: Example of an STBR neighbor table (as in [123′])

master node from this junction uses the two-hop neighbor table to find the junction which is reachable and has the most progress to the destination. Then, the master starts the forwarding of the packet to the chosen junction. This second part is repeated until the destination is located on a street which is directly connected to the current junction. Then, the junction master uses the street nodes to forward the packet to the destination node.

There are many special cases the protocol has to take care of, e.g., if there is no junction with progress in the neighbor table. For more details, please refer to [123′]. Also, [233′] describes a similar protocol.

Geographic Source Routing Like STBR, *Geographic Source Routing* (GSR) [80, 210⋆] uses a map and a position-based address scheme to send packets to the destination. As before, the source node uses a location service to acquire the position of the destination node. Now the source node evaluates the shortest path between itself and the destination (Dijkstra [105] or Breadth-First-Search [19]). All junctions on this shortest path are added to the header of the packet, as in DSR [165]. The packet is forwarded from street to junction, from junction to junction and from junction to street in a position-based routing (PBR) fashion. Therefore, every node continuously sends beacons carrying its own position and its node id. With the position information of the beacon, every node can build a one-hop neighbor table. So a receiving node can select the neighbor with the highest progress to the position of the next junction as the next hop. Once the junction is reached, it is deleted from the packet header, and the position of the next one is used as the new destination. After the last junction, the position of the destination node is chosen.

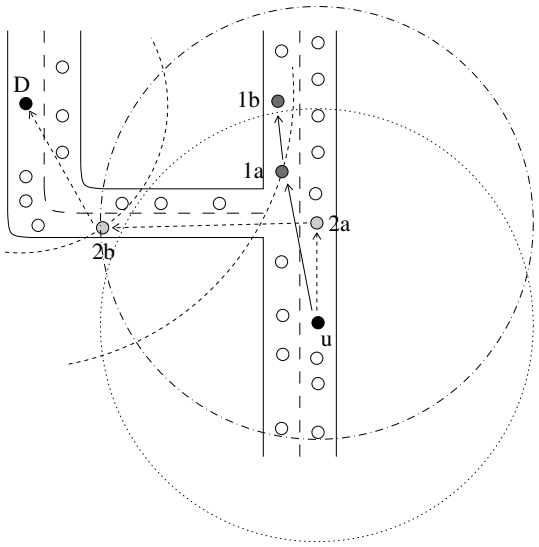


Figure 4.22: Example of GPCR's "Restricted Greedy" (as described in [208′])

Greedy Perimeter Coordinator Routing The basic idea of Greedy Perimeter Coordinator Routing (GPCR) [208′, 209⋆] is to use position-based routing without map knowledge and with a better recovery strategy than the perimeter mode of GPSR. Like in GPSR, every node continuously sends beacon packets with their own position and their node id. Additionally, the beacon includes information about whether the sender is located on a junction or on a street. When a node wants to send a packet, a location service is used to estimate the position of the destination. Similarly to GPSR, packets are sent to the one-hop neighbor closest to the position of the destination. The major difference is that neighbors on junctions are preferred even if their progress to the position of the destination is lesser. This selection strategy is called restricted greedy routing. Figure 4.22 depicts a small example. Node u wants to send a packet to D. The normal greedy mode would select 1a, because it is the neighbor that is located closest to D. Node 1a would select 1b, reaching a local optimum. In restricted greedy mode, u knows that 2a is located on a junction and would prefer this one. 2a has, compared to 1a, the advantage that it has no obstacles between itself and 2b. We can see in this small example that junction nodes have the benefit that they can reach neighbors located on all connected streets.

The right-hand rule only finds a path if the graph is planar. For this reason, GPSR uses a distributed algorithm to build a planar one-hop neighbor table by virtually deleting links. GPCR's use of the right-hand rule is slightly different: In the recovery mode, a junction node decides based on the right-hand rule, to which junction the packet should be forwarded. Between two junctions restricted greedy routing is

used. So the recovery mode uses a graph which is identical to the real-world street map, and thus is planar[7]. As in GPSR's recovery mode, no flooding is necessary because GPCR also only uses existing neighborhood information.

To detect whether or not a node is on a junction, [208′] proposes two strategies. First, all nodes add the positions of their neighbors to the continuously sent beacon packets. A node is located at a junction if two neighbors are in transmission range but do not list each other in their neighbor tables. It is assumed that an obstacle is between these two nodes and thus they are not on the same street. In the second approach, no additional information has to be sent. The node is on a street if all one-hop neighbors are roughly located on a line in the driving direction of the car. Otherwise the node is on a junction. This criterion is checked with a correlation coefficient [26].

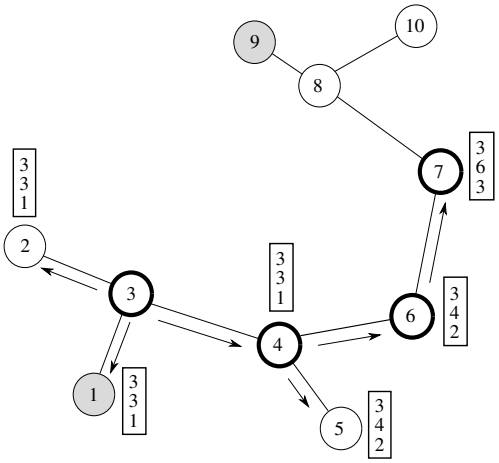
Ad-hoc On-Demand Distance-Vector Routing *Ad-hoc On-Demand Distance-Vector Routing* (AODV) [241] is a reactive, and completely topology-based routing protocol, already described in Section 2.5.3.

Position-Based Routing with Distance-Vector Recovery In this section, we present *Position-Based Routing with Distance Vector-Recovery* (PBR-DV), our proposal to combine position-based greedy routing with AODV-style recovery. This approach uses the well-known position-based greedy routing scheme, which is also used in GPSR (see above). Thus, every node periodically sends beacon packets with their position and node id. If the position of the destination is unknown, the source node uses a location service to acquire it.

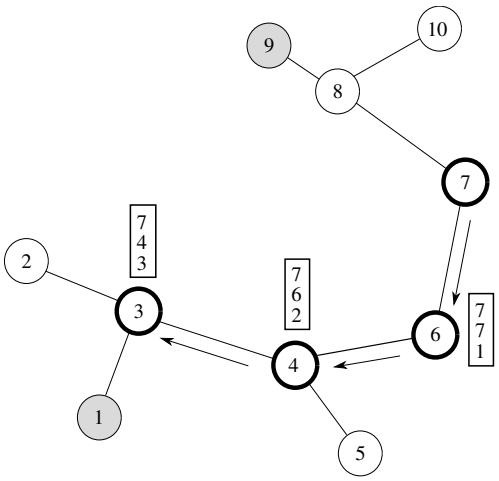
While greedy routing is the default behavior, the novelty starts when a node reaches a local optimum. In that case, PBR-DV changes to a distance vector mode. The task of this recovery mode is to forward the packet to a node which is closer to the position of the destination than the node starting the recovery. This also includes the destination node itself. The route discovery is similar to AODV. Thus, the node at the local optimum broadcasts a route request packet. The route request packets of PBR-DV additionally include the position of the node which started the recovery, and the position of the destination node. If the receiver of the route request is not located closer to the destination than the node which starts the recovery, it rebroadcasts the packet and stores the id of the sender in the routing table. Otherwise, i.e., when it provides distance progress, it sends a route reply packet with its own position to the neighbor which broadcasts the request. The receiver of the reply packet uses the entry in the routing table to forward the

[7] Streets cross each other, e.g., at bridges. In that case, and when there is radio connectivity between the crossing streets, a vertex is defined at each crossing. If there is no radio connectivity, planar routing can fail.

packet to the previous node which broadcast the request, and so on. As the packet travels back to the node which starts the recovery, every forwarding node stores the previous hop in the routing table. In this way, the node which started the recovery can forward the packet to a node which is closer to the destination. On this closer node, the mode is changed back to PBR.



(a) Request Packet



(b) Reply Packet

Figure 4.23: PBR-DV: Path setup in recovery mode

On the left side of Figure 4.23, we see a small route request example. Node 1 wants to send a packet to node 9. Node 1 forwards the packet to node 3 because this is the node with the most progress to the position of node 9. However, node

3 has no neighbor closer to the destination and switches the packet to distance vector mode. Node 3 broadcasts a route request packet to nodes 1, 2 and 4. The receivers store the information in the routing table that they can reach node 3 in one hop, and rebroadcast the request packet, because no node is closer to node 9 than the recovery starting node. Node 6 receives the forwarded request packet, and rebroadcasts it. Finally, the packet reaches node 7, which is closer than node 3 to node 9. In its neighbor table, node 7 stores the information that it can reach node 3 over node 6 in three hops, and it sends a route reply packet to node 3. On the right side of Figure 4.23, we see the path of the route reply packet. After node 3 receives the route reply, it forwards the packet along the acquired path to node 7, which changes the packet back to PBR mode and forwards the packet to node 8. Finally, node 8 forwards the packet to the destination node.

The recovery strategy of PBR-DV—compared to those of GPSR and GPCR—has the disadvantage that an additional flooding is necessary to discover the non-greedy part of the route. This effect can be minimized by flooding the request with a small hop limit.

Contention-Based Forwarding *Contention-Based Forwarding* [134⋆] is the main contribution of this thesis and has been extensively described in Section 3.3 in a general MANET context and in Section 4.3.3 in a VANET/highway context.

In its purest form, CBF was only usable for greedy routes based on position. CBDV [202, 200′], or *Contention-Based Distance-Vector Routing*, however is the contention-based cousin[8] of PBR-DV and can be applied to both 1D and 2D routing.

While CBF simulation results indicate it's supremacy to explicit next hop selection algorithms, the basic problem for practical CBF application is that its more difficult to operate with standard 802.11 MAC hardware and operating system user space implementations due to its sensitivity to timer granularity.

While simulation results (Sections 3.3 and 4.3.3) indicate performance supremacy of CBF over standard position-based methods with explicit selection, we have already argued in Section 4.3.3 why we nevertheless stick to the inferior protocols. Since the same arguments hold for city scenarios, we will stick to explicit-select protocols for the remainder of this section. However, since with CBDV (Section 3.4), we already have a recovery companion to CBF, this couple could also be applied to a city context.

However, most protocols we discuss could be modified to support CBF operation. In detail, while the distance-vector can be modified to be contention-capable as described above, GSR could incorporate CBF as a junction-to-junction forwarding,

[8] Standard DV routing has to know about a node's potential forwarders/neighbors to select them explicitly. CBDV operates truly contention-based without explicit forwarder selection, by greedily forwarding on the hop-count property.

method as in STBR. Contrarily, this is quite different with GPSR and GPSR, due to their being based on distributed graph planarization. This planarization works on the respective neighbor tables of the involved nodes by removing virtual links such that the resulting network graph is planar. CBF, however, does not have neighbor information, due to the lack of beaconing. Therefore, the planarization is not applicable making GPSR and GPCR less compatible for a forwarding method based on contention.

4.4.3 Suitability Analysis

In this section, we analyze the suitability of the presented routing approaches in a city scenario. Therefore, we classify the approaches into the ones using maps, methods based on distance vectors, and finally algorithms that only use node positions.

Map-Based Approaches

The first group consists of the routing approaches GSR and STBR. Both approaches assume that the communication system has access to the digital representation of a street map, e.g., as provided by a navigation system. In theory, the additional map knowledge can improve the routing capabilities, since the node positions, and therefore the network structure, are highly correlated with the streets. Nevertheless, following GSR's and STBR's description, we have to keep in mind that this improvement is bought dearly, with a more complex architecture of the communication system. Furthermore, the system is required to have up-to-date information on the street situation, moreover, in the algorithms discussed, all maps used have to be the same.[9]

The path calculation in GSR assumes that all links (streets) are up. A link is called up if there are enough cars to forward the packets from the start junction to the end junction of the street. If this is not the case, GSR switches to the global position-based routing mode and forwards the packet directly to the position of the destination node. In this mode, the receiving node selects the neighbor with the highest distance progress to the position of the destination as the next hop. If there is no neighbor node closer to the destination than the current node, the packet is in a local optimum, and will be discarded. For this reason, GSR is most suitable for scenarios with many nodes resulting in many "up links", i.e., streets with a continuous node distribution, being able to forward packets from one junction to the next. However, especially in sparse scenarios, it is very unlikely that most links

[9] I.e., when packet headers use street or junction IDs for addressing, these IDs and their structure have to be globally agreed upon. This practically means that most of navigation maps used today could not be used since they have different map data sets or versions.

are up. Altogether, GSR is a comparatively simple routing approach, which does not exploit all the potentials of the map usage and is largely ignorant of the real connectivity situation.

STBR uses the map in a more complex manner and considers whether a link (street) is up or down. To measure the connectivity, STBR sends beacons within the first level of the junction neighbor table from junction to junction. Every street node is used as a possible forwarder for the beacons. If an application needs the local neighborhood information, PBR can be used instead of CBF to forward the beacons from junction to junction. In this case, there are both junction beacons and beacons used for PBR forwarding. These packets create continuous traffic to build all necessary neighbor tables, even if there are no data packets. In the scenario we have in mind, unicast traffic is only one communication type. Its importance is minor compared to active safety applications using geocast, and even if CBF is used, the continuous multi-hop control traffic might conflict with active safety data, which is consequently a disadvantage of this approach. Another challenge is that without modification, STBR would try to send junction beacons along a highway because it is not suitable for mixed scenarios. Furthermore, the high number of special cases the protocol has to handle, e.g., the selection of the junction master, or the transfer of the two-hop neighbor table to the new master when the old one leaves the junction, and so on, increases the complexity of the protocol state machine. Altogether, STBR is a heavy-weight protocol most likely providing an advantage in city scenarios with the need of long-distance (in terms of hops) unicast communication, at least spanning multiple junction traversals.

Distance Vector-Based Approaches

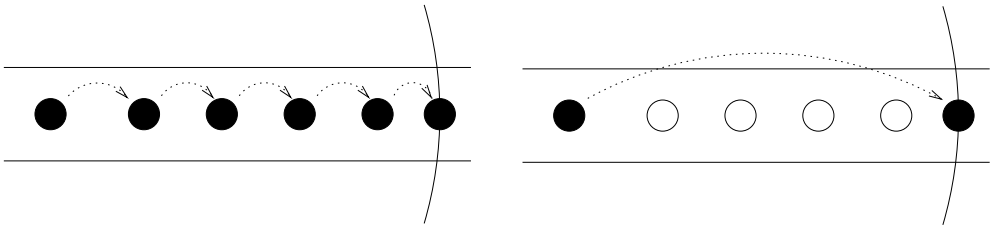
AODV is the only distance vector-based approach in this thesis. In [208′, 210⋆, 123′, 262⋆, 263⋆], AODV is compared to other routing approaches in vehicular scenarios by means of simulation. In most of these simulations, AODV performs well for communication over a few hops in the city case. However, a disadvantage of AODV is that it uses explicit routes. If a route is disrupted, a new route request will be flooded. PBR-based approaches decide the next hop only with the knowledge of the local neighborhood. Therefore, they can respond better to changes in the environment like moving cars, but only if the next hop is greedy. This effect gains gravity with increasing mobility, causing any topology-based approach to behave poorly.

An important VANET use case is likely to be (multi-hop) communication to a static hot spot. In such a scenario the position of the hot spot is known or it is only necessary to acquire it once. If the position is known, position-based approaches can use this position directly and do not even need a location service. AODV is not position-based, and has to establish an explicit route every time. Thus, AODV is only suitable for few-hop communication and non-static nodes.

PBR-Based Approaches

The next group consists of GPSR, GPCR, and PBR-DV. All of them are based on PBR and use a special recovery strategy. GPCR also slightly modifies greedy forwarding by preferring cars having radio access to multiple streets. However, GPCR only achieves low junction detection rate (see below), lessening the effect of this modification. Consequently, we stick to the differences in recovery strategies to analyze the projected protocol performance.

In the recovery mode, **GPSR** use the right-hand rule to find a path. However, this rule only works if the graph is planar. Therefore, GPSR uses a distributed algorithm to build a planar one-hop neighbor table at every node. This algorithm deletes intersecting links. A drawback of this method is that the deleted links are mostly the longer ones. On the left side of Figure 4.24, we see a typical routing process in perimeter mode, and on the right side, we see the same situation in greedy mode. So the average progress per hop is much lower than in greedy mode. This leads to increased delays, and to hop counts that may exceed the threshold.



(a) Perimeter Mode Routing (b) Greedy Routing

Figure 4.24: GPSR: Progress per hop (from [210⋆])

There are two common planarization methods, Gabriel Graph (GG) and Relative Neighborhood Graph (RNG). Both assume that the connectivity between nodes only depends on the node distances. If there are obstacles like buildings which prevent the communication between nodes, it is possible that the planarization algorithm deletes non intersecting links, virtually creating an unconnected graph. Figure 4.25 depicts such a situation.

Following the planarization rules, node u deletes the link to node v from its neighbor table because there is another node in the gray-shaded region (node w). This is due to the assumption that all nodes inside this area are able to communicate with each other. However, removing $\overline{uv}$ disconnects the graph, because v is no longer reachable since an obstacle blocks the assumed link $\overline{wv}$.

Another problem of GPSR is that mobility can induce routing loops for packets being routed in perimeter mode. For that reason we can summarize that the perimeter mode is not suitable for city scenarios.

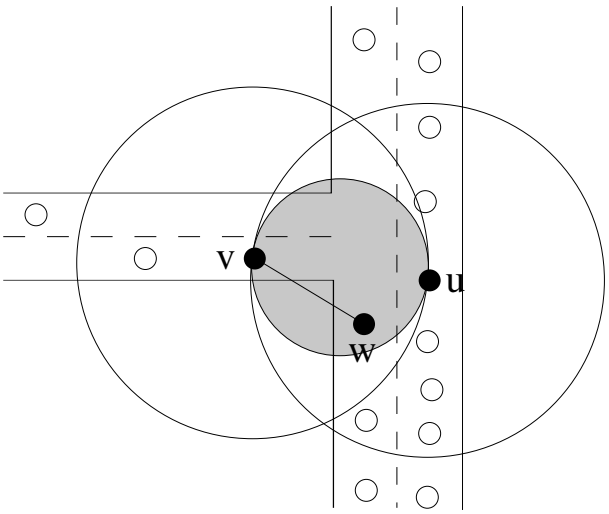


Figure 4.25: Planarization problems (as in [210⋆])

The recovery mode of **GPCR** requires a reliable junction detection method. [208′] has evaluated the combination of both junction detection approaches described in Section 4.4.2. There, a node is only assumed to be on a junction if both checks agree on this result. The authors perform a simulation study in a city scenario to analyze how well this combination works. One problem of GPCR is that only 55% of all junction checks correctly detect an existing junction. The remaining 45% lead to incorrect results, where the study differentiates two kinds of errors. a) A node does not recognize that it is on a junction. b) A node is on a street, but considers itself at a junction. 33% of the incorrect results are of the first error type and 12% of the second. Without better junction detection, the recovery mode and the right-hand rule used do not work properly. Both parts of the junction detection use neighborhood information. These methods work well if there are many neighbors, but fail in sparse scenarios. This problem could be alleviated by using a map to detect junctions. However, even with perfect detection, the right-hand rule performs badly in many situations.

In Figure 4.26, we see a bidirectional routing flow. Node u sends a packet to node v, which sends a packet back to node u. The complete route to node v is greedy, and works without a recovery mode. However, the route back to node u is only greedy until node w, where GPCR will switch to the recovery mode. The right-hand rule prefers the right-hand direction to the destination, and is a good choice if a short route lies to the right. The figure, however, depicts the path a packet travels if the right-hand direction is not available. This creates a very long route, with the potential consequence of a packet drop induced by the hop limit.

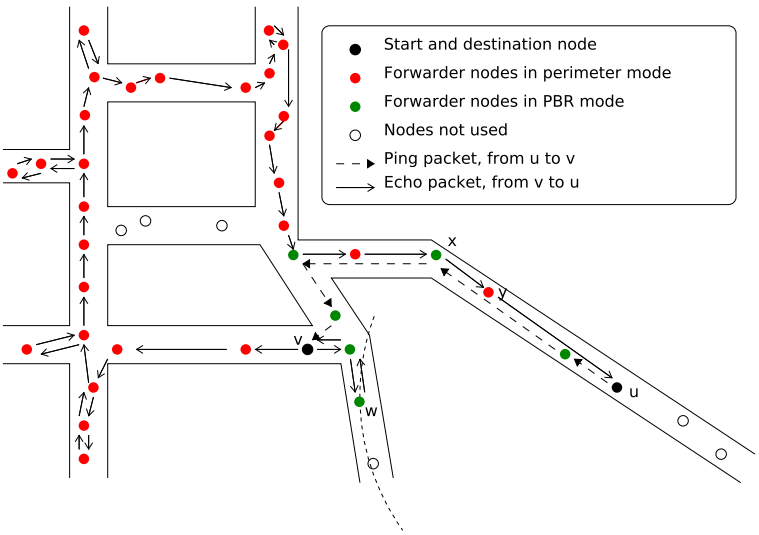


Figure 4.26: Problem of the right-hand rule (from [208′])

[125⋆, 175⋆] indicate that PBR performs very well for routing on highways, whereas AODV suffers from link-ruptures due to the high mobility. Furthermore, PBR works well for communication over few hops in city scenarios since recovery is not needed very often. Thus, PBR is a good choice as a basic routing scheme[10]. In recovery mode, both GPSR and GPCR use a variant of the right-hand rule to route packets. Due to the reasoning outlined above, these methods are not suitable for city scenarios. The basic idea of PBR-DV is to combine a position-based greedy heuristic with a general-purpose recovery mode based on distance vectors. This method is highly suitable because it finds non-greedy routes and performs well for small-hop-count communication in a city scenario (as shown with the example of AODV in [210⋆]). Moreover, it is practically indifferent to radio obstacles, and distance vector-based approaches are well-researched and understood.

In the situation shown in Figure 4.26, node v did not find a usable — since reasonably short — route to node u by the application of the right-hand rule. The packet is greedily forwarded to node w, stalling in a local optimum. The distance vector-based recovery mode of PBR-DV would then start a route discovery process, resulting in a topology-based route to node x. At node x, PBR-DV changes back to greedy forwarding because node x is closer than node w to the position of node u.

Altogether, PBR-DV is the combination of two well-researched and understood routing approaches. Since both methods are hybridly applied in the scenarios in which they perform best, i.e., greedy when possible and DV when not, we are con-

[10]Plain PBR would benefit from the “restricted greedy” mode proposed in [210⋆].

vinced that it is highly suitable for the given task. With this, it combines the strengths of a general-purpose routing scheme like AODV with the power of geographic forwarding. Geographic unicast routing to a node or a hot spot located on the same street comes at minimal cost.

4.4.4 Simulative Evaluation of GPCR

While we did not pursue the goal of quantitatively evaluating the protocols discussed above, we did so for GPCR using the DaimlerChrysler city movement scenarios.

We simulated the performance of GPCR with the ns-2 simulator version ns-2.1b9a. For the simulations we used a real city topology which is a part of Berlin, Germany. The scenario consists of 955 cars (nodes) on 33 streets in an area of 6.25 km × 3.45 km. The movement of the nodes was generated with a dedicated vehicular traffic simulator and represents a real world movement pattern for this given scenario [210⋆]. IEEE 802.11 was used as MAC with a transmission rate of 2 $\frac{MBit}{s}$. The transmission range was set to 500. Real world tests with cars have shown this to be a reasonable value when using external antennas. For each simulation run we randomly selected ten sender-receiver pairs. Each pair exchanges 20 packets over 5 seconds. We measured the achieved packet delivery rate (Fig. 4.27) versus the distance between the two communication partners and the number of hops (Fig. 4.28).

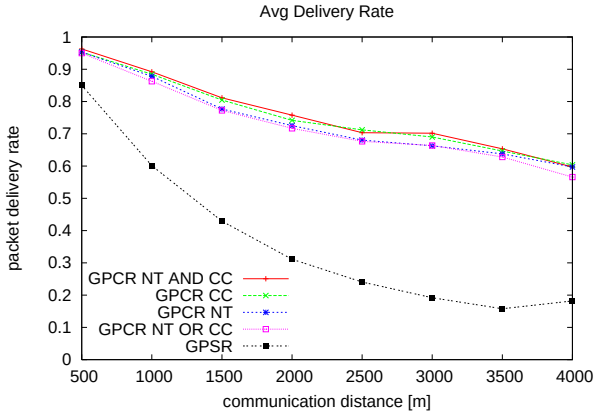


Figure 4.27: GPCR vs. GPSR. – Delivery rate

The communication distance between two nodes is calculated as the minimal distance based on the street topology at the beginning of the communication. Each point in the graphs is based on 10 independent simulation runs.

Fig. 4.27 also depicts how the delivery rate is influenced by the algorithms used for junction detection. It shows that calculating the correlation (CC) coefficient

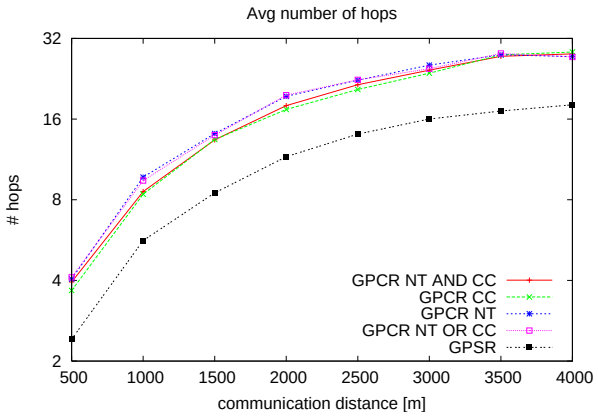


Figure 4.28: GPCR vs. GPSR. – Average number of hops

performs slightly better than relying on the comparison of the neighbor tables of the neighbors (NT). We also analyzed a compound decision consisting of the neighbor table comparison and correlation coefficient, concatenated by logical OR as well as by logical AND. The latter one outperforms the other approaches slightly but it does not come for free: the size of the beacon packets increases for each of the two approaches. Therefore, GPCR simply uses the correlation coefficient. In general, the study on achievable packet delivery rate (Fig. 4.27) shows good results for our approach compared to GPSR. This improvement in performance comes at the expense of a higher average number of hops and a slight increase in latency. This increase in hop counts and latency is mainly caused by those packets that could not be delivered at all by GPSR and thus did not impact the hop-count and latency for GPSR.

Additional simulations for some of the protocols can be found in [128⋆, 209⋆, 208′, 210⋆, 123′, 233′].

4.5 Conclusions and Perspectives

In this chapter, we have presented a collection of MANET routing methods applicable to vehicular scenarios, separated into highway and city situations. Furthermore, we have analyzed them with regard to their expected performance properties, and deduced their suitability for these scenarios. For the highway case, we have shown that any routing method should exploit position information for packet forwarding, having been the first to adapt greedy position-based routing for vehicular scenarios. Moreover, we have enhanced position-based routing with beacons by the application

of position estimation via dead reckoning, significantly lowering the number of link-layer transmissions/bytes transmitted.

Moving from explicit next hop selection to contention-based forwarding, we have created an interesting alternative with even lower transmission cost and lower delay. Also, we have deduced the convergence of addressing schemes for highway scenarios.

In city scenarios, we have contributed by studying well-known methods like AODV and GPSR, analyzing problematic scenarios and complementing these protocols with elements to alleviate them. Especially with GSR and GPCR and the street-to-link abstraction protocols, we provide protocols suitable for different communication platforms, e.g., for platforms with or without a digital map.

However, for early-generation VANETs, we stick to proposing PBR-DV, mainly for the following reasons.

1. PBR is the protocol-of-choice of the FleetNet [8] project, and has been shown to work very well for small-hop-count scenarios, since recovery is not needed very often. However, there can be — even static — cases, where PBR will not find any route at all, making it unacceptable as a stand-alone solution.

2. PBR performs very well on highways, excelled only by *Contention-Based Forwarding*.

3. AODV shows reasonable performance in city scenarios with their low mobility. It works practically indifferently to radio obstacles, and does not require any map information.

4. DV routing is well-researched and understood for *Mobile Ad-Hoc Networks*. Also, it can be used in a purely ad-hoc fashion.

5. While DV routing does not exploit the geometric situation in the city, it also does not require a map and is of fairly low complexity, as opposed to protocols like STBR, which we expect to take over the performance lead for a growing number of hops.

6. While GPCR does not need extra flooding in the recovery case as opposed to DV, it has the problem of finding the right direction to bypass the void.

While the implementation of PBR-DV has just recently been completed [177′] both for use in the real-world and for simulation, we expect the optimization of that protocol to be a straight-forward but time-consuming engineering endeavor. In the context of this procedure, the following design decisions could be answered under the precise assumptions the scenario dictates:

- Should recovery only go round the void, or should it go from source to destination (or from void to destination)?

- Should the restricted greedy variant of GPCR go into PBR?
- What are reasonable values for soft-state parameters, like time outs or beacon intervals?

Another significant advantage of PBR-DV is that it is replaceable with CBDV [202], creating the possibility to use the performance-promising contention-based forwarding methods.

For the future of unicast routing in vehicular scenarios, we expect hybrid wired/wireless protocols to gain significance. [207] shows that including some infrastructure helps to overcome the theoretical boundaries existing for multi-hop *Ad-Hoc Networks*. While this has already been looked into in [82], it will be interesting to see if it can be used for vehicle-to-vehicle communication.

However vehicle-to-vehicle communication may prosper, we expect safety to be the motor for VANETs. Thus, the most important contributions we might have offered are the awareness of unicast's limits, the understanding of vehicular movements and their impact on connectivity, and the multitude of protocols designed for unicast, but also usable as efficient methods for different addressing schemes, their flagship being *Contention-Based Forwarding* with its opportunistic forwarding. In a limited hop-scope and together with geo-cast addressing, this scheme will also be promising for cities, because the unicast graph problems do not apply when the originating node is also part of the addressed area.

Chapter 5

Implementing VANETs in the Real World

> It doesn't matter how beautiful your theory is, it doesn't matter how smart you are.
> If it doesn't agree with (the) experiment, it's wrong.
>
> *(Richard P. Feynman)*
>
> Programming is understanding.
>
> *(Kristen Nygaard)*

Chapter Outline

Most of the previous work in this thesis has been purely theoretical in the sense that no real packets were transmitted. However, the industry-related nature of our work demanded for a real-world implementation and evaluation, which we summarize in this chapter. The main bundle of real-world activities evolves around a sub-project of *FleetNet* called the *FleetNet Demonstrator*, whose goal is to prove the concept of `VANET` communication. The major aspects of the demonstrator are covered in Section 5.1, as are some evaluation results gathered during our work with the system. Section 5.2 concludes this chapter with a discussion of lessons learned and perspectives evolving from them. As before, most of the work presented here has been published in smaller batches. In reality, the amount of documentation for the system is significantly higher.

As in previous chapters, refer to the original papers for more details, i.d., for the Demonstrator [149*, 228*, 129*] and for future systems / lessons learned [132*, 266*, 265*, 130*].

5.1 The FleetNet Demonstrator

5.1.1 Introduction

In the framework of the *FleetNet* project, a demonstrator system was to be built with the capability of demonstrating selected research achievements. To accomplish this, a task force was built which was centered around NEC Network Labs and the University of Mannheim to provide the software part of the communication system, and around DaimlerChrysler to integrate the communication system into a 'fleet' of vehicles and to provide basic demonstrable applications. Overall, the goal was to build a vehicle-carried multi-hop communication system capable of exploiting position information and communicating by means of off-the-shelf 802.11a/b/g hardware.

In this chapter, we describe the efforts made and results achieved in the process of building this system. Section 5.1.2 describes the demonstrator system from a bird's eye view in terms of both hardware and software engineering perspectives. The following section 5.1.3 deals with the protocol deployed in the demonstrator. After initial testing and debugging, we performed measurements, which are described in Section 5.1.4.

5.1.2 The Demonstrator System

System Overview

The *FleetNet* demonstrator consists of at least 6 *Smart*™ cars [51][1], each car representing an IPv4, subnet connected to the others through a wireless *Mobile Ad-Hoc Network*. Each vehicle is equipped with a Windows-based application PC and a Linux-based router. The router is connected to the application machine via Ethernet, while the connectivity to the routers of other nodes is provided by IEEE 802.11b. External antennas are used that have a gain of 4 *dBi* [4] (for a more illustrative description see [271]). Position-based greedy forwarding is implemented as the routing protocol (see Section 2.5.8). Since there is no immediate possibility to include position information into the IPv4 header, a layer 2.5 architecture was chosen: All data needed for routing are stored in the *FleetNet* routing header, which is located between the MAC and IP headers.

A position-based routing approach with beacons requires that each node has information about its own position and about the position of its single-hop neighbors, as well as about the position of the destination node. To get its own position, the router is connected to the on-board navigation system (with integrated GPS). This position information is distributed among other nodes in radio range through

[1] Actually, there were up to ten cars, but 6 of them were stationed in Ulm and the rest in Berlin and only the Ulm-based cars were always available.

beacon messages sent out periodically. Furthermore, all data packets include the position information of the sender in order to provide piggybacked beaconing. Each node distributes its own position at least once per *beacon interval* or BINT. Nodes receiving a packet store the sender/position-pair in a neighbor table. The position information on a certain neighbor remains valid for a period of time called the *beacon expiry interval* of BEXP. If no beaconing information is received for one BEXP, the concerned node is removed from the neighbor table. In the greedy forwarding approach, the router determines as the next hop the neighbor that is closest to the destination. The destination's position is provided by a location service; in our case, we make use of the *Reactive Location Service* (RLS) ([193*], Section 3.1), which is based on a simple flooding mechanism, to determine the position of a requested node. Location information can also be extracted from the *FleetNet* header of received data packets. Location information remains valid for *location expiry interval* or LEXP.

Hardware Perspective Each car has at least one computer running Linux that is dedicated to routing. In the *FleetNet Demonstrator*, applications are principally executed on a different system not further described here. For the tests we have run, the computer for routing was a *Dell Latitude 600* notebook with a 600 MHz *Intel Mobile Pentium* processor and 256 $MByte$ of memory.

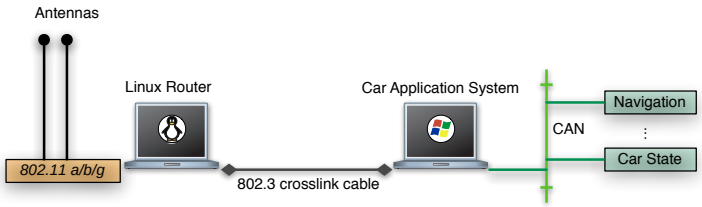


Figure 5.1: Hardware component diagram

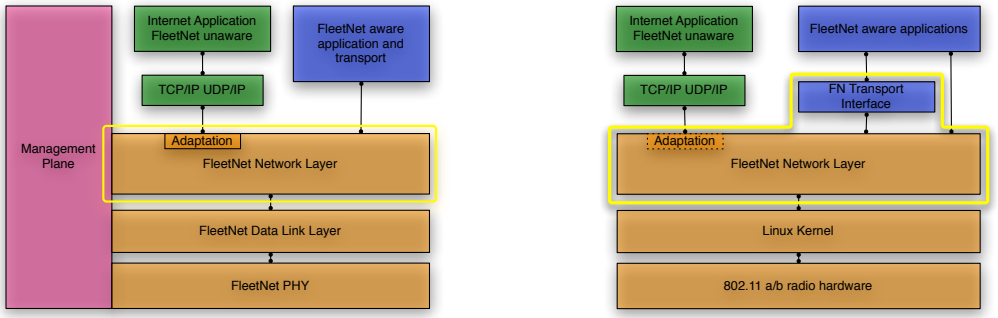
Figure 5.1 depicts how the systems are connected. The in-car network (or *FleetNet Car Area Network* (FCAN)) is based on a standard 100BaseT Ethernet. Since each of the cars is currently equipped with a single (car) application system only, the Ethernet network is a single cross-link Ethernet cable. However, the point-to-point connection could easily be extended to multiple application systems by means of Ethernet hubs or switches. For communication on this network, IPv4 [242] is used. In order to support more than a single application system, most of the interfaces to the routing system are IP-enabled, i.e., they can be accessed from outside the router system. The router uses standard IEEE 802.11 [1] PCMCIA / PC-card [47] hardware for radio communication. The *FleetNet* router software was originally optimized for use with Lucent Orinoco wireless LAN network interface cards and

the Linux driver software for these cards. Some of these optimizations is restricted to `IEEE 802.11b` (2.4 GHz) network interface cards. Consequently, the optimizations cannot be used for advanced `IEEE 802.11a` (5 GHz) network interface cards, which are controlled by a custom non-open-source driver. Nevertheless, the system should work with different drivers if these provide the basic functionality of the Linux wireless tools. For all tests, we have used either the `IEEE 802.11b` or the `IEEE 802.11a` equipment.

To enhance radio connectivity, both radio systems are connected to passive external omni-directional antennas mounted on the roof of the car (one for 2.4 GHz and one for 5 GHz).

The application system was mainly in the domain of DaimlerChrysler and was operated using Microsoft Windows. While we did not 'use' it for communication purposes, it served as an interface to the Controller Area Network (or `CAN` [23]), i.e., the car-internal communication bus connecting all digital car components. Among other things, this is the way to access navigation functions. Thus, the application `PC` was used to provide necessary information to the communication system when deployed in the cars. However, for testing and lab use, we connected a `GPS` directly to the router.

Network Perspective Figure 5.2(a) shows the original proposal for the *FleetNet* network architecture as proposed by the *FleetNet* consortium.



(a) FleetNet network architecture

(b) FleetNet demonstrator architecture (simplified)

Figure 5.2: FleetNet network architecture

This protocol architecture recognizes two types of applications. The first type of applications is called *FleetNet*-unaware (`FNU`), whereas the other one is referred to as *FleetNet*-aware (`FNA`). `FNU` applications are basically standard `IP` applications. Since they should behave as in any other system with Internet access, the *FleetNet*-

specific functionality is hidden from them. The interface between these applications and the routing sub-system is standard IP sockets [276].

FleetNet-aware applications, the second application type, are regarded as more important for inter-vehicle communications. These applications are aware of communicating over an inter-vehicular network using position-based routing for packet forwarding, and are capable of exploiting its special features.

Figure 5.2(b) shows a simplified version of the protocol layers as they are used in the *FleetNet Demonstrator*. The lower protocol layers as depicted in Figure 5.2(a) are realized in the hardware (network interface card) and in the driver software, in the Linux operating system typically implemented as a *kernel module*.[2].

A historical problem of *FleetNet* is the absence of a sub-project working on the transport layer. Nevertheless, transport functionality is required, even if it is only used for application port (de)multiplexing (like in [243]). Therefore, basic transport interface functionality for FNA applications is provided by the routing system[3]. This is indicated by the extension of the yellow box denoting the functionality scope of the *FleetNet Demonstrator* router. FNA applications directly interact with this interface for data transport. An additional interface is provided to access management information such as router state.

To allow for applications to reside on a different computer system than the router, all interfaces above the yellow boundary are implemented using UDP/IP. While the router transparently handles FNU packets, the FNA data forwarding / management interfaces are directly addressed by using UDP on a specific port.

This directly leads to the question of the IP addressing schemes used in the *FleetNet Demonstrator*. As outlined above, only IPv4 [242] is used, and the (yet) private nature of the network implies the usage of a reserved private address space [251]. The 192.168.x.y address partition we chose for the router is subdivided in the following manner: The third byte of the address denotes the car, i.e., all cars have a different x byte. All systems in the same car or — to be more precise — all network interfaces of the in-car IP systems differ in the last byte y. In subnet-mask notation, the *FleetNet Demonstrator* is thus a collection of 192.168.x.0/24 class C subnets [53] that is interconnected by the *FleetNet* network acting as an intelligent link-layer and completely hiding its (potential) multi-hop property. For convenience, the in-car Ethernet interface of the router always has 1 as the y byte. All other systems use increasing y bytes. Since we usually have only a single in-car system, its IP address is 192.168.x.2 (see also Figure 5.1).

[2]For simplicity, the interface of the driver is depicted by only one line, although one might also draw a more complex picture, distinguishing between data forwarding and network management as it is actually used.

[3]Compared to TCP [245, 275], the transport functionality is only rudimentary, lacking support for, e.g., reliable end-to-end transport.

Implementation

The protocol described in Section 5.1.3 is implemented in `ANSI C` for the Linux operating system as a forking user-space software daemon [29]. The implementation does not include any modifications to the Linux kernel. In order to execute the routing daemon, only a few kernel settings, including iptables rules and network interface settings, need to be defined.

The implementation design is illustrated in Figure 5.3: The router is equipped with two interfaces — a wireless interface (`IEEE 802.11`) and a wired interface (Ethernet 100BaseT). The protocol stack in the kernel space consists of the drivers for the interfaces and `TCP/UDP` on top of `IP` layers. The routing daemon is executed in the user space and accesses the kernel space by means of 5 different interfaces marked by numbers in Fig. 5.3. [4] The arrows between the interfaces and the router show whether packets are received from the interface, or sent via the interface, or both.

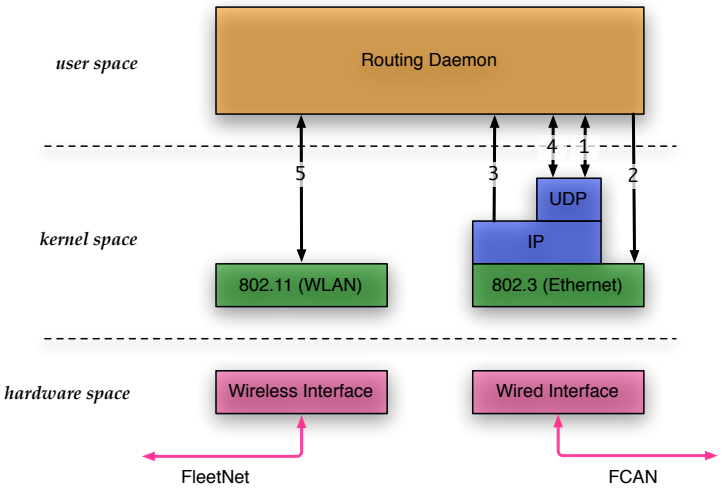


Figure 5.3: Design of the router and its interfaces

In the following, the implementation and purpose of the interfaces are briefly described.

Management interface (1) The management interface is implemented as a `UDP` socket listening to default port 1501. The purpose of the interface is to facil-

[4] For the remainder of this section, the term interfaces does not mean the network interfaces, but the different ways to send and receive packets of the router.

itate the data exchange between applications executed in the car application system and the router, more precisely to exchange location information so as to update internal data structures in the router, as well as in the car application system. The data exchange includes the location updates sent from the car application system to the router, the triggering of a location query by the car application system, as well as the transfer of the *location table* from the router to the car application system on a request/reply basis.

Car interface (2) is the interface used to send packets from the router to the car application system. Note that the router does not receive any packets via this interface. It is implemented as an *IP_RAW* socket, i.e., `IP` packets and their according header are written to the socket and thereby handed over to the bottom of the `IP` stack. The `IP` frame is not modified after being captured by the router of the sending node, except that the `TTL` field is decremented by one. Upon receiving the packet from the car interface, the `IP` stack executes a single task before handing the packet over to the data link, namely the calculation of the header checksum.

***FleetNet*-unaware interface (3)** Unlike the car interface, the *FleetNet*-unaware interface only receives packets. Both the car interface and the *FleetNet*-unaware interface handle the *FleetNet*-unaware traffic with the car application system. All traffic unaware of the layer 2.5 architecture enters the router through the *FleetNet*-unaware interface. which is implemented as an *ipq* file handle that is provided by the *libipq* library[5]. To obtain the packets, the file-handle has to register at the *Netlink* socket created by the *ip_queue* driver. The registration enables the router to receive packets from *iptables* according to previously defined rules.

***FleetNet*-aware interface (4)** handles the traffic of *FleetNet*-aware applications, i.e., applications making use of *FleetNet*-specific capabilities. The interface is realized as a common `UDP` socket listening to port 1500. Applications send `UDP` packets that contain information regarding destination `ID` and payload. The *FleetNet*-aware interface is also used to send FN-aware data from other *FleetNet* nodes to the corresponding car application system attached to the router.

***FleetNet* interface (5)** is the interface to the driver of the wireless network interface card. The interface is used to send and receive *FleetNet* packets only (see Section 5.1.3). The *FleetNet* interface is implemented as a *PF_PACKET* socket, i.e., all packets are directly handed over to the network device, after

[5]The *libipq* library [14] provides a socket-like interface to `IP` packets. With this library, the user can create a filter to extract certain `IP` packets to the user space.

the router assembles an Ethernet header itself. The IEEE 802.3 header is then transformed into the required IEEE 802.11 format by the logical link control. To enable the routing daemon to access all packets on the wireless link at the MAC packet level [276], the socket is run in promiscuous mode.

5.1.3 The Protocol

Principally, the protocol offers a *best-effort datagram transport* between source and destination. The best-effort delivery does not provide any guarantees in terms of packet delivery delay, the order of packet delivery, or packet loss. Reliability is a task assigned to upper protocol layers.

For datagram transport, the protocol offers three basic services, namely *beaconing*, a *location service*, and *forwarding*. While *beaconing* is used to advertise the current location of the node to its neighbors, the *location service* provides means to query the geographic location of a node characterized by its identifier. *Forwarding* is the process used to handle datagrams in a node, including the sending of a datagram to other nodes.

Regarding the datagram transport, the following transport types are defined:

Unicast. Unicast is an undirected data transport service from a single node (source) to a single node (destination) by means of direct communication or by multiple hops based on specific node addresses that include node ID, position, and timing information. Principally, unicast as provided by the IP layer is directly mapped to the unicast provided by the *FleetNet Demonstrator*.

Topologically-Scoped Broadcast (TSB). Topologically-scoped broadcast is a data transport service from a single node (source) to all nodes covered by the *Ad-Hoc Network*. A topologically-scoped broadcast is scoped to limit the number of hops. *Single-hop broadcast* is a special case of topologically-scoped broadcast in which messages are sent to the neighbors only but not forwarded for multi-hop communication. A *single-hop broadcast* is mapped to a broadcast service at the data link control layer.

Geographically-Scoped Broadcast (GSB) In contrast to Topologically-Scoped Broadcast, the addressing scope here is defined by a geometric region containing the position of the sending node.

Geocast. Geocast is a data transport from a single node to a group of nodes within a geographically specified region. If the destination region contains the sending node's position, this forwarding mode simply maps to a geographically-scoped broadcast. If not, a more sophisticated algorithm is needed. A possible approach would be to first reach the region via unicast and then use GSB (as described in [119], DaimlerChrysler has added this to the Demonstrator) .

5.1.4 Experiences and Measurements

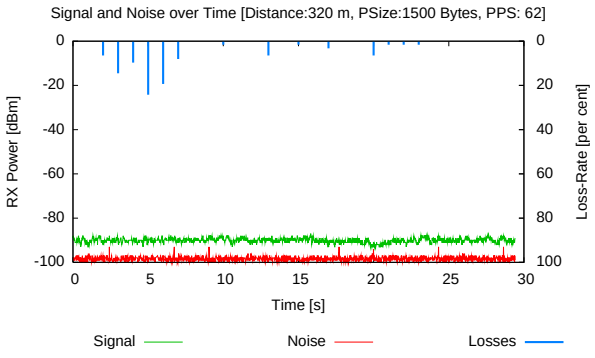
After the implementation and system integration reached a stable state, the *FleetNet* agenda planned for intensive testing and evaluation of the system's performance. The complexity of testing was steadily increased during the field trials, spanning a range from stand-alone router systems [227′] over non-moving cars [149⋆] to a number of cars driving in a row on a circular street course [228⋆]. The remainder of this section describes measurements made in the cars.

Static one-hop measurements

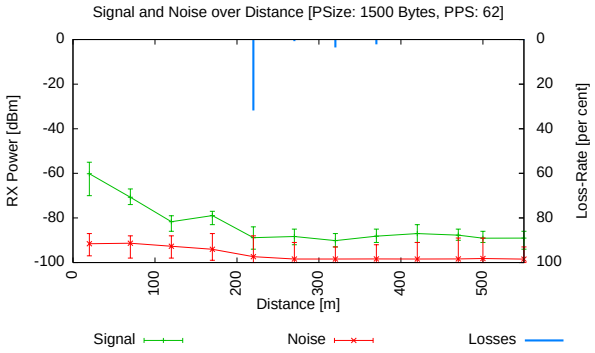
As a first step, we conducted a large number of static one-hop measurements to determine the fluctuation of the received power and loss rate over time, in dependence of the distance between the two cars. The sending car sent `MAC` broadcasts of a predefined packet size (results are shown for a packet size of 1500 bytes) at a rate of 62 packets per second while the receiver stored reception power and noise gathered via functionality provided by the `iwpriv` tool [13]. Clearly, environmental factors like other cars, buildings, and weather conditions affect the results. Figure 5.4(a) shows a typical graph for measuring radio fluctuations and loss rate for a fixed distance (320 m) over 30 seconds. Note that noise and reception power are only available for successfully transmitted packets. The losses were caused by a passing non-*FleetNet* vehicle driving from the receiving car to the sender. The highest losses occurred as the interfering car passed the receiver (seconds 2 to 7), thereby producing a higher noise value. But even afterwards, the car caused some packet losses by reflecting or disturbing the signal. Figure 5.4(b) shows received power and loss rate depending on the distance between the two cars. Even at communication ranges above 500 m no noteworthy loss rate occurred. The only problem was the unstable communication at a distance of 220 m.

Static three-hop scenario

For the static and mobile three-hop measurements, we decided to artificially reduce the 'transmission range' of the nodes by using a suppression mechanism to make the router drop all received packets from senders farther away than a predefined distance. By so doing, the setup of multi-hop communication is facilitated at the cost of reducing spectrum re-use and by affecting corresponding `MAC` mechanisms. In previous tests with a laptop testbed, we became aware of the fact that basic position-based routing protocols depend heavily on link stability. Upon receiving a beacon from a node, the router assumes the link to this node to be active until no beacon or packet is received from that node for a certain period of time. However, we observed frequently that beacons were received over long distances (several hundred meters) but the corresponding link quality was too low to make use of the link for



(a) Single run



(b) Accumulated runs

Figure 5.4: Static one-hop scenario: power and loss-rate measurements

successive packet transfers. Results depend heavily on the antennas used. With the standard antennas built into the WLAN card, we could not obtain results close to the one shown and discussed below in a 'real' outdoor network environment.

For the static three-hop scenario demonstrator, cars were positioned as shown in Figure 5.5 with a distance between two 'successive' nodes of approximately 150 to 200 meters. `UDP`, as well as `TCP` tests were performed.[6] To evaluate `UDP` performance, packets of different sizes were sent from the first node in the chain to the last. Beginning with 50 $\frac{kBit}{s}$ the sending-rate was slowly increased to 500 $\frac{kBit}{s}$. The last node acknowledged packets by sending back a small `ACK` packet. As shown in Figure 5.6(a) the maximum achievable throughput depends on the packet size since a larger number of packets results in a higher probability of collisions, and thus retransmissions or even losses. A throughput of about 400 $\frac{kBit}{s}$ is achievable

[6] While we do not advocate that plain `TCP` should be used in `VANET`s, the measurement of the bandwidth of a `TCP` flow in stable multi-hop radio conditions is still interesting.

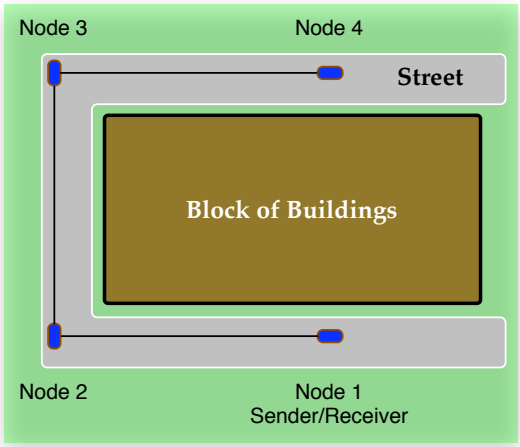


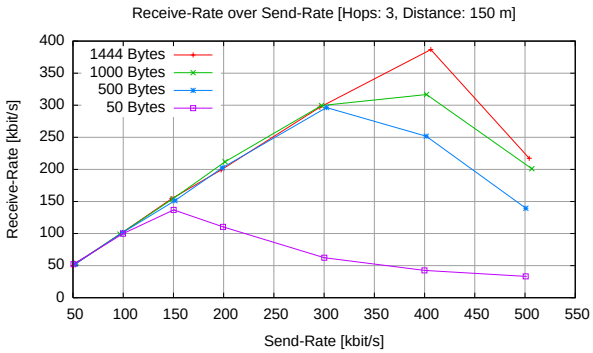
Figure 5.5: Static three-hop scenario: Geographic arrangement

with a packet size of 1444 bytes. The graphs also demonstrate very well the well-known problem of IEEE 802.11, which is that the achievable rate, once it reaches its maximum, does not remain at the maximum but drops afterwards. Each additional packet per second degrades the performance of the network[7] since the delay it causes is worse than the benefit of successfully transmitting the packet.

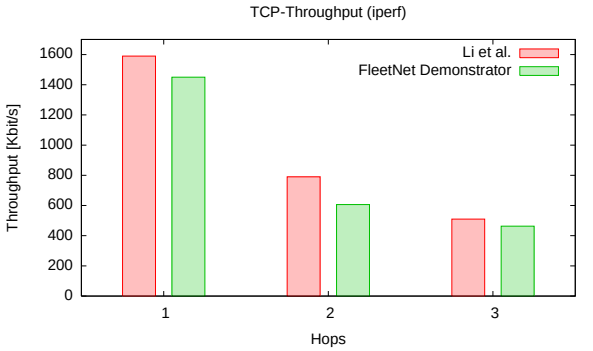
To evaluate TCP performance, *iperf* [12], a widely-used network measurement tool was used. Figure 5.6(b) shows the TCP throughput in relation to the number of hops. For comparison, the results of Li et al. [203], taken by simulation and validated with a laptop testbed, are also plotted in the graph. Our values are close to those of Li et al but a precise comparison is not possible due to the incomparable test parameters like distances between the nodes, hardware equipment or environmental influences. For three hops, we can achieve a TCP throughput of about 450 $\frac{kBit}{s}$.

To properly analyze the effect of the quality of the various links involved we chose a visualization method as depicted in Figure 5.7. Figure 5.7 shows the effects of a bad link on the network performance. The x-axis represents the different measurement points that a packet and the corresponding acknowledgment have to traverse. Having been created by the sending tool, the data packet has to be processed by every router. Once a packet reaches the destination, an acknowledgment of its arrival travels back to the receiving tool. There are two values per measurement point, one for the number of packets received, and one for the number of packets successfully

[7] If nothing else, this is a strong argument for application layer traffic policing.



(a) UDP throughput



(b) TCP throughput

Figure 5.6: Static three-hop Scenario: throughput measurements

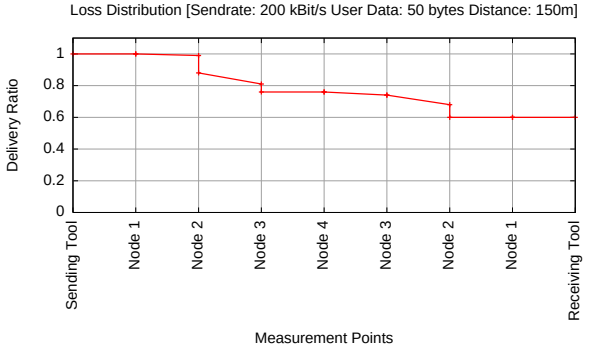


Figure 5.7: Static three-hop scenario: loss distribution

handed over to the network interface. Thus, a vertical line indicates the packet drops inside the router of the corresponding node due to full kernel queues. These drops are typical indications of congestion. Diagonal lines, representing losses between two nodes, stand for link-layer losses. In Figure 5.7, the bad link is between nodes 2 and 3, where about 10% of the data packets get lost on this link, while another 10% are dropped in router number 2, since the kernel queues are full due to the high delay produced by a high number of `802.11` retransmission retries.

Mobile three-hop scenario

Evaluation of the performance of mobile scenarios is a challenging task since reproducibility and comparability are much harder to achieve (if at all possible) than in fully controllable simulation environments or partly controllable static outdoor scenarios. In particular, the challenge is to keep track of all factors that might influence results, e.g., environmental factors that influence radio propagation. In the following, we present our methodology for performance evaluations in mobile scenarios and present some key observations. However, we do not give a 'final' analysis of achievable maximal throughput or delay since the system is still under development.

In our mobile scenario, four cars were driving on a circuit that is about 5 kilometers long; it is depicted in Figure 5.8. The first car sent data packets of a predefined

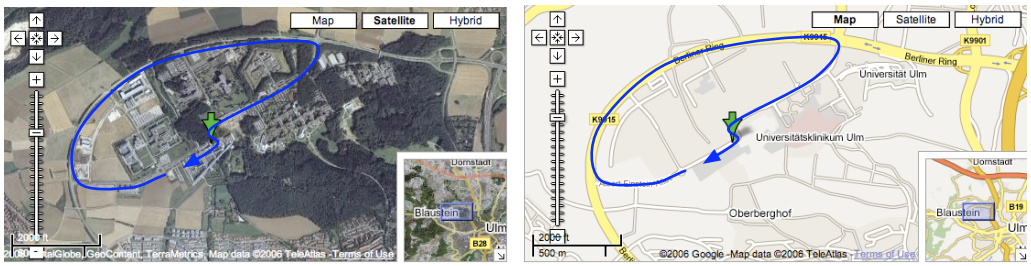
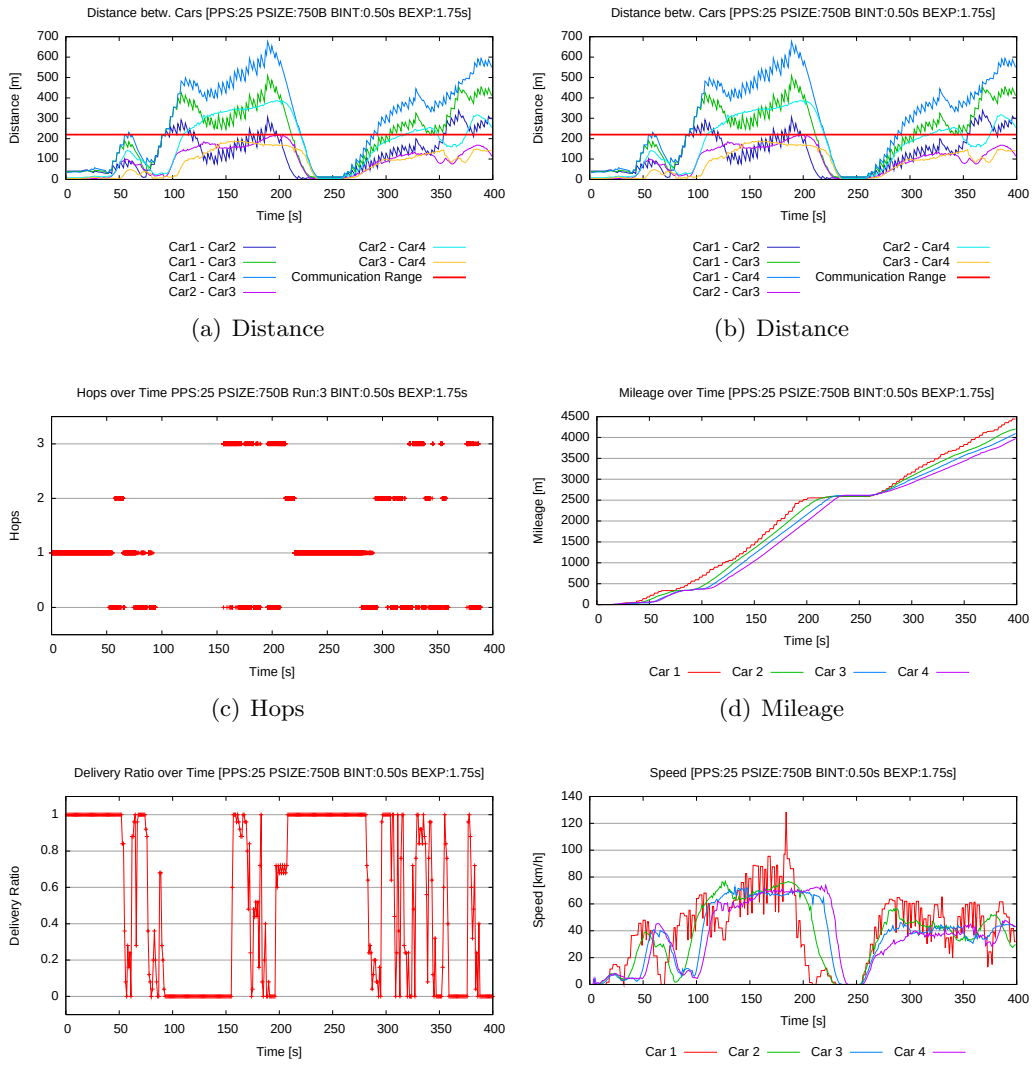


Figure 5.8: Mobile three-hop scenario: map of the test circuit (from [9])

size to the last car at a rate of 25 packets per second. The last car, number 4, acknowledged receipt of the packets by means of small `ACK`s, as already mentioned in the previous sections. At the beginning of the test run, all cars were within communication range of each other, enabling car 1 and car 4 to communicate directly. During the run, we tried to built a three-hop communication chain, but had to respect other cars and their right of way as well as traffic lights. Figures 5.9(a) to 5.9(f) show key parameters and measurements of a test run for a data traffic load of 150 $\frac{kBit}{s}$, e.g., the size of the packets chosen was 750 bytes. The first two graphs



(a) Distance

(b) Distance

(c) Hops

(d) Mileage

(e) Packet delivery ratio

(f) Speed

Figure 5.9: Mobile three-hop scenario: measurements. (The first two graphs are identical. They help understanding the other graphs of the same column that all share the same x axis.)

show the distance between every pair of nodes on the y-axis, while the duration of the run is visualized on the x-axis. For a better understanding, the maximum transmission range (suppression range of 220 m[8]) is also plotted. In other words, if a curve is above the line of the communication range, the corresponding pair of cars is unable to communicate. The distance graph is displayed twice to make it easier to analyze the other graphs in the left and right columns with respect to the distance between the cars. Figure 5.9(c) shows the length of the communication chain in hops. For each packet that is injected into the ad hoc network, the number of hops it had to travel to reach the destination node is displayed. A hop-count of 0 indicates that the packet has never reached the destination node. Figure 5.9(e) displays the packet delivery ratio on the y-axis over the duration of the test on the x-axis (aggregated in steps of 1 second). Figures 5.9(d) and 5.9(f) on the right side show the mileage of the car, as well as their speed. These two figures make is easier to reproduce the car's movement and match it to the map shown in Figure 5.8.

Now, we outline the expected behavior of a vehicular ad hoc network using position-based routing and compare it to the effective behavior during the test. At the start of the run, car 1 and car 4 communicate directly. Afterwards, the distance between them increases. As soon as the distance exceeds 220 meters, the routing algorithm will react and send the packets through an intermediate hop, in this case, if possible, with car 3. The fact that car 4 is no longer within communication range of car 1 is determined via a timeout mechanism. Upon receiving no position update of car 4 for at least `BEXP` seconds, car 1 deletes car 4 from its neighbor table and has to choose another node as forwarder. If there is no node available as the next hop, the packets are dropped.

Looking at Figures 5.9(a), 5.9(c), and 5.9(e), the lossless one-hop communication can easily be identified. It lasted until second 52, when car 4 exceeded the maximum communication range to car 1. It is followed by a short period in which all packets got lost, since car 1 still tried to send data directly to car 4 until it recognized after `BEXP` seconds that car 4 was no longer within communication range. Afterwards, car 3, as intermediate node, forwarded the data packets and a two-hop connection was established. This connection did not last long since car 1 had to respect the right of way at point 1 on the map, and car 4 entered the communication range again. With car 1 having already turned right and cars 2, 3 and 4 still waiting at the crossroad, car 1 directly communicated with car 4, however, the link quality was rather poor since there were many obstacles between sender and receiver. Shortly thereafter, at second 95, car 1 had no cars left within the communication range and thus stopped to inject packets into the ad hoc network. At second 125, car 4 could have been reached via three hops, but in the meantime, in car 1, the location

[8] In practical work with the demonstrator, this value worked very well for us since right after 220 m, unicast performance degraded significantly.

information on car 4 had expired (LEXP was set to 20 seconds). Due to the poor link quality, it took quite a long time until car 1 picked up the location of car 4 and was able to start to send packets over three hops.

Even in this brief look at the test run, some key observations can be made. First of all, the router either has to have information on the quality of the link or needs notification if a link is broken. This enables the router to react to poor or broken links without having to rely on timeout mechanisms. Now, we have implemented this 'lost link feature' in order to reduce packet losses after changes in the path a packet travels. Additionally, packet losses due to unstable links have to be handled by a proper transport layer, which does not exist in the current implementation of the *FleetNet* demonstrator. Another problem is the location service, which is based on MAC broadcasts. Even in this small setting it took a long period of time to detect the location of car 4 over three unstable links using unacknowledged broadcast packets.

5.2 Conclusions and Perspectives

In this chapter, we have presented the *FleetNet* demonstrator, a real-world multi-hop *Ad-Hoc Network* based on cars. In addition to giving an overview of the system, we have presented measurement results and deployment experiences. While the whole sub-project was built around demonstrability and to prove the concept, the future work in bringing vehicular networks onto the street is manifold. From a software engineering perspective, the software has to be brought to production quality on-board units. Since the focus of the first-to-be-deployed vehicle-to-vehicle networks will most likely be active safety rather than 'Internet on the road', the protocol requirements are shifting from multi-hop unicast to multi-hop geocast, and from end-to-end datagram routing to end-to-end information forwarding. For the next steps, protocol development has to converge with real-world measurements, including the precise radio hardware that will be deployed on the street. Considering this, the *FleetNet* demonstrator served its purpose, although there is still a lot to do before a final system is in place.

Beyond the scope of the system presented above, we can state that real-world implementation and testing mostly tempers down highly promising simulation results since every little difference in modeling reality is a source of great bother when the systems are run in a real car. Simulation work profits very much from a backfeed of real-world experience. In view of current ad-hoc research, it can be seen that this has created a trend in MANET research [295] because many groups no longer found their work on simulation only. However, another clear result of the practical work is the unbelievable complexity and cost of these efforts, especially compared to those of simulation studies. Thus, many groups will further be condemned to

mainly work with simulators. Personally, we have always considered ourselves lucky to be able to do this work, especially coming as we do from "practical" computer science.

A different point presented here is the significance of system architectures in the building of a system. In the process of *FleetNet*, designing the system architecture was an arduous task, and was an issue of subsequent misunderstandings. Partly, this was due to different scientists coming from different fields like electrical engineering, computer science, etc. Mainly, however this was due to the fact that classical protocol architectures for unicast communication are hard to integrate with broadcast-oriented `VANET` safety applications. Thus, we have integrated the experience from the real-world system and the corresponding communication process into some thoughts about a protocol architecture for `VANET` systems [132⋆], summarizing opposing views about protocol architecture and the impact of following either one of them. In a subsequent paper [266⋆], we start with bringing these architectural thoughts into a real software platform that alleviates the construction of `VANET` safety protocols. With this step we acknowledge the fact that safety protocols tightly integrate the know-how of vehicular safety experts who — on the other hand — still have to use the communication system in some abstract manner. Previous approaches have either reduced the communication system to a packet broadcaster or have built sub-optimal protocols.

Chapter 6

Summary, Conclusions, and Future Work

I prefer skeptics to true believers.

(Bjarne Stroustrup)

The road goes ever on and on.

(J.R.R. Tolkien, The Fellowship of the Ring.)

Battered Visions

When we started looking into *Mobile*, and particularly *Vehicular Ad-Hoc Networks*, the sky seemed to be the limit: While general randomized movement scenarios — even in simulation — hit the ceiling after a couple of hops, the geometric nature of a vehicular highway scenario did not really stress the route-finding capability of position-based routing. Thus, only the communication load appeared to be a limiting factor to which we, thinking in terms of layer separation, did devote much attention. As time went by, however, researchers like Christian Tschudin [295] started to worry about why `MANET`s are rarely deployed yet, concluding finally that people worry much more about theoretical scalability than about real-world operability. Here, it is especially the simplified radio modeling that is creating performance estimates that are often too optimistic.

Approaching the problem from an economic point of view, it is quite clear that `MANET` research largely ignored its greatest enemy, which is infrastructure. In every setting with available infrastructure, especially almost everywhere in the first world, infrastructure has a lot of fundamental technological advantages on its side, the foremost being that cell sizes are scalable due to the regulated access to the radio channel. Also, Quality-of-Service, which is of great importance for some services, e.g., voice, is a lot easier to achieve. Consequently, in the presence of infrastructure, `MANET`s can only compete by being cheaper, which is an uphill battle in view of dropping provider prices, especially when `MANET`s are by far less convenient to use.

Own Contributions

Summarizing the main contributions of this thesis, we started out complementing the formidable position-based routing method GPSR with a simple and efficient location service called *Reactive Location Service* or RLS. Previous simulation studies [171, 169] were based on the assumption that communicating nodes not only know about their own position, but also about that of the destination node. This assumption abstracts from the specific location service used. However, it tilts the advantage slightly in favor of position-based over topology-based methods. Thus, it was not clear whether or not position-based routing really outperforms topology-based methods on a quantitative scale. The study we performed for RLS, however, demonstrates the applicability of RLS in a variety of randomized scenarios. Moreover, the flooding part of RLS was shown to improve neighbor table accuracy right before routes are selected.

The most fundamental achievement of this thesis is certainly *Contention-Based Forwarding*, or CBF. With this method, introduced in Section 3.3, we not only present a routing scheme that is very well suited for location-enabled high-density and high-mobility networks, but also offer a new design paradigm for MANET routing protocols. The list of paradigms that is given in Section 2.4 can now be completed by **opportunistic vs. explicit next-hop selection** meaning that with opportunistic methods address all neighbors, who then decide for themselves, if they are well suited to accomplish the task. With traditional explicit methods, this decision is done by the current hop on the basis of its current information.

As the first opportunistic unicast routing scheme, CBF makes use of this technique to overcome the problem of outdated neighbor tables, from which GPSR suffers at high mobility. Also, the opportunistic nature of CBF allows for every single packet broadcast to utilize the best neighbor that was able to decode the packet as a forwarder. The older, explicit, methods selected a forwarder they assumed to be the best, and then potentially needed more than one transmission to actually get the packet there, sometimes they were not able to reach it at all. For the new methods, we have studied the issue of packet duplication introduced by *plain* CBF by providing to enhanced duplication avoidance strategies, namely *area / Reuleaux - based* and *explicit forwarder selection.*

addressed the issue of packet duplication

Pure CBF is only able to find 'greedy' routes, i.e., routes where every next hop is geographically closer to the destination than the current node. However, the set of greedy routes is a real subset of the set of all possible routes, with the obvious consequence that pure CBF is not a complete routing algorithm. However, when studying the likelihood of non-greedy routes in scenarios with random node placement, simulation showed that 'greedy' still finds many of the available routes,

keeping in mind, that in combination with CBF, mobility almost does not matter any more.

Still, we have proposed CBDV, or *Contention-Based Distance-Vector Routing* as a complete and opportunistic routing method, being able to (a) route stand-alone, or (b) as a non-greedy companion to CBF. CBDV exploits the fact that distance-vector routing is, in principal, a greedy algorithm, but unlike CBF, on the basis of topological distance from a desired target rather than on the basis of geographical distance. In other words, a distance-vector route request cycle within a certain area determines the topological distance of every node from the target. CBDV then simply performs a contention process on this topological ordering. A remarkable feature of CBDV is the applicability of most of the research that has been done around AODV, which could be regarded as its non-opportunistic uncle.

With a promising solution for routing in general *Ad-Hoc Networks*, we now come back to the roots of our work within the *Vehicular Ad-Hoc Network* community. As constituted in Chapter 3, the VANET routing problem can be separated into routing on highways and routing in cities. For the former, we have studied position-based routing with beacons, complemented by our *Reactive Location Service*. As we had expected, we found that topology-based routing practically collapses for everything further away than only a couple of hops, while position-based routing is able to sustain its performance. On the contra-side of beacon-based methods, we found that with low beaconing frequencies, the damage induced by the inaccuracy of the neighbor tables grows increasingly. However, in a subsequent simulation study, we were able to show that this disadvantage can largely be compensated by using dead-reckoning to increase the prediction of the neighbors' positions.

The superiority of position-based routing on highways is further amplified by using CBF because it is able to bring in its superior performance in those highly mobile networks without the negative issues of potential packet duplication and non-greedy routes. These effects are eliminated by the geometric nature of the scenario. In a very recent study, we have confirmed CBF's superiority even in the presence of probabilistic radio models. In scenarios with high node densities, the probabilistic nature of the channel always caused at least one distant node to hear (and consequently to forward) the message. This effect created a favorably low number of hops, resulting in low transmission costs and a very low end-to-end delay.

The more complicated city, or 2D, scenarios have also attracted of our research attention. Studying these scenarios, we have shown that while GPSR's perimeter strategy theoretically completes position-based routing, it does not work well to solve the city routing problem. Thus, we have created various improvements for planar graph routing. *Restricted greedy* prefers forwarders on junctions because they can forward into more than one street. GPCR goes one step further by applying

the left-hand rule on the planar street graph. Also, it uses statistical methods to detect junctions.

A slightly different approach was presented with STBR. This algorithms treats the whole street graph as a network in which the junctions are the nodes and the streets are the links. Links can be either up or down, depending on whether or not there is a valid forwarding chain between adjacent junctions. Standard topology-based routing methods are now applied to this graph, acknowledging the highly stochastic nature of the street links.

All of the above is rather theoretical, having been created in simulators in scenarios that would have been very difficult or expensive to realize. However, we take pride in the fact that we also have created a real position-based routing system for vehicles, even if on a much smaller scale. This was possible only with the kind support of our colleagues at DaimlerChrysler Research in Ulm and at NEC's network labs in Heidelberg. With these partners, we have built the *FleetNet* demonstrator, consisting of Smart Cars stationed in Ulm and equipped with 802.11 radio antennas and GPS systems. For these cars, we used standard notebooks with PCMCIA wireless cards to create an ad-hoc network able to (a) send vehicle-specific geocast messages, and (b) to transparently connect IP subnets. For this demonstrator, we have both developed the system architecture and programmed the routing software as a Linux user space daemon. While the building of a beacon-based greedy router might seem trivial, it proved to be quite a challenge to make it efficient. Finally, we have learned a multitude of lessons from the practical system and fed them back to the more abstract part of our research.

The same holds true for the practical implementation of CBF on a sensor platform. While it clearly showed that CBF is an applicable and functional concept, it also revealed problems with regard to modeling assumptions that do not hold in reality.

New Hope and Outlook

Clearly, MANETs might still be indispensable in traditional "absence of infrastructure" scenarios. So one obvious research challenge is (a) to make routing more robust under realistic radio and system conditions, and (b), to complement it with congestion control methods capable of both exploiting the available bandwidth and not choking the network in the process.[1]

Also, there is room for MANETs where the purpose of communication is very local and needs an extremely low delay. As you might have guessed, a valid example

[1] Fortunately, people have already started to look into these directions. For (a) not only are protocols getting more robust, e.g., [212], but also try to better understand radio system modeling [181, 163]. For (b), CXCC [7] and *Path Density Protocol* [236] are interesting approaches to tackling the problem.

of this is the exchange of safety messages in *Vehicular Ad-Hoc Networks*. In these scenarios, we identify the main challenge to build protocols that can cope with both extremely low and extremely high node density. In this context, we proclaim information forwarding as opposed to packet forwarding to be the next step, meaning that nodes modify the packet payload between hops. In fact, Prof. Hartenstein's group is extending CBF in that direction.

For the extension of CBF, we would like to see it integrated with *network coding*, a very interesting technique to save bandwidth by transmitting invertible bit combinations of packets [46]. Also, it will be interesting to see how CBF integrates with new MAC/PHY proposals like *ultra wideband* [57] or CDMA [24]. Also, one could think of different strategies for duplicate reduction, e.g., by physically creating a higher reception range for packet headers, or by using different frequencies and MACs, one for forwarding and one for duplicate suppression.

When it comes to architecture, standard protocol layer separation is difficult to maintain when relaxing end-to-end payload integrity, demanding instead new system designs [266*, 265*]. It is very likely that real implementations and field testing will gain increased significance in the next couple of years preluding VANET roll-out.

Also, we identify the topic of hybrid ad-hoc networks, that is wireless-cum-wired communication scenarios, as a promising approach to solve the problem of non-local communication. Recent research discusses their improved scalability [207]. Also, [90] identifies the enormous number of available free access points, which might be integrated into VANETs in the future, especially for communication applications that are not directly related to safety. Even more, VANET applications can be made more useful by the integration of cellular networks, especially at the beginning of deployment.

Appendix A

RLS — Distribution of the Back-Off Timer

(Related to Section 3.1)

Let X be a two-dimensional vector that denotes a point in a circle with radius d_{rrange}, chosen randomly from the set of all points uniformly distributed in the area enclosed by said circle. Furthermore, let $D = d_{last}/d_{rrange}$ be the normalized, Euclidean distance of X from the center of the circle with radius d_{rrange}. Then

$$F_D(d) = P(D \leq d) = \begin{cases} 0 & d < 0 \\ d^2 & d \in [0;1] \\ 1 & d > 1 \end{cases} \tag{A.1}$$

is the quadratic distribution function of D.

By applying the conversion function $g(D)$ with

$$g(d) = d^2 \qquad d \in [0;1] \tag{A.2}$$

we can transform D into a new random variable U that, according to the proof in [237], is uniformly distributed over $[0;1]$.

Thus, the back-off timer function

$$t_{backoff} = T_{max} \cdot \left(1 - \left(\frac{d_{last}}{d_{rrange}}\right)^2\right) \tag{A.3}$$

is uniformly distributed over $[0;T_{max}]$, with the smallest back-off times for the largest values of d_{last}. This means that the nodes farthest away from the center of the circle have the smallest back-off period.

Bibliography

The citations marked with an ⋆ are all publications Holger Füßler has co-authored. The ones with a ′ are student theses he has tutored. Both lists are in order of their publication, resp. submission.

Own Publications	[301⋆, 194⋆, 125⋆, 195⋆, 193⋆, 222⋆, 126⋆, 134⋆, 133⋆, 219⋆, 127⋆, 220⋆, 210⋆, 221⋆, 135⋆, 149⋆, 290⋆, 201⋆, 124⋆, 228⋆, 173⋆, 291⋆, 288⋆, 261⋆, 185⋆, 131⋆, 209⋆, 260⋆, 132⋆, 129⋆, 128⋆, 289⋆, 259⋆, 175⋆, 292⋆, 262⋆, 265⋆, 176⋆, 266⋆, 263⋆, 130⋆]
Tutored Theses	[198′, 287′, 208′, 172′, 233′, 227′, 192′, 89′, 258′, 200′, 285′, 123′, 196′, 184′, 177′]

[1] IEEE 802.11 Task Group. http://grouper.ieee.org/groups/802/11/, 2006. [Online: Accessed at 2006-04-07]. (Cited on pages 18, 25, 103, and 189.)

[2] ACM Annual International Conference on Mobile Computing and Networking (MobiCom). http://www.sigmobile.org/mobicom, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 6.)

[3] ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc). http://www.sigmobile.org/mobihoc, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 6.)

[4] ARTEM GmbH. http://www.artem.de, 2006. [Online: Accessed at 2003-11-22]. (Cited on page 188.)

[5] boost — C++ Libraries. http://boost.org, 2006. [Online: Accessed at 2006-05-09]. (Cited on page 104.)

[6] The CMU Monarch Wireless and Mobility Extensions to ns-2. http://www.monarch.cs.cmu.edu/cmu-ns.html, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 59.)

[7] Cooperative Crosslayer Congestion Control. http://www.cn.uni-duesseldorf.de/projects/CXCC, 2006. [Online: Accessed at 2006-04-21]. (Cited on pages 57 and 208.)

[8] The FleetNet Project Homepage. http://www.et2.tu-harburg.de/fleetnet/, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 185.)

[9] Google Maps: Eselsberg, Ulm, Germany. http://maps.google.com/maps?f=q&hl=en&q=Albert-Einstein-Allee,+89081+Ulm,+Ulm,+Baden-Wurttemberg,+Germany&ie=UTF8&z=14&ll=48.421796,9.951124&spn=0.016006,0.047293&t=k&om=1, 2006. [Online: Accessed at 2006-09-25]. (Cited on page 199.)

[10] Huginn Homepage. http://www.informatik.uni-mannheim.de/pi4/projects/Huginn/, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 59.)

[11] HWGui and HighwayMovement Homepage. http://www.informatik.uni-mannheim.de/pi4/projects/HWGui/, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 147.)

[12] IPerf - The TCP/UDP Bandwidth Measurement Tool. http://dast.nlanr.net/Projects/Iperf/. (Cited on page 197.)

[13] Wireless Tools for Linux. http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 195.)

[14] Manpage of LIBIPQ. http://www.cs.princeton.edu/~nakao/libipq.htm, 2007. [Online: Accessed at 2007-02-01]. (Cited on page 193.)

[15] The ns-2 network simulator. http://www.isi.edu/nsnam/ns/, 2006. [Online: Accessed at 2006-04-07]. (Cited on pages 59, 108, and 163.)

[16] MIT RoofNet Homepage. http://pdos.csail.mit.edu/roofnet/, 2006. [Online: Accessed at 2006-04-07]. (Cited on pages 19 and 134.)

[17] Wikipedia: IEEE 802.11. http://en.wikipedia.org/w/index.php?title=IEEE_802.11&oldid=47004190, 2006. [Online: Accessed at 2006-04-07]. (Cited on pages 25 and 163.)

[18] Wikipedia: Autobahn. http://en.wikipedia.org/w/index.php?title=Autobahn&oldid=73189009, 2006. [Online: Accessed at 2006-09-01]. (Cited on page 143.)

[19] Wikipedia: Breadth-first search. http://en.wikipedia.org/w/index.php?title=Breadth-first_search&oldid=54584556, 2006. [Online: Accessed at 2006-05-24]. (Cited on page 173.)

[20] Wikipedia: Big O notation — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Big_O_notation&oldid=87142616, 2006. [Online: Accessed at 2006-11-13]. (Cited on page 6.)

[21] Wikipedia: Bloom filter. http://en.wikipedia.org/w/index.php?title=Bloom_filter&oldid=45226975, 2006. [Online: Accessed at 2006-04-20]. (Cited on page 47.)

[22] Wikipedia: Voice over IP. http://en.wikipedia.org/w/index.php?title=Cache&oldid=48544690, 2006. [Online: Accessed at 2006-04-19]. (Cited on page 36.)

[23] Wikipedia: Controller Area Network. http://en.wikipedia.org/w/index.php?title=Controller_Area_Network&oldid=77479972, 2006. [Online: Accessed at 2006-09-25]. (Cited on page 190.)

[24] Wikipedia: Code division multiple access — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Code_division_multiple_access&oldid=104743346, 2007. [Online: Accessed at 2007-02-01]. (Cited on page 209.)

[25] WIKIPEDIA: Christopher Columbus. http://en.wikipedia.org/w/index.php?title=Christopher_Columbus&oldid=47697892, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 31.)

[26] WIKIPEDIA: Correlation. "http://en.wikipedia.org/w/index.php?title=Correlation&oldid=53253922, 2006. [Online: Accessed at 2006-05-24]. (Cited on page 175.)

[27] WIKIPEDIA: C++. http://en.wikipedia.org/w/index.php?title=C++&oldid=49875062, 2006. [Online: Accessed at 2006-04-24]. (Cited on page 59.)

[28] WIKIPEDIA: Cyclic redundancy check. http://en.wikipedia.org/w/index.php?title=Cyclic_redundancy_check&oldid=46287781, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 26.)

[29] WIKIPEDIA: Daemon (computer software). http://en.wikipedia.org/w/index.php?title=Daemon_9&oldid=77042215, 2006. [Online: Accessed at 2006-09-25]. (Cited on page 192.)

[30] WIKIPEDIA: DARPA. http://en.wikipedia.org/w/index.php?title=Defense_Advanced_Research_Projects_Agency&oldid=71571853, 2006. [Online: Accessed at 2006-09-04]. (Cited on page 2.)

[31] WIKIPEDIA: Dead Reckoning. http://en.wikipedia.org/w/index.php?title=Dead_reckoning&oldid=72107454, 2006. [Online: Accessed at 2006-09-01]. (Cited on page 154.)

[32] WIKIPEDIA: Dijkstra's algorithm. http://en.wikipedia.org/w/index.php?title=Dijkstra's_algorithm&oldid=47066816, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 28.)

[33] WIKIPEDIA: Dominating set problem. http://en.wikipedia.org/w/index.php?title=Dominating_set_problem&oldid=46556637, 2006. [Online: Accessed at 2006-04-18]. (Cited on page 40.)

[34] WIKIPEDIA: Distance-vector routing protocol. http://en.wikipedia.org/w/index.php?title=Distance-vector_routing_protocol&oldid=46305150, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 28.)

[35] WIKIPEDIA: Ethernet. http://en.wikipedia.org/w/index.php?title=Ethernet&oldid=47362998, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 25.)

[36] WIKIPEDIA: Euclidean distance — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Euclidean_distance&oldid=76839317, 2006. [Online: Accessed at 2006-11-15]. (Cited on page 30.)

[37] WIKIPEDIA: Forward error correction. http://en.wikipedia.org/w/index.php?title=Forward_error_correction&oldid=51293847, 2006. [Online: Accessed at 2006-05-11]. (Cited on page 136.)

[38] WIKIPEDIA: Galileo Positioning System. http://en.wikipedia.org/w/index.php?title=Galileo_positioning_system&oldid=47704047, 2006. [Online: Accessed at 2006-04-10]. (Cited on page 32.)

[39] WIKIPEDIA: Global Positioning System. http://en.wikipedia.org/w/index.php?title=Global_Positioning_System&oldid=47464797, 2006. [Online: Accessed at 2006-04-10]. (Cited on page 32.)

[40] WIKIPEDIA: Greedy Algorithm. http://en.wikipedia.org/w/index.php?title=Greedy_algorithm&oldid=48080989, 2006. [Online: Accessed at 2006-04-20]. (Cited on pages 30 and 49.)

[41] WIKIPEDIA: HIPERLAN. http://en.wikipedia.org/w/index.php?title=HIPERLAN&oldid=44660616, 2006. (Cited on page 25.)

[42] WIKIPEDIA: ISM Band. http://en.wikipedia.org/w/index.php?title=ISM_band&oldid=73441938, 2006. [Online: Accessed at 2006-09-04]. (Cited on page 1.)

[43] WIKIPEDIA: Link-state routing protocol. http://en.wikipedia.org/w/index.php?title=Link-state_routing_protocol&oldid=45549607, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 28.)

[44] WIKIPEDIA: Navier-Stokes equations. http://en.wikipedia.org/w/index.php?title=Navier-Stokes_equations&oldid=74916844, 2006. [Online: Accessed at 2006-09-12]. (Cited on page 143.)

[45] WIKIPEDIA: Navigation. http://en.wikipedia.org/w/index.php?title=Navigation&oldid=46849791, 2006. [Online: Accessed at 2006-04-10]. (Cited on page 30.)

[46] WIKIPEDIA: Network Coding — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Network_coding&oldid=100780292, 2007. [Online: Accessed at 2007-02-01]. (Cited on page 209.)

[47] WIKIPEDIA: PCMCIA. http://en.wikipedia.org/w/index.php?title=PC_card&oldid=77278513, 2006. [Online: Accessed at 2006-09-25]. (Cited on page 189.)

[48] WIKIPEDIA: Radar. http://en.wikipedia.org/w/index.php?title=Radar&oldid=76161287, 2006. [Online: Accessed at 2006-09-20]. (Cited on page 22.)

[49] WIKIPEDIA: Request for Comments. http://en.wikipedia.org/w/index.php?title=Request_for_Comments&oldid=48545800x. (Cited on page 42.)

[50] WIKIPEDIA: Right-hand rule. http://en.wikipedia.org/w/index.php?title=Right-hand_rule&oldid=46812021, 2006. [Online: Accessed at 2006-04-21]. (Cited on page 52.)

[51] WIKIPEDIA: Smart (automobile). http://en.wikipedia.org/w/index.php?title=Smart_77385965, 2006. [Online: Accessed at 2006-09-25]. (Cited on page 188.)

[52] WIKIPEDIA: Signal-to-noise ratio. http://en.wikipedia.org/w/index.php?title=Signal-to-noise_ratio&oldid=72343333, 2006. [Online: Accessed at 2006-09-01]. (Cited on page 162.)

[53] WIKIPEDIA: Subnetwork. `http://en.wikipedia.org/w/index.php?title=Subnetwork&oldid=75399489`, 2006. [Online: Accessed at 2006-09-25]. (Cited on page 191.)

[54] WIKIPEDIA: Tcl. `http://en.wikipedia.org/w/index.php?title=Tcl&oldid=48654914`, 2006. [Online: Accessed at 2006-04-24]. (Cited on page 59.)

[55] WIKIPEDIA: Transmission Control Protocol. `http://en.wikipedia.org/w/index.php?title=Transmission_Control_Protocol&oldid=49012843`, 2006. [Online: Accessed at 2006-04-21]. (Cited on page 56.)

[56] WIKIPEDIA: Time Division Multiple Access. `http://en.wikipedia.org/w/index.php?title=Time_division_multiple_access&oldid=45262254`, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 26.)

[57] WIKIPEDIA: Ultra-wideband — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Ultra-wideband&oldid=103976326`, 2007. [Online: Accessed at 2007-02-01]. (Cited on page 209.)

[58] WIKIPEDIA: Voice over IPy. `http://en.wikipedia.org/w/index.php?title=Voice_over_IP&oldid=47303960`, 2006. [Online: Accessed at 2006-04-07]. (Cited on page 1.)

[59] Norman Abramson. Development of the ALOHANET. *IEEE Transactions on Information Theory*, 31(2):119–123, March 1985. (Cited on page 22.)

[60] Aarti Agarwal and Samir R. Das. Dead Reckoning in Mobile Ad Hoc Networks. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '03)*, New Orleans, LA, March 2003. (Cited on page 154.)

[61] ANSI/IEEE Std 802.11, 1999. (Cited on pages 25, 96, and 103.)

[62] Hari Balakrishnan, Venkata N. Padmanabhan, and Randy H. Katz. The effects of asymmetry on TCP performance. *Mobile Networks and Applications*, 4(3):219–241, October 1999. (Cited on page 56.)

[63] Hari Balakrishnan, Venkata N. Padmanabhan, Srinivasan Seshan, and Randy H. Katz. A comparison of mechanisms for improving TCP performance over wireless links. *IEEE/ACM Transactions on Networking (TON)*, 5(6):756–769, December 1997. (Cited on pages 56 and 169.)

[64] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama. Dynamical model of traffic congestion and numerical simulation. *Physical Review*, E 51:1035–1042, 1995. (Cited on page 170.)

[65] Lichun Bao and J. J. Garcia-Luna-Aceves. Transmission scheduling in ad hoc networks with directional antennas. In *Proceedings of the Eighth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '02)*, pages 48–58, Atlanta, Georgia, September 2002. (Cited on page 136.)

[66] Chris Barrett, Achia Marathe, Madhav V. Marathe, and Martin Drozda. Characterizing the interaction between routing and MAC protocols in ad-hoc networks. In *Proceedings of the Third ACM international Symposium on Mobile and Ad Hoc*

Networking & computing (MobiHoc '02), Lausanne, Switzerland, June 2002. (Cited on page 27.)

[67] Stefano Basagni, Imrich Chlamtac, and Violet R. Syrotiuk. Dynamic source routing for ad hoc networks using the global positioning system. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '99)*, pages 301–305, New Orleans, LA, September 1999. (Cited on page 44.)

[68] Stefano Basagni, Imrich Chlamtac, Violet R. Syrotiuk, and Barry A. Woodward. A Distance Routing Effect Algorithm for Mobility (DREAM). In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '98)*, pages 76–84, Dallas, Texas, October 1998. (Cited on pages 45, 48, 49, 54, and 64.)

[69] R.E. Bellmann. *Dynamic Programming*. Princeton University Press, N.J., 1957. (Cited on page 29.)

[70] Jon Bentley. *Programming Pearls*. Addison-Wesley, 2nd edition, 2000. (Cited on page 36.)

[71] T. Benz, L. Schäfers, C. Stiller, and D. Vollmer. Feasibility Study on Truck Planning on European Motorways. Deliverable D08.1 of ITS project PROMOTE-CHAUFFEUR, 1999. (Cited on page 143.)

[72] Christian Bettstetter. Mobility Modeling in Wireless Networks: Categorization, Smooth Movement, and Border Effects. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 5(3):55–67, July 2001. (Cited on pages 9, 59, and 108.)

[73] Christian Bettstetter. On the minimum node degree and connectivity of a wireless multihop network. In *Proceedings of the Third ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '02)*, pages 80–91, Lausanne, Switzerland, June 2002. (Cited on page 9.)

[74] Christian Bettstetter, Hannes Hartenstein, and Xavier Pérez-Costa. Stochastic Properties of the Random Waypoint Mobility Model. *ACM/Kluwer Wireless Networks, Special Issue on Modeling & Analysis of Mobile Networks*, 9(2), 2003. (Cited on page 9.)

[75] Vaduvur Bharghavan, Alan Demers, Scott Shenker, and Lixia Zhang. MACAW: a media access protocol for wireless LAN's. In *Proceedings of the Conference on Communications architectures, protocols and applications (SIGCOMM '94)*, pages 212–225, London, United Kingdom, August 1994. (Cited on page 25.)

[76] Sanjit Biswas. Opportunistic Routing in Multi-Hop Wireless Networks. Master's thesis, M.I.T., March 2005. (Cited on page 134.)

[77] Sanjit Biswas and Robert Morris. Opportunistic Routing in Multi-Hop Wireless Networks. *ACM SIGCOMM Computer Communication Review (CCR)*, 34(1):69–74, January 2004. (Cited on page 134.)

[78] Sanjit Biswas and Robert Morris. ExOR: Opportunistic Routing in Multi-Hop Wireless Networks. In *Proceedings of the 2005 Conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '05)*, pages 133–144, Philadelphia, PA, August 2005. (Cited on page 134.)

[79] Ljubica Blažević. *Scalable Routing Protocols with Applications to Mobility*. PhD thesis, Swiss Federal Institute of Technology (EPFL), February 2002. (Cited on page 134.)

[80] Ljubica Blažević, Silvia Giordano, and Jean-Yves LeBoudec. Self-Organizing Wide-Area Routing. In *Proceedings of SCI 2000/ISAS 2000*, Orlando, July 2000. (Cited on page 173.)

[81] Burton H. Bloom. Space/Time Trade-offs in Hash Coding with Allowable Errors. *Communications of the ACM*, 13(7):422–426, July 1970. (Cited on page 47.)

[82] Bernd Bochow and Marc Bechler. Internet Integration. In Walter Franz, Hannes Hartenstein, and Martin Mauve, editors, *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles—The FleetNet Project*, pages 175–211. Universitätsverlag Karlsruhe, Karlsruhe, Germany, November 2005. (Cited on pages 17 and 186.)

[83] Rajendra V. Boppana and Satyadeva P. Konduru. An Adaptive Distance Vector Routing Algorithm for Mobile, Ad Hoc Networks. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2001)*, pages 1753–1762, Anchorage, Alaska, April 2001. (Cited on page 42.)

[84] Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting Mobile Communications: The Insecurity of 802.11. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '01)*, Rome, Italy, July 2001. (Cited on page 26.)

[85] Prosenjit Bose, Pat Morin, Ivan Stojmenovic, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc Wireless Networks. In *Proceedings of the 3rd international Workshop on Discrete Algorithms and Methods for Mobile Computing and communications (DIAL-M '99)*, pages 48–55, Seattle, WS, August 1999. (Cited on pages 51, 115, and 117.)

[86] Linda Briesemeister, Lorenz Schäfer, and Günter Hommel. Disseminating messages among highly mobile hosts based on inter-vehicle communication. In *Proc. of IEEE Intelligent Vehicles Symposium (IV2000)*, pages 522–527, Dearborn, MI, October 2000. (Cited on page 134.)

[87] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta G. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '98)*, pages 85–97, Dallas, Texas, October 1998. (Cited on pages 44 and 59.)

[88] Andrei Broder and Michael Mitzenmacher. Network Applications of Bloom Filters: A Survey, 2002. (Cited on page 47.)

[89′] Thomas Butter. Contention-Based Multicast Forwarding for Mobile Ad-Hoc Networks. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, 2004. (Cited on page 213.)

[90] Vladimir Bychkovsky, Brett Hull, Allen Miu, Hari Balakrishnan, and Samuel Madden. A Measurement Study of Vehicular Internet Access Using In Situ Wi-Fi Networks. In *Proceedings of the Twelvth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '06)*, Los Angeles, CA, September 2006. (Cited on page 209.)

[91] Tracy Camp, Jeff Boleng, and Vanessa Davies. A Survey of Mobility Models for Ad Hoc Network Research. *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502, 2002. (Cited on page 9.)

[92] Tracy Camp, Jeff Boleng, and Lucas Wilcox. Location Information Services in Mobile Ad Hoc Networks. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 3318–3324, New York City, New York, April 2002. (Cited on page 49.)

[93] Tracy Camp, Jeff Boleng, Brad Williams, Lucas Wilcox, and William Navidi. Performance Comparison of Two Location Based Routing Protocols for Ad Hoc Networks. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, pages 1678–1687, New York City, New York, June 2002. (Cited on pages 44 and 45.)

[94] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626, `http://www.ietf.org/rfc3626.txt`, 2003. Status: EXPERIMENTAL. (Cited on page 45.)

[95] Thomas Clausen, Philippe Jacquet, Anis Laouiti, Pascale Minet, Paul Muhlethaler, Amir Quayyum, and Laurent Viennot. Optimized Link State Routing Protocol, October 2001. (Cited on page 45.)

[96] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT press, 2001. (Cited on pages 28, 49, and 126.)

[97] M. Scott Corson and Joseph Macker. Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, January 1999. (Cited on page 127.)

[98] S. Corson and J. Macker. Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations. RFC 2501, `http://www.ietf.org/rfc2501.txt`, 1999. Status: INFORMATIONAL. (Cited on page 36.)

[99] David Culler, Deborah Estrin, and Mani Srivastava. Overview of Sensor Networks. *IEEE Computer—Special Issue on Sensor Networks*, 37(8):41–49, August 2004. (Cited on page 18.)

[100] Samir R. Das, Charles E. Perkins, and Elizabeth M. Royer. Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks. In *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2000)*, pages 3–12, Tel Aviv, Israel, March 2000. (Cited on page 54.)

[101] Douglas S. J. DeCouto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the Ninth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '03)*, pages 134–146, San Diego, California, September 2003. (Cited on page 41.)

[102] Douglas S. J. DeCouto, Daniel Aguayo, Benjamin A. Chambers, and Robert Morris. Performance of multihop wireless networks: shortest path is not enough. *ACM SIGCOMM Computer Communication Review (CCR)*, 33(1):82–88, 2003. (Cited on page 41.)

[103] Douglas S. J. DeCouto and Robert Morris. Location Proxies and Intermediate Node Forwarding for Practical Geographic Forwarding. Technical Report MIT-LCS-TR-824, MIT, June 2001. (Cited on page 117.)

[104] Reinhard Diestel. *Graph Theory*. Springer-Verlag, New York, electronic edition, 2000. (Cited on page 28.)

[105] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, (1):269–271, 1959. (Cited on page 173.)

[106] Wolfgang Domschke and Andreas Drexl. *Einführung in Operations Research*. Springer, Berlin, 6th edition, October 2004. (Cited on page 35.)

[107] Olivier Dousse, François Baccelli, and Patrick Thiran. Impact of Interferences on Connectivity in Ad Hoc Networks. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*, San Francisco, CA, March 2003. (Cited on page 22.)

[108] Thomas D. Dyer and Rajendra V. Boppana. A comparison of TCP performance over three routing protocols for mobile ad hoc networks. In *Proceedings of the Second ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '01)*, pages 56–66, Long Beach, California, October 2001. (Cited on page 56.)

[109] André Ebner, Lars Wischhof, Hermann Rohling, Rüdiger Halfmann, and Matthias Lott. Time Synchronization in Highly Dynamic Ad Hoc Networks. In Walter Franz, Hannes Hartenstein, and Martin Mauve, editors, *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles—The FleetNet Project*, pages 1–28. Universitätsverlag Karlsruhe, Karlsruhe, Germany, November 2005. (Cited on page 27.)

[110] Hala Elaarag. Improving TCP performance over mobile networks. *ACM Computing Surveys (CSUR)*, 34(3):357–374, 2002. (Cited on page 57.)

[111] Wilfried Enkelmann. FleetNet - Applications for Inter-Vehicle Communication. In *Proc. of IEEE Intelligent Vehicles Symposium (IV2003)*, pages 162–167, Columbus, OH, June 2003. (Cited on page 18.)

[112] Wilfried Enkelmann. Applications for Inter-Vehicle Communication. In Walter Franz, Hannes Hartenstein, and Martin Mauve, editors, *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles—The FleetNet Project*, pages 213–231. Universitätsverlag Karlsruhe, Karlsruhe, Germany, November 2005. (Cited on page 18.)

[113] Deborah Estrin, Ramesh Govindan, John Heidemann, and Satish Kumar. Next century challenges: scalable coordination in sensor networks. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, pages 263–270, Seattle, Washington, August 1999. (Cited on page 18.)

[114] Kevin Fall and Kannan Varadhan. *The ns Manual (formerly ns Notes and Documentation)*. UC Berkeley, LBL, USC/ISI, and Xerox PARC, April 2006. (Cited on page 59.)

[115] András Faragó. Scalable Analysis and Design of Ad Hoc Networks Via Random Graph Theory. In *Proceedings of the Sixth international Workshop on Discrete Algorithms and Methods for Mobile Computing and communications (DIAL-M '02)*, pages 43–50, Atlanta, Georgia, September 2002. (Cited on pages 60 and 124.)

[116] Niels Ferguson and Bruce Schneier. *Practical Cryptography*. Wiley Publishing, Inc., 2003. (Cited on page 26.)

[117] Gregory G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI/RR-87-180, ISI, March 1987. (Cited on pages 50 and 116.)

[118] Roland Flury and Roger Wattenhofer. MLS: An Efficient Location Service for Mobile Ad Hoc Networks. In *Proceedings of the 7th ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '06)*, Florence, Italy, May 2006. (Cited on page 49.)

[119] Walter Franz and Christian Maihöfer. Geographical Addressing and Forwarding in FleetNet, 2003. (Cited on page 194.)

[120] James A. Freebersyser and Barry Leiner. A DoD Perspective on Mobile Ad Hoc Networks. In Charles E. Perkins, editor, *Ad Hoc Networking*, pages 29–48. Addison-Wesley, New Jersey, 2001. (Cited on page 6.)

[121] Zhenghua Fu, Xiaoqiao Meng, and Songwu Lu. How Bad TCP Can Perform In Mobile Ad Hoc Networks. In *Proceedings of the Seventh IEEE Symposium on Computers and Communications (ISCC '02)*, Taormina/Giardini Naxos, Italy, July 2002. (Cited on pages 56 and 169.)

[122] Thomas T. Fuhrmann and Jörg Widmer. On the Scaling of Feedback Algorithms for Very Large Multicast Groups. *Special Issue of Computer Communications on Integrating Multicast into the Internet*, 24(5-6):539–547, March 2001. (Cited on pages 96 and 134.)

[123′] Daniel Förderer. Street-Topology Based Routing. Diplomarbeit, University of Mannheim, May 2005. (Cited on pages 172, 173, 179, 184, and 213.)

[124⋆] Holger Füßler, Hannes Hartenstein, Jörg Widmer, Martin Mauve, and Wolfgang Effelsberg. Contention-Based Forwarding for Street Scenarios. In *Proceedings of the 1st International Workshop on Intelligent Transportation*, pages 155–159, Hamburg, Germany, March 2004. (Cited on pages 139, 162, and 213.)

[125⋆] Holger Füßler, Martin Mauve, Hannes Hartenstein, Michael Käsemann, and Dieter Vollmer. A Comparison of Routing Strategies for Vehicular Ad Hoc Networks. Technical Report TR-02-003, Department of Computer Science, University of Mannheim, July 2002. (Cited on pages 44, 139, 145, 147, 157, 161, 182, and 213.)

[126⋆] Holger Füßler, Martin Mauve, Hannes Hartenstein, Michael Käsemann, and Dieter Vollmer. Poster: Location-Based Routing for Vehicular Ad-Hoc Networks. In *Proceedings of the Eighth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '02) (electronic edition)*, Atlanta, Georgia, September 2002. (Cited on pages 44, 139, and 213.)

[127⋆] Holger Füßler, Martin Mauve, Hannes Hartenstein, Michael Käsemann, and Dieter Vollmer. MobiCom Poster: Location-Based Routing for Vehicular Ad-hoc Networks. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 7(1):47–49, January 2003. (Cited on pages 139 and 213.)

[128⋆] Holger Füßler, Martin Mauve, Hannes Hartenstein, Christian Lochert, Dieter Vollmer, Dagmar Herrmann, and Walter Franz. Position-Based Routing in Ad-Hoc Wireless Networks. In Walter Franz, Hannes Hartenstein, and Martin Mauve, editors, *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles—The FleetNet Project*, pages 117–143. Universitätsverlag Karlsruhe, Karlsruhe, Germany, November 2005. (Cited on pages 139, 145, 147, 184, and 213.)

[129⋆] Holger Füßler, Michael Möske, Hannes Hartenstein, Walter Franz, Andreas Festag, and Christian Wagner. A Position-Based Router: Design, Implementation and Measurements. In Walter Franz, Hannes Hartenstein, and Martin Mauve, editors, *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles—The FleetNet Project*, pages 145–174. Universitätsverlag Karlsruhe, Karlsruhe, Germany, November 2005. (Cited on pages 187 and 213.)

[130⋆] Holger Füßler, Sascha Schnaufer, Matthias Transier, and Wolfgang Effelsberg. Vehicular Ad-Hoc Networks: From Vision to Reality and Back. In *Proceedings of the Fourth Annual IEEE/IFIP Conference on Wireless On demand Network Systems and Services (WONS '07))*, Obergurgl, Austria, January 2007. (Cited on pages 187 and 213.)

[131⋆] Holger Füßler, Marc Torrent-Moreno, Roland Krüger, Matthias Transier, Hannes Hartenstein, and Wolfgang Effelsberg. Studying Vehicle Movements on Highways and their Impact on Ad-Hoc Connectivity. Technical Report TR-2005-003, Department of Mathematics and Computer Science, University of Mannheim, 2005. (Cited on pages 147, 157, and 213.)

[132⋆] Holger Füßler, Marc Torrent-Moreno, Matthias Transier, Andreas Festag, and Hannes Hartenstein. Thoughts on a Protocol Architecture for Vehicular Ad-Hoc Networks. In *Proceedings of 2nd International Workshop on Intelligent Transportation*, pages 41–45, Hamburg, Germany, March 2005. (Cited on pages 187, 203, and 213.)

[133⋆] Holger Füßler, Jörg Widmer, Michael Käsemann, Martin Mauve, and Hannes Hartenstein. Beaconless Position-Based Routing for Mobile Ad-Hoc Networks. Tech-

nical Report TR-03-001, Department of Computer Science, University of Mannheim, 2003. (Cited on pages 44, 64, 126, and 213.)

[134⋆] Holger Füßler, Jörg Widmer, Michael Käsemann, Martin Mauve, and Hannes Hartenstein. Contention-Based Forwarding for Mobile Ad-Hoc Networks. *Elsevier's Ad Hoc Networks*, 1(4):351–369, 2003. (Cited on pages 64, 133, 177, and 213.)

[135⋆] Holger Füßler, Jörg Widmer, Martin Mauve, and Hannes Hartenstein. A Novel Forwarding Paradigm for Position-Based Routing (with Implicit Addressing). In *Proceedings of the IEEE 18th Annual Workshop on Computer Communications (CCW 2003)*, pages 194–200, Dana Point, CA, October 2003. (Cited on pages 44, 64, and 213.)

[136] K. Gabriel and R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969. (Cited on page 51.)

[137] Matthew S. Gast. *802.11 Wireless Networks—The Definitive Guide*. O'Reilly, Sebastopol, CA, 2002. (Cited on page 25.)

[138] Silvia Giordano and Maher Hamdi. Mobility Management: The Virtual Home Region. Technical Report SSC/1999/037, EPFL-ICA, October 1999. (Cited on pages 48, 83, and 92.)

[139] Silvia Giordano, Ivan Stojmenovic, and Ljubica Blažević. Position-Based Routing Algorithms for AdHoc Networks: A Taxonomy. *Ad Hoc Wireless Networking*, November 2003. (Cited on page 53.)

[140] Winfried Gleißner and Herbert Zeitler. The Reuleaux Triangle and Its Center of Mass. *Results in Mathematics*, 37:335–344, 2000. (Cited on page 101.)

[141] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics*. Addison-Wesley Professional, 2nd edition, 1994. (Cited on pages 6 and 28.)

[142] Matthias Grossglauser and Martin Vetterli. Locating Nodes with EASE: Last Encounter Routing in Ad Hoc Networks through Mobility Diffusion. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*, San Francisco, CA, March 2003. (Cited on page 49.)

[143] Piyush Gupta, Robert Gray, and P. R. Kumar. An Experimental Scaling Law for Ad Hoc Networks. Technical report, University of Illinois at Urbana-Champaign, 2001. (Cited on pages 6, 57, and 169.)

[144] Piyush Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory*, 46(2):388–404, March 2000. (Cited on pages 6, 57, and 169.)

[145] Andrei Gurtov and Sally Floyd. Modeling wireless links for transport protocols. *ACM SIGCOMM Computer Communication Review (CCR)*, 34(2):85–96, 2004. (Cited on page 57.)

[146] Zygmunt J. Haas and Marc R. Haas. The Performance of Query Control Schemes for the Zone Routing Protocol. *IEEE/ACM Transactions on Networking (TON)*, 9(4):427–438, August 2001. (Cited on page 44.)

[147] Zygmunt J. Haas and Ben Liang. Ad Hoc mobility management with uniform quorum systems. *IEEE/ACM Transactions on Networking (TON)*, 7(2):228–240, April 1999. (Cited on page 48.)

[148] Hannes Hartenstein, Bernd Bochow, André Ebner, Matthias Lott, Markus Radimirsch, and Dieter Vollmer. Position-aware ad hoc wireless networks for inter-vehicle communications: The FleetNet project. In *Proceedings of the Second ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '01)*, Long Beach, California, October 2001. (Cited on page 93.)

[149⋆] Hannes Hartenstein, Holger Füßler, Martin Mauve, and Walter Franz. Simulation Results and Proof-of-Concept Implementation of the FleetNet Position-Based Router. In *Proceedings of the IFIP-TC6 8th International Conference on Personal Wireless Communications (PWC '03)*, pages 192–197, Venice, Italy, September 2003. (Cited on pages 157, 169, 187, 195, and 213.)

[150] Bernhard Hechenleitner and Karl Entacher. On Shortcomings of the ns-2 Random Number Generator. In *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference*, pages 71–77, San Antonio, TX, January 2002. (Cited on page 60.)

[151] John Heidemann, Nirupama Bulusu, Jeremy Elson, Chalermek Intanagonwiwat, Kun-Chan Lan, Ya Xu, Wei Ye, Deborah Estrin, and Ramesh Govindan. Effects of Detail in Wireless Network Simulation. In *Proceedings of SCS Multiconference on Distributed Systems*, pages 3–11, Phoenix, Arizona, January 2001. (Cited on page 60.)

[152] Marc Heissenbüttel and Torsten Braun. BLR: Beacon-Less Routing Algorithm for Mobile Ad-Hoc Networks. *Elsevier Computer Communications*, 27(11):1076–1086, 2004. (Cited on page 134.)

[153] Dirk Helbing. Traffic and Related Self-Driven Many-Particle Systems. *Reviews of Modern Physics*, 73(4):1067–1141, 2001. (Cited on page 143.)

[154] Jeffrey Hightower and Gaetano Borriello. A Survey and Taxonomy of Location Systems for Ubiquitous Computing. Technical Report 01-08-03, University of Washington, Department of Computer Science and Engineering, 2001. (Cited on page 32.)

[155] Gavin Holland and Nitin H. Vaidya. Analysis of TCP Performance over Mobile Ad Hoc Networks. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, Seattle, Washington, August 1999. (Cited on page 56.)

[156] Ting-Chao Hou and Victor O.K. Li. Transmission Range Control in Multihop Packet Radio Networks. *IEEE Transactions on Communications*, 34(1):38–44, January 1986. (Cited on page 50.)

[157] Pai-Hsiang Hsiao. Geographical Region Summary Service for Geographical Routing. *ACM Sigmobile Mobile Computing and Communications Review (MC2R)*, 5(4):25–39, October 2001. (Cited on pages 47 and 48.)

[158] Yih-Chun Hu and David B. Johnson. Caching Strategies in On-Demand Routing Protocols for Wireless Ad Hoc Networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 231–242, Boston, Massachusetts, August 2000. (Cited on page 44.)

[159] Yih-Chun Hu and David B. Johnson. Implicit Source Routing for On-Demand Ad Hoc Network Routing. In *Proceedings of the Second ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '01)*, pages 1–10, Long Beach, California, October 2001. (Cited on page 44.)

[160] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. In *Proceedings of the Eighth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '02)*, pages 12–23, Atlanta, Georgia, September 2002. (Cited on page 44.)

[161] Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model (ISO/IEC 7498-1:19940, 1994. (Cited on page 21.)

[162] Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, John Heidemann, and Fabio Silva. Directed Diffusion for Wireless Sensor Networking. *IEEE/ACM Transactions on Networking (TON)*, 11(1):2–16, February 2003. (Cited on page 6.)

[163] Kyle Jamieson, Brett Hull, Allen Miu, and Hari Balakrishnan. Understanding the real-world performance of carrier sense. In *Proc. of the 2005 ACM SIGCOMM Workshop on Experimental approaches to wireless network design and analysis (E-WIND 2005)*, pages 52–57, Philadelphia, PA, August 2005. (Cited on page 208.)

[164] Ping Ji, Zihui Ge, Jim Kurose, and Don Towsley. A Comparison of Hard-state and Soft-state Signaling Protocols. In *Proceedings of ACM SIGCOMM 2003 Conference on Computer Communications*, pages 251–262, Karlsruhe, Germany, August 2003. (Cited on page 33.)

[165] David B. Johnson and David A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In Tomasz Imielinski and Hank Korth, editors, *Mobile Computing*, volume 353, pages 153–181. Kluwer Academic Publishers, 1996. (Cited on pages 9, 43, 64, 70, 93, 108, 118, 148, and 173.)

[166] John Jubin and Janet D. Tornow. The DARPA Packet Radio Network Protocols. *Proceedings of IEEE*, 75(1):21–32, January 1987. (Cited on page 5.)

[167] Elliot B. Kaplan. *Understanding GPS*. Artech House, 1996. (Cited on page 32.)

[168] Phil Karn. MACA - A New Channel Access Method for Packet Radio. In *Proceedings of the 9th ARRL/CRRL Amateur Radio Computer Networking Conference*, pages 134–140, September 1990. (Cited on pages 24 and 103.)

[169] Brad N. Karp. *Geographic Routing for Wireless Networks*. PhD thesis, Harvard University, October 2000. (Cited on pages 51 and 206.)

[170] Brad N. Karp. Challenges in Geographic Routing: Sparse Networks, Obstacles, and Traffic Provisioning, May 2001. (Cited on pages 52, 117, and 168.)

[171] Brad N. Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 243–254, Boston, Massachusetts, August 2000. (Cited on pages 32, 51, 52, 54, 64, 70, 71, 72, 76, 83, 94, 108, 111, 115, 117, 134, 148, 149, and 206.)

[172′] Wolfgang Kieß. Hierarchical Location Service for Mobile Ad-hoc Networks. Diplomarbeit, Department of Computer Science, University of Mannheim, 2003. (Cited on pages 64, 92, and 213.)

[173⋆] Wolfgang Kieß, Holger Füßler, Jörg Widmer, and Martin Mauve. Hierarchical Location Service for Mobile Ad-Hoc Networks. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 8(4):47–58, October 2004. (Cited on pages 64, 92, and 213.)

[174] Yongjin Kim, Jae-Joon Lee, and Ahmed Helmy. Modeling and analyzing the impact of location inconsistencies on geographic routing in wireless networks. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 8(1):48–60, January 2004. (Cited on page 66.)

[175⋆] Thomas King, Holger Füßler, Matthias Transier, and Wolfgang Effelsberg. On the Application of Dead-Reckoning to Position-Based Routing for Vehicular Highway Scenarios. In *Proceedings of CoNEXT 2005*, pages 258–259, Toulouse, France, October 2005. (Cited on pages 139, 154, 182, and 213.)

[176⋆] Thomas King, Holger Füßler, Matthias Transier, and Wolfgang Effelsberg. Dead-Reckoning for Position-Based Forwarding on Highways. In *Proceedings of 3rd International Workshop on Intelligent Transportation*, pages 199–204, Hamburg, Germany, March 2006. (Cited on pages 139, 154, and 213.)

[177′] Ruben Kirsch. Implementation of a Distance-Vector-Based Recovery- Strategy for Position-Based-Routing. Bachelor's thesis, Department of Mathematics and Computer Science, University of Mannheim, January 2007. (Cited on pages 185 and 213.)

[178] Leonard Kleinrock and Fouad A. Tobagi. Packet Switching in Radio Channels: Part I – Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics. *IEEE Transactions on Communications*, 23(12):1400–1416, December 1975. (Cited on pages 23 and 99.)

[179] Young-Bae Ko and Nitin H. Vaidya. Location-Aided Routing (LAR) in Mobile Ad Hoc Networks. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '98)*, pages 66–75, Dallas, Texas, October 1998. (Cited on page 44.)

[180] Young-Bae Ko and Nitin H. Vaidya. Geocasting in Mobile Ad Hoc Networks: Location-Based Multicast Algorithms. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 101–110, New Orleans, LA, February 1999. (Cited on page 6.)

[181] Andrzej Kochut, Arunchandar Vasan, A. Udaya Shankar, and Ashok K. Agrawala. Sniffing Out the Correct Physical Layer Capture Model in 802.11b. In *Proceedings*

of the 12th IEEE International Conference on Networking Protocols (ICNP '04), pages 252–261, Berlin, Germany, October 2004. (Cited on pages 22 and 208.)

[182] Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia. Compass Routing on Geometric Networks. In *Proceedings of the 11th Canadian Conference on Computational Geometry (CCCG 1999)*, pages 51–54, Vancouver, CA, August 1999. (Cited on page 50.)

[183] W. Kronjäger and D. Hermann. Travel time estimation on the base of microscopic traffic flow simulation. ITS World Congress, 1999. (Cited on page 170.)

[184′] Roland Krüger. Automatisierte Simulationsverteilung für SimpleSim. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, June 2006. (Cited on page 213.)

[185⋆] Roland Krüger, Holger Füßler, Marc Torrent-Moreno, Hannes Hartenstein, and Wolfgang Effelsberg. Statistical Analysis of the FleetNet Highway Movement Patterns. Technical Report TR-2005-004, Department of Mathematics and Computer Science, University of Mannheim, 2005. (Cited on pages 139, 147, and 213.)

[186] Fabian Kuhn, Roger Wattenhofer, Yan Zhang, and Aaron Zollinger. Geometric Ad-Hoc Routing: Of Theory and Practice. In *Proceedings of the 22nd ACM Annual Symposium on Principles of Distributed Computing (PODC '03)*, pages 63–72, Boston, MA, July 2003. (Cited on page 52.)

[187] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Asymptotically optimal geometric mobile ad-hoc routing. In *Proceedings of the Sixth international Workshop on Discrete Algorithms and Methods for Mobile Computing and communications (DIAL-M '02)*, pages 24–33, Atlanta, Georgia, September 2002. (Cited on page 52.)

[188] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Ad-Hoc Networks Beyond Unit Disk Graphs. In *Proceedings of the 1st ACM DIALM-POMC Joint Workshop on Foundations of Mobile Computing (DIALM-POMC '03)*, San Diego, California, September 2003. (Cited on pages 52 and 60.)

[189] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Worst-Case optimal and average-case efficient geometric ad-hoc routing. In *Proceedings of the Fourth ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '03)*, pages 267–278, Annapolis, Maryland, June 2003. (Cited on pages 52 and 117.)

[190] James F. Kurose and Keith W. Ross. *Computer Networking—A Top-Down Approach Featuring the Internet*. Pearson Education, Inc., 2nd edition, 2003. (Cited on pages 23 and 56.)

[191] Taek Jin Kwon and Mario Gerla. Efficient Flooding with Passive Clustering (PC) in Ad Hoc Networks. *ACM SIGCOMM Computer Communication Review (CCR)*, 32(1):44–56, January 2002. (Cited on page 40.)

[192′] Michael Käsemann. Beaconless Position-Based Routing for Mobile Ad-Hoc Networks. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, February 2003. (Cited on pages 44, 64, 109, and 213.)

[193⋆] Michael Käsemann, Holger Füßler, Hannes Hartenstein, and Martin Mauve. A Reactive Location Service for Mobile Ad Hoc Networks. Technical Report TR-02-014, Department of Computer Science, University of Mannheim, November 2002. (Cited on pages 64, 157, 170, 189, and 213.)

[194⋆] Michael Käsemann, Hannes Hartenstein, Holger Füßler, and Martin Mauve. A Simulation Study of a Location Service for Position-Based Routing in Mobile Ad Hoc Networks. Technical Report TR-07-002, Department of Computer Science, University of Mannheim, 2002. (Cited on pages 47, 64, 76, and 213.)

[195⋆] Michael Käsemann, Hannes Hartenstein, Holger Füßler, and Martin Mauve. Analysis of a Location Service for Position-Based Routing in Mobile Ad Hoc Networks. In *Proceedings of the 1st German Workshop on Mobile Ad-Hoc Networking (WMAN 2002)*, GI – Lecture Notes in Informatics, pages 121–133, Ulm, Germany, March 2002. (Cited on pages 47, 64, and 213.)

[196′] Jan Kästle. Contention-Based Forwarding for Uni- and Multicast in Sensor Networks. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, September 2005. (Cited on pages 134 and 213.)

[197] Houda Labiod and Hasnaa Moustafa. The Source Routing-based Multicast Protocol for Mobile Ad Hoc Networks (SRMP), November 2001. (Cited on page 44.)

[198′] Thomas Lang. Location Based Multicast for Wireless Networks. Diplomarbeit, Department of Computer Science, University of Mannheim, 2002. (Cited on page 213.)

[199] Averill M. Law and W. David Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, 3rd edition, 2000. (Cited on page 58.)

[200′] Angelika Leibscher. Recovery-Strategies for Beaconless, Position-Based Routing in Mobile Ad-hoc Networks. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, March 2004. (Cited on pages 64, 117, 177, and 213.)

[201⋆] Angelika Leibscher, Holger Füßler, Jörg Widmer, and Wolfgang Effelsberg. Contention-Based Distance-Vector Routing for Mobile Ad-Hoc Networks. Technical report, Department of Computer Science, University of Mannheim, 2004. (Cited on pages 64 and 213.)

[202] Angelika Leibscher, Jörg Widmer, Matthias Transier, and Wolfgang Effelsberg. Contention-Based Distance-Vector Routing (CBDV) for Mobile Ad-Hoc Networks. In *12th IEEE International Conference on Networking Protocols—Student Poster Session(ICNP '04)*, Berlin, Germany, October 2004. (Cited on pages 64, 177, and 186.)

[203] Jinyang Li, Charles Blake, Douglas S. J. DeCouto, Hu Imm Lee, and Robert Morris. Capacity of Ad Hoc Wireless Networks. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '01)*, pages 61–69, Rome, Italy, July 2001. (Cited on pages 58, 169, and 197.)

[204] Jinyang Li, John Jannotti, Douglas S. J. DeCouto, David R. Karger, and Robert Morris. A Scalable Location Service for Geographic Ad Hoc Routing. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 120–130, Boston, Massachusetts, August 2000. (Cited on pages 45, 46, 64, 70, 71, and 83.)

[205] Xu Lin, Mouhsine Lakshdisi, and Ivan Stojmenovic. Location Based Localized Alternate, Disjoint, Multi-path and Component Routing Schemes for Wireless Networks (Poster Abstract). In *Proceedings of the Second ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '01)*, Long Beach, California, October 2001. (Cited on page 53.)

[206] Xu Lin and Ivan Stojmenovic. Location based localized alternate, disjoint and multipath routing algorithms for wireless networks. *Journal of Parallel and Distributed Computing*, 63(1):22–32, 2003. (Cited on pages 53 and 117.)

[207] Benyuan Liu, Zhen Liu, and Don Towsley. On the Capacity of Hybrid Wireless Networks. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*, San Francisco, CA, March 2003. (Cited on pages 186 and 209.)

[208′] Christian Lochert. Ad Hoc Routing für die Kommunikation zwischen Fahrzeugen in Stadtszenarien. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, 2003. (Cited on pages 44, 139, 174, 175, 179, 181, 182, 184, and 213.)

[209⋆] Christian Lochert, Holger Füßler, Martin Mauve, and Hannes Hartenstein. Geographic Routing in City Scenarios. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 9(1):69–72, January 2005. (Cited on pages 44, 139, 174, 184, and 213.)

[210⋆] Christian Lochert, Hannes Hartenstein, Jing Tian, Holger Füßler, Dagmar Herrmann, and Martin Mauve. A Routing Strategy for Vehicular Ad Hoc Networks in City Environments. In *Proc. of IEEE Intelligent Vehicles Symposium (IV2003)*, pages 156–161, Columbus, OH, June 2003. (Cited on pages 44, 117, 139, 173, 179, 180, 181, 182, 183, 184, and 213.)

[211] Matthias Lott, Rüdiger Halfmann, Egon Schulz, Michael Meincke, Maria Dolores Perez Guirao, and Klaus Jobmann. Data Link Control. In Walter Franz, Hannes Hartenstein, and Martin Mauve, editors, *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles—The FleetNet Project*, pages 29–82. Universitätsverlag Karlsruhe, Karlsruhe, Germany, November 2005. (Cited on page 27.)

[212] Henrik Lundgren, Erik Nordström, and Christian Tschudin. Coping with communication gray zones in IEEE 802.11b based ad hoc networks. In *Proceedings of the Fifth ACM international Workshop on Wireless mobile multimedia*, pages 49–55, Atlanta, GA, September 2002. (Cited on pages 42, 60, 164, and 208.)

[213] Jun Luo, Jean-Pierre Hubaux, and Patrick Th. Eugster. PAN: Providing Relieable Storage in Mobile Ad Hoc Networks with Probabilistic Quorum Systems. In *Proceedings of the Fourth ACM international Symposium on Mobile and Ad Hoc Networking*

& computing (MobiHoc '03), pages 1–12, Annapolis, Maryland, June 2003. (Cited on page 48.)

[214] G. Malkin. RIP Version 2. RFC 2453, `http://www.ietf.org/rfc2453.txt`, 1998. Status: STANDARD. (Cited on page 29.)

[215] David A. Maltz, Josh Broch, Jorjeta G. Jetcheva, and David B. Johnson. The Effects of On-Demand Behavior in Routing Protocols for Multi-Hop Wireless Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications (special issue on mobile and wireless networks)*, 17(8):1439–1453, August 1999. (Cited on pages 44, 56, and 169.)

[216] David A. Maltz, Josh Broch, and David B. Johnson. Experiences designing and building a multi-hop wireless ad hoc network testbed. Technical Report CMU-CS-99-116, School of Computer Science, Carnegie Mellon University, 1999. (Cited on pages 44 and 169.)

[217] Mahesh K. Marina and Samir R. Das. Routing performance in the presence of unidirectional links in multihop wireless networks. In *Proceedings of the Third ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '02)*, pages 12–23, Lausanne, Switzerland, June 2002. (Cited on page 42.)

[218] Makoto Matsumoto and Takuji Nishimura. Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30, January 1998. (Cited on page 104.)

[219⋆] Martin Mauve, Holger Füßler, Jörg Widmer, and Thomas Lang. Position-Based Multicast Routing for Mobile Ad-Hoc Networks. Technical Report TR-03-004, Department of Computer Science, University of Mannheim, 2003. (Cited on pages 45 and 213.)

[220⋆] Martin Mauve, Holger Füßler, Jörg Widmer, and Thomas Lang. Position-based multicast routing for mobile Ad-hoc networks. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 7(3):53–55, July 2003. (Cited on page 213.)

[221⋆] Martin Mauve, Holger Füßler, Jörg Widmer, and Thomas Lang. Poster: Position-Based Multicast Routing for Mobile Ad-Hoc Networks. In *Proceedings of the Fourth ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '03, electronic edition)*, Annapolis, Maryland, June 2003. (Cited on page 213.)

[222⋆] Martin Mauve, Hannes Hartenstein, Holger Füßler, Jörg Widmer, and Wolfgang Effelsberg. Positionsbasiertes Routing für die Kommunikation zwischen Fahrzeugen. *it + ti*, 44(5):278–286, October 2002. (Cited on pages 139 and 213.)

[223] Martin Mauve, Jörg Widmer, and Hannes Hartenstein. A Survey on Position-Based Routing in Mobile Ad-Hoc Networks. *IEEE Network*, 15(6):30–39, November/December 2001. (Cited on pages 48, 50, and 53.)

[224] Michael Mitzenmacher. Compressed Bloom Filters. In *Proceedings of the 20th Annual ACM Symposium on Principles of Distributed Computing*, pages 144–150, Newport, Rhode Island, USA, August 2001. (Cited on page 47.)

[225] Robert Morris, John Jannotti, Frans Kaashoek, Jinyang Li, and Douglas S. J. DeCouto. CarNet: A Scalable Ad Hoc Wireless Network System. In *Proceedings of the 9th ACM SIGOPS European workshop: Beyond the PC: New Challenges for the Operating System*, Kolding, Denmark, September 2000. (Cited on page 93.)

[226] J. Moy. OSPF Version 2. RFC 2328, `http://www.ietf.org/rfc2328.txt`, 1998. Status: STANDARD. (Cited on page 28.)

[227′] Michael Möske. Real-World Evaluation of a Vehicular Ad Hoc Network using Position-Based Routing. Diplomarbeit, Department of Computer Science, University of Mannheim, 2003. (Cited on pages 195 and 213.)

[228⋆] Michael Möske, Holger Füßler, Hannes Hartenstein, and Walter Franz. Performance Measurements of a Vehicular Ad Hoc Network. In *Proceedings of the IEEE Vehicular Technology Conference (VTC'04 Spring)*, Milan, Italy, May 2004. (Cited on pages 187, 195, and 213.)

[229] Julio C. Navas and Tomasz Imielinski. GeoCast – Geographic Addressing and Routing. In *Proceedings of the Third Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '97)*, pages 66–76, Budapest, Hungary, September 1997. (Cited on page 6.)

[230] Randolph Nelson and Leonard Kleinrock. The Spatial Capacity of a Slotted ALOHA Multihop Packet Radio Network with Capture. *IEEE Transactions on Communications*, 32(6):684–694, June 1984. (Cited on page 50.)

[231] Sze-Yao Ni, Tseng Yu-Chee, Chen Yuh-Shyan, and Sheu Jang-Ping. The Broadcast Storm Problem in a Mobile Ad Hoc Network. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, pages 151–162, Seattle, Washington, August 1999. (Cited on pages 39, 64, 66, 69, 70, and 169.)

[232] Jörg Nonnenmacher and Ernst W. Biersack. Scalable Feedback for Large Groups. *IEEE/ACM Transactions on Networking (TON)*, 7(3):375–386, 1999. (Cited on pages 96 and 134.)

[233′] Arndt Oberhoeffken. MIRP - Map Information Routing Protocol for Mobile Ad-Hoc Networks. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, 2003. (Cited on pages 173, 184, and 213.)

[234] R. Ogier, F. Templin, and M. Lewis. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF). RFC 3684, `http://www.ietf.org/rfc3684.txt`, 2004. Status: EXPERIMENTAL. (Cited on page 45.)

[235] Richard G. Ogier, Fred L. Templin, Bhargav Bellur, and Mark G. Lewis. Topology Broadcast Based on Reverse-Path Forwarding (TBRPF), November 2001. (Cited on page 45.)

[236] Evgeny Osipov and Christian Tschudin. A Path Density Protocol for MANETs. In *Proceedings of IEEE ICPS Workshop on Multi-Hop Ad Hoc Networks: From Theory to Reality (REALMAN '05)*, Santorini, Greece, July 2005. (Cited on pages 57 and 208.)

[237] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes*. WCB/McGraw-Hill, 3rd edition, 1991. (Cited on page 211.)

[238] Krzysztof Pawlikowski, Hae-Duck Joshua Jeong, and Jong-Suk Ruth Lee. On Credibility of Simulation Studies of Telecommunications Networks. *IEEE Communications Magazine*, 40(1):132–139, January 2002. (Cited on page 60.)

[239] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, `http://www.ietf.org/rfc3561.txt`, 2003. Status: EXPERIMENTAL. (Cited on page 42.)

[240] Charles E. Perkins and Pravin Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV). In *Proceedings of the Conference on Communications architectures, protocols and applications (SIGCOMM '94)*, London, United Kingdom, August 1994. (Cited on pages 33, 41, and 54.)

[241] Charles E. Perkins and Elizabeth M. Royer. Ad-Hoc On-Demand Distance Vector Routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 90–100, New Orleans, LA, February 1999. (Cited on pages 93, 118, and 175.)

[242] J. Postel. DoD standard Internet Protocol. RFC 760, `http://www.ietf.org/rfc760.txt`, 1980. Status: UNKNOWN. (Cited on pages 189 and 191.)

[243] J. Postel. User Datagram Protocol. RFC 768, `http://www.ietf.org/rfc768.txt`, 1980. Status: STANDARD. (Cited on page 191.)

[244] J. Postel. Internet Protocol. RFC 791, `http://www.ietf.org/rfc791.txt`, 1981. Status: STANDARD. (Cited on page 29.)

[245] J. Postel. Transmission Control Protocol. RFC 793, `http://www.ietf.org/rfc793.txt`, 1981. Status: STANDARD. (Cited on pages 56 and 191.)

[246] Radio Equipment and Systems (RES): High Performance Radio Local Area Network (HIPERLAN) Type 1: Functional Specification, October 1996. (Cited on page 25.)

[247] Jyoti Raju and J. J. Garcia-Luna-Aceves. Scenario-based Comparison of Source-Tracing and Dynamic Source Routing Protocols for Ad Hoc Networks. *Computer Communication Review*, 31(5):70–81, October 2001. (Cited on page 44.)

[248] Hari Rangarajan and J. J. Garcia-Luna-Aceves. Using labeled paths for loop-free on-demand routing in ad hoc networks. In *Proceedings of the 5th ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '04)*, pages 43–54, Tokyo, Japan, June 2004. (Cited on page 44.)

[249] Theodore S. Rappaport. *Wireless Communications - Principle and Practice*. Prentice Hall PTR, 1996. (Cited on pages 21, 60, and 163.)

[250] Sylvia Ratnasamy, Brad N. Karp, Li Yin, Fang Yu, Deborah Estrin, Ramesh Govindan, and Scott Shenker. GHT: a geographic hash table for data-centric storage. In *Proceedings of the First ACM international Workshop on Wireless sensor Networks and applications (WSNA 2002)*, pages 78–87, Atlanta, Georgia, September 2002. (Cited on page 49.)

[251] Y. Rekhter, B. Moskowitz, D. Karrenberg, and G. de Groot. Address Allocation for Private Internets. RFC 1597, `http://www.ietf.org/rfc1597.txt`, 1994. Status: INFORMATIONAL. (Cited on page 191.)

[252] Lawrence G. Roberts. Extension of Packet Communication Technology to a Hand-Held Personal Terminal. In *Proceedings of the Spring Joint Computer Conference*, pages 295–298, Atlantic City, New Jersey, May 1972. (Cited on page 23.)

[253] Christian Rohner, Erik Nordström, Per Gunningberg, and Christian Tschudin. Interactions between TCP, UDP and Routing Protocols in Wireless Multi-hop Ad hoc Networks. In *Proceedings of IEEE ICPS Workshop on Multi-Hop Ad Hoc Networks: From Theory to Reality (REALMAN '05)*, Santorini, Greece, July 2005. (Cited on page 57.)

[254] Sheldon M. Ross. *Simulation (Statistical Modeling & Decision Science)*. Academic Press, 2nd edition, 1996. (Cited on page 58.)

[255] Elizabeth M. Royer and Charles E. Perkins. Multicast Operation of the Ad-hoc On-Demand Distance Vector Routing Protocol. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, pages 207–218, Seattle, Washington, August 1999. (Cited on page 42.)

[256] Elizabeth M. Royer and Charles E. Perkins. An Implementation Study of the AODV Routing Protocol. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '00)*, Chicago, IL, September 2000. (Cited on page 42.)

[257] Elizabeth M. Royer and Chai-Keong Toh. A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. *IEEE Personal Communications*, pages 46–55, April 1999. (Cited on page 45.)

[258′] Björn Scheuermann. Dreidimensionale Visualisierung von Simulationsdaten Mobiler Ad-Hoc-Netzwerke. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, 2004. (Cited on pages 59 and 213.)

[259⋆] Björn Scheuermann, Holger Füßler, Matthias Transier, Marcel Busse, Martin Mauve, and Wolfgang Effelsberg. Huginn: A 3D Visualizer for Wireless ns-2 Traces. In *Proceedings of the Eighth ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '05)*, pages 134–150, Montreal, Canada, October 2005. (Cited on pages 59 and 213.)

[260⋆] Björn Scheuermann, Holger Füßler, Matthias Transier, Marcel Busse, Martin Mauve, and Wolfgang Effelsberg. Huginn: A 3D Visualizer for Wireless ns-2 Traces. Technical Report TR-2005-003, Department of Mathematics and Computer Science, University of Mannheim, June 2005. (Cited on pages 59 and 213.)

[261⋆] Björn Scheuermann, Holger Füßler, Matthias Transier, Martin Mauve, and Wolfgang Effelsberg. MobiCom Demo: Visualizing Wireless ns-2 Traces in 3D. In *Proceedings of the Eleventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '05)*, Cologne, Germany, August/September 2005. (Cited on pages 59 and 213.)

[262⋆] Felix Schmidt-Eisenlohr, Marc Torrent-Moreno, Holger Füßler, and Hannes Hartenstein. Packet Forwarding in VANETs, the Complete Set of Results. Technical Report TR-2006-02, University of Karlsruhe, January 2006. (Cited on pages 136, 162, 163, 179, and 213.)

[263⋆] Felix Schmidt-Eisenlohr, Marc Torrent-Moreno, Holger Füßler, and Hannes Hartenstein. Effects of a Realistic Channel Model on Packet Forwarding in Vehicular Ad Hoc Networks. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '06)*, Las Vegas, NV, April 2006, to appear. (Cited on pages 136, 139, 162, 163, 179, and 213.)

[264] Ralf Schmitz. *Advanced Transport Protocols for Mobile Ad Hoc Networks.* PhD thesis, University of Mannheim, 2006. (Cited on pages 57 and 60.)

[265⋆] Sascha Schnaufer, Holger Füßler, Matthias Transier, and Wolfgang Effelsberg. MobiSys Poster: SLOPE: A System for rapid Deployment of VANET Communication Protocols. In *The Fourth International Conference on Mobile Systems, Applications, and Services*, Uppsala, Sweden, June 2006. (Cited on pages 187, 209, and 213.)

[266⋆] Sascha Schnaufer, Holger Füßler, Matthias Transier, and Wolfgang Effelsberg. Vehicular Ad-Hoc Networks: Single-Hop Broadcast is not enough. In *Proceedings of 3rd International Workshop on Intelligent Transportation*, pages 49–54, Hamburg, Germany, March 2006. (Cited on pages 17, 187, 203, 209, and 213.)

[267] Christoph Schneeweiß. *Distributed Decision Making.* Springer, Berlin, 2nd edition, March 2003. (Cited on page 35.)

[268] Christian Schwingenschlögl and Timo Kosch. Geocast enhancements of AODV for vehicular networks. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, 6(3):96–97, July 2002. (Cited on page 42.)

[269] Prasun Sinha, Narayanan Venkitaraman, Raghupathy Sivakumar, and Vaduvur Bharghavan. WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, pages 231–241, Seattle, Washington, August 1999. (Cited on page 57.)

[270] Prasun Sinha, Narayanan Venkitaraman, Raghupathy Sivakumar, and Vaduvur Bharghavan. WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks. *Wireless Networks*, 8(2):301–316, 2002. (Cited on page 57.)

[271] Smart Networks. *DaimlerChrysler Research HighTechReport*, (1):38–41, 2003. (Cited on page 188.)

[272] Dava Sobel. *Longitude.* Fourth Estate, 1998. (Cited on page 31.)

[273] Marcelo Spohn and J. J. Garcia-Luna-Aceves. Neighborhood Aware Source Routing. In *Proceedings of the Second ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '01)*, pages 11–21, Long Beach, California, October 2001. (Cited on page 44.)

[274] Charles E. Spurgeon. *Ethernet: The Definitive Guide.* O'Reilly, Sebastopol, CA, 2000. (Cited on page 25.)

[275] W. Richard Stevens. *TCP/IP Illustrated*, volume 1. Addison Wesley Longman, 1994. (Cited on pages 56, 169, and 191.)

[276] W. Richard Stevens. *UNIX Network Programming*, volume 1. Prentice Hall, 2nd edition, 1998. (Cited on pages 191 and 194.)

[277] Ivan Stojmenovic and Xu Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless network. *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1023–1032, October 2001. (Cited on page 53.)

[278] Ivan Stojmenovic and Bosko Vukojevic. A routing strategy and quorum based location update scheme for ad hoc wireless networks. Technical Report TR-99-09, Computer Science, SITE, University of Ottawa, September 1999. (Cited on page 48.)

[279] Jonathan Stone, Michael Greenwald, Craig Partridge, and James Hughes. Performance of Checksums and CRC's over Real Data. *IEEE/ACM Transactions on Networking (TON)*, 6(5):529–543, October 1998. (Cited on page 26.)

[280] Bjarne Stroustrup. *The C++ Programming Language*. Addison-Wesley Longman, Berlin, 3rd edition, February 2000. (Cited on page 59.)

[281] Karthikeyan Sundaresan, Vaidyanathan Anantharaman, Hung-Yun Hsieh, and Raghupathy Sivakumar. ATP: A Reliable Transport Protocol for Ad-hoc Networks. In *Proceedings of the Fourth ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '03)*, pages 64–75, Annapolis, Maryland, June 2003. (Cited on page 57.)

[282] Hideaki Takagi and Leonard Kleinrock. Optimal Transmission Ranges for Randomly Distributed Packet Radio Terminals. *IEEE Transactions on Communications*, 32(3):246–257, 1984. (Cited on pages 6, 50, and 97.)

[283] Mineo Takai, Jay Martin, and Rajive Bagrodia. Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks. Technical report, UCLA, 2002. (Cited on page 60.)

[284] Andrew S. Tanenbaum. *Computer Networks*. Addison-Wesley, 4th edition, 2003. (Cited on pages 5, 21, 23, 24, 26, 28, and 37.)

[285′] Lucifer Crispian Tonn. Implementierung und Verifikation von Kanalzugriffsprotokollen für SimpleSim. Bachelor's thesis, Department of Mathematics and Computer Science, University of Mannheim, August 2005. (Cited on page 213.)

[286] Godfried Toussaint. The relative neighborhood graph of a finite planar set. *Pattern Recognition*, 12(4):261–268, 1980. (Cited on page 52.)

[287′] Matthias Transier. Dynamic Load Balancing for Position-Based Routing Algorithms. Diplomarbeit, University of Mannheim, July 2002. (Cited on page 213.)

[288⋆] Matthias Transier, Holger Füßler, Thomas Butter, and Wolfgang Effelsberg. Implementing Scalable Position-Based Multicast for the Linux Kernel. In *Proceedings of the 2nd German Workshop on Mobile Ad-Hoc Networking (WMAN '04)*, pages 105–110, Ulm, Germany, September 2004. (Cited on page 213.)

[289⋆] Matthias Transier, Holger Füßler, Martin Mauve, Jörg Widmer, and Wolfgang Effelsberg. Dynamic Load Balancing for Position-Based Routing. In *Proceedings of CoNEXT 2005*, pages 290–291, Toulouse, France, October 2005. (Cited on page 213.)

[290⋆] Matthias Transier, Holger Füßler, Jörg Widmer, Martin Mauve, and Wolfgang Effelsberg. A Hierarchical Approach to Position-Based Multicast for Mobile Ad-hoc Networks. Technical Report TR-04-002, Department for Mathematics and Computer Science, University of Mannheim, 2004. (Cited on page 213.)

[291⋆] Matthias Transier, Holger Füßler, Jörg Widmer, Martin Mauve, and Wolfgang Effelsberg. Scalable Position-Based Multicast for Mobile Ad-hoc Networks. In *Proceedings of the First International Workshop on Broadband Wireless Multimedia: Algorithms, Architectures and Applications (BroadWIM 2004)*, San José, CA, October 2004. (Cited on page 213.)

[292⋆] Matthias Transier, Holger Füßler, Jörg Widmer, Martin Mauve, and Wolfgang Effelsberg. A Hierarchical Approach to Position-Based Multicast for Mobile Ad-hoc Networks. *Wireless Networks – The Journal of Mobile Communication, Computation and Information*, 2006. (Cited on page 213.)

[293] Christian Tschudin and Richard Gold. LUNAR: Lightweight Underlay Network Ad-hoc Routing. Technical Report Technical Report 2003-021, Uppsala University, 2003. (Cited on page 44.)

[294] Christian Tschudin, Richard Gold, Olof Rensfelt, and Oskar Wibling. LUNAR: a Lightweight Underlay Network Ad-hoc Routing Protocol and Implementation. In *Proceedings of the International Conference on Next Generation Teletraffic and Wired/Wireless Advanced Networking (NEW2AN'04)*, St. Petersburg, Russia, February 2004. (Cited on page 44.)

[295] Christian Tschudin, Henrik Lundgren, and Erik Nordström. Embedding MANETs in the Real World. In *Proceedings of the IFIP-TC6 8th International Conference on Personal Wireless Communications (PWC '03)*, pages 578–589, Venice, Italy, September 2003. (Cited on pages 3, 6, 60, 202, and 205.)

[296] D. Vollmer, B. Balasubramanian, and E. Siegert. Fahrtsimulation unter realistischen Umfeldbedingungen (in German). VDI-Berichte, 1992. (Cited on page 143.)

[297] Kilian Weniger and Martina Zitterbart. Address Autoconfiguration in Mobile Ad Hoc Networks: Current Approaches and Future Directions. *IEEE Network Magazine*, 18(4):6–11, July/August 2004. (Cited on pages 12 and 38.)

[298] Jörg Widmer. *Equation-Based Congestion Control for Unicast and Multicast Data Streams*. PhD thesis, University of Mannheim, 2003. (Cited on page 134.)

[299] Jörg Widmer and Thomas T. Fuhrmann. Extremum Feedback for Very Large Multicast Groups. In *Proceedings of the Third International Workshop on Networked Group Communication (NGC)*, pages 56–75, London, November 2001. (Cited on page 134.)

[300] Jörg Widmer and Mark Handley. Extending Equation-based Congestion Control to Multicast Applications. In *Proceedings of ACM SIGCOMM 200 Conference on Computer Communications*, pages 275–286, San Diego, CA, August 2001. (Cited on page 134.)

[301⋆] Jörg Widmer, Martin Mauve, Hannes Hartenstein, and Holger Füßler. Position-Based Routing in Ad-Hoc Wireless Networks. In Mohammad Ilyas, editor, *The Handbook of Ad Hoc Wireless Networks*. CRC Press, Boca Raton, FL, U.S.A., December 2002. (Cited on page 213.)

[302] Brad Williams and Tracy Camp. Comparison of Broadcasting Techniques for Mobile Ad Hoc Networks. In *Proceedings of the Third ACM international Symposium on Mobile and Ad Hoc Networking & computing (MobiHoc '02)*, pages 194–205, Lausanne, Switzerland, June 2002. (Cited on pages 40, 64, 67, and 69.)

[303] Lars Wischhof, André Ebner, Hermann Rohling, Matthias Lott, and Rüdiger Halfmann. Adaptive Broadcast for Travel and Traffic Information Distribution Based on Inter-Vehicle Communication. In *Proc. of IEEE Intelligent Vehicles Symposium (IV2003)*, Columbus, OH, June 2003. (Cited on page 18.)

[304] Lars Wischhof, André Ebner, Hermann Rohling, Matthias Lott, and Rüdiger Halfmann. SOTIS - A Self-Organizing Traffic Information System. In *Proceedings of the IEEE 57th Vehicular Technology Conference (VTC'03 Spring)*, Jeju, Korea, April 2003. (Cited on page 18.)

[305] Lars Wischhof, André Ebner, Hermann Rohling, Matthias Lott, and Rüdiger Halfmann. Self-Organizing Traffic Information System (SOTIS). In Walter Franz, Hannes Hartenstein, and Martin Mauve, editors, *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles—The FleetNet Project*, pages 233–258. Universitätsverlag Karlsruhe, Karlsruhe, Germany, November 2005. (Cited on page 18.)

[306] Yuan Xue, Baochun Li, and Klara Nahrstedt. A Scalable Location Management Scheme in Mobile Ad-Hoc Networks. In *Proceedings of the 26th Annual IEEE Conference on Local Computer Networks (LCN'01)*, pages 102–111, Tampa, Florida, November 2001. (Cited on page 48.)

[307] Jungkeun Yoon, Mingyan Liu, and Brian Noble. Random Waypoint Considered Harmful. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*, San Francisco, CA, March 2003. (Cited on pages 9 and 59.)

[308] Michele Zorzi and Ramesh R. Rao. Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Energy and Latency Performance. *IEEE Transactions on Mobile Computing*, 2(4):349–365, October 2003. (Cited on page 134.)

[309] Michele Zorzi and Ramesh R. Rao. Geographic Random Forwarding (GeRaF) for ad hoc and sensor networks: multihop performance. *IEEE Transactions on Mobile Computing*, 2(4):337–348, October 2003. (Cited on page 134.)

[310] F. Zurheide. Dynamische Verkehrsumlegung in einer mikroskopischen Simulation. Master's thesis, Fachhochschule Braunschweig/Wolfenbüttel, 1999. (Cited on page 170.)