Jarmo Prokkola

ENHANCING THE PERFORMANCE OF AD HOC NETWORKING BY LOWER LAYER DESIGN
JARMO PROKKOLA

ENHANCING THE PERFORMANCE OF AD HOC NETWORKING BY LOWER LAYER DESIGN

Academic dissertation to be presented, with the assent of the Faculty of Technology of the University of Oulu, for public defence in Auditorium IT116, Linnanmaa, on December 5th, 2008, at 12 noon

OULUN YLIOPISTO, OULU 2008
Abstract

The research of early ad hoc-like networks, namely multi-hop packet radio networks, was mainly concentrated on lower layers (below network layer). Interestingly, the research of modern ad hoc networks has been mainly restricted to routing protocols. This is understandable, since routing is very challenging in such dynamic networks, but the drawback is that the lower layer models used in the studies are often highly simplified, giving inaccurate or even incorrect results. In addition, modern ad hoc network solutions are usually suboptimal because lower layers, not designed especially for ad hoc networking, are used. Thus, ad hoc networking performance, in general, can be notably enhanced by considering also the design of lower layers. The simple deployment and robustness make wireless ad hoc networks attractive for several applications (e.g., military, public authority, peer-to-peer civilian, and sensor networking), but the performance of the current solutions is typically not adequate.

The focus of this work is on the effects of lower layer functionalities on the performance of ad hoc networks, while also taking into account the effects of upper layers (e.g., the effect of application traffic models). A CDMA (Code Division Multiple Access) based dual channel flat ad hoc network solution, incorporating cross-layering between all three lowest layers, is proposed and analyzed. The main element of this is the Bi-Code Channel Access (BCCA) method, in which a common code channel is used for broadcast information (e.g., route discovery), while a receiver-specific code channel is used for all directed transmissions. In addition, a new MAC (Medium Access Control) solution designed for BCCA is presented. Moreover, a novel network layer spreading code distribution (NSCD) method is presented. The advantage of these methods is that they are designed especially to be used in ad hoc networks.

With an extensive set of case studies, it is shown that the presented methods outperform the typically used ad hoc network solutions (based on IEEE 802.11) in different kind of scenarios, environments, modeling concepts, and with different parameters. Detailed simulations are carried out in order to analyze the effects of different features at the lower layers, finding also interesting phenomena and dependencies between different layers. It is also shown that close attention should be paid to lower layer modeling even though the overall network performance would be in focus. In addition, various interesting features and behavior models regarding ad hoc networking are brought up.

Keywords: BC-MAC, BCCA, CDMA, NSCD, Spread spectrum
Tiivistelmä

Ensimmäiset tutkimukset rakenteettomasta (ad hoc) verkoista esiintyvät nimellä monihypypapettiradioverkot, ja ne koskivat pääasiassa verkkokerroksen alapuolella olevia tietoliikennekerrokia, mutta nykyiset tutkimukset ovat kuitenkin keskityneet pääasiassa reititysprotokolliin. Tämä on sikäli ymmärrettävää, että reititys on hyvin haasteellista tämän tyyppisissä dynaamisissa verkoissa, mutta ongelma on, että käytetyt alempien kerrosten mallit ovat usein hyvinkin yksinkertaiset, mikä voi johtaa epätarkoihin tai jopa vääriin tuloksiin. Tämän lisäksi nykyiset ehdotetut rakenteettomien verkkojen ratkaisut ovat usein tehottomia, sillä käytettyjä alempien kerrosten mallit ovat usein hyvin yksinkertaiset, mikä voi johtaa epätarkoihin tai jopa vääriin tuloksiin.

Tämän lisäksi nykyiset ehdotetut rakenteettomien verkkojen suorituskykyä voidaan parantaa huomattavasti kiinnittämällä huomiota alempien kerrosten suunnitteluun. Verkkojen rakenteettomuus on ajatuksena houkutteleva useissa käyttökohteissa (esiintyvät esimerkiksi sotilasympäristöissä, viranomaiskäytössä, käyttäjien välisissä suorissa yhteyksissä ja sensoriverkoissa), mutta suorituskyky ei useinkaan ole riittävällä tasolla käytännön sovelluksiin.

Työssä tutkitaan pääasiassa alempien kerrosten toiminnallisuuden vaikutusta rakenteettomien verkkojen suorituskykyyn ottaen huomioon myös ylemmät kerrokat, kuten sovellustason mallit. Työssä esitellään ja analysoidaan koodijakomonikäyttöön (CDMA, Code Division Multiple Access) perustuva kaksikanavainen tasainen rakenteettoman verkon ratkaisu, jossa käytetään esimerkiksi sotilasympäristöissä, viranomaiskäytössä, käyttäjien välissä suorissa yhteyksissä ja sensoriverkoissa, mutta suorituskyky ei useinkaan ole riittävällä tasolla käytännön sovelluksiin.

Työssä esitellään ja analysoidaan koodijakomonikäyttöön (CDMA, Code Division Multiple Access) perustuva kaksikanavainen tasainen rakenteettoman verkon ratkaisu, jossa käytetään esimerkiksi sotilasympäristöissä, viranomaiskäytössä, käyttäjien välissä suorissa yhteyksissä ja sensoriverkoissa, mutta suorituskyky ei useinkaan ole riittävällä tasolla käytännön sovelluksiin.

Työssä esitellään ja analysoidaan koodijakomonikäyttöön (CDMA, Code Division Multiple Access) perustuva kaksikanavainen tasainen rakenteettoman verkon ratkaisu, jossa käytetään esimerkiksi sotilasympäristöissä, viranomaiskäytössä, käyttäjien välissä suorissa yhteyksissä ja sensoriverkoissa, mutta suorituskyky ei useinkaan ole riittävällä tasolla käytännön sovelluksiin.
Preface

The background work for this thesis was started around the year 2002, when I was working in the Centre for Wireless Communications (CWC) / Telecommunications Laboratory at the University of Oulu. The first publications made by our group were about ad hoc networking performance, showing that ad hoc networks fail easily under a high traffic load, and, thus, there is need for new solutions in order to enhance the performance. Inspired by this, the work on the new lower layer solutions (what this thesis is all about) started. During the CWC period, the work was carried out under the Finnish Software Radio Programme, funded by the Finnish Defense Forces. In 2004, I moved to work in VTT, Valtion Teknillinen Tutkimuskeskus (Technical Research Centre of Finland) in Oulu, where I was able to continue and finalize the work for this thesis. In VTT, the work was mainly carried out under the ITEA Easy Wireless and the CELTIC Easy Wireless 2 projects funded by Tekes (The Finnish Funding Agency for Technology and Innovation).

I thank both CWC and VTT for providing me with the facilities to make this work possible. In particular, the OPNET simulation tool has been essential to this work. Thanks also to the whole staff of CWC and VTT for providing a professional and pleasant atmosphere for research work. Special thanks to Professor Pentti Leppänen, Head of the Telecommunication Laboratory, for highlighting the importance of post graduate studies, and also to my superior in VTT, Dr. Marko Jurvansuu, for encouraging me towards a doctoral degree.

I thank my former colleagues Dr. Timo Bräysy, Mr. Lasse Leppänen, and Mr. Teemu Vanninen for their contributions to the publications we did together on this topic. Also, I thank Mr. Kai Lieska, who provided the connectivity analysis tool during my CWC period.

I am grateful to my thesis supervisors Professor Jari Iinatti and Professor Carlos Pomalaza-Ráez, who have assisted me especially in finalizing the thesis. I thank also Professors Todor Cooklev and Frank H. P. Fitzek for their pre-examination of the thesis and valuable comments.

Finally, I am indebted to my wife Jenny and my family for supporting me and understanding the time that was required to accomplish this.

Oulu, October 20, 2008

Jarmo Prokkola
A list of symbols and abbreviations

- $\lambda$: wavelength
- $\gamma_b$: signal to noise ratio per bit
- $\sigma$: standard deviation
- $\sigma_L$: standard deviation of fading (location variability)
- $\Omega$: relative mobility
- $\Gamma$: normalized network control load
- $B$: receiver bandwidth
- $c(c)$: spreading signal (a chip stream)
- $d_l$: path length (propagation)
- $d_{rt}$: radio range (effective value, not a cut signal)
- $F_{Su}$: node $n$’s fraction of the total generated traffic in the network
- $F_{sf}$: spreading factor
- $G$: offered traffic load (normalized)
- $G_p$: processing gain
- $G_R$: receiver antenna gain
- $G_T$: transmitter antenna gain
- $k$: Boltzmann’s constant ($1.3805 \times 10^{-23}$ J/K)
- $l_{prev_pk}$: the total length (bits) of the previously sent data packet (BC-MAC)
- $l_{pk}$: the total length (bits) of the data packet
- $L$: a common impression for attenuation [dB]
- $L_{fad}$: attenuation due to fading component
- $n$: number of samples
- $N_{bo}$: number of back-off slots
- $N_c$: spreading code length
- $N_{CW_{max}}$: contention window limit (802.11)
- $N_n$: number of nodes in the network
- $N_m$: number of network segments
- $N_{iter}$: number of averaged simulation iterations
- $N_{iter_pk}$: number of simulated packets per iteration
- $N_{reTX}$: number of re-transmissions (maximum)
- $N_{sim_pk}$: number of simulated packets
- $p_{dodge}$: probability of successful dodging
- $p_{CDo}$: probability of dodging period
$p_{\text{CDo}_\text{fail}}$ probability of unsuccessful dodging

$p_{l_c}$ probability of packet loss due to connectivity

$p_{\text{loss}}$ probability of packet loss

$p_{\text{ber}}$ target bit error rate

$p_{\text{dth}}$ power level threshold for detection

$p_{\text{loss_res}}$ packet loss resolution

$P_i$ total interference power

$P_{\text{loss_res}}$ packet loss resolution

$P_{ri}$ the received power from the node $i$

$P_T$ transmission power

$r_{\text{chip}}$ chip rate

$r_{\text{data}}$ data rate

$r_d$ physical layer data rate (bit/s)

$s$ estimator for standard deviation (sample standard deviation)

$t_{\text{ACK}}$ ACK transmission time

$t_{\text{ACK_to}}$ ACK timeout

$t_{\text{data}}$ data packet transmission time

$t_{\text{dodge}}$ dodging period (BC-MAC, CDo)

$t_{\text{prop}}$ propagation delay

$t_{\text{pros}}$ processing time

$t_{\text{Rx/Tx}}$ average Rx/Tx switch time

$T$ receiver effective noise temperature

$\bar{x}$ calculated sample mean

Floor($x$) rounds $x$ to the nearest integer (downwards)

Rand($x$) gives a random number from $[0, x]$

Unif($x$) uniformly distributed random variable from $[0, x]$.

3G Third generation cellular networks

AACAMA Adaptive Acquisition Collision Avoidance Multiple Access

AART AODV Active Route Timeout (a parameter in AODV)

ASIC Application Specific Integrated Circuit

ACK Acknowledgement (packet)

ADC Analog to Digital Converter/Conversion

AFC Automatic Frequency Control

AODV Ad Hoc On-Demand Distance Vector

AP Access Point

ARP Address Resolution Protocol

10
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARQ</td>
<td>Automatic Repeat reQuest</td>
</tr>
<tr>
<td>AWGN</td>
<td>Additive White Gaussian Noise</td>
</tr>
<tr>
<td>BCCA</td>
<td>Bi-Code Channel Access</td>
</tr>
<tr>
<td>BC-MAC</td>
<td>Bi-Code MAC</td>
</tr>
<tr>
<td>BER</td>
<td>Bit Error Ratio</td>
</tr>
<tr>
<td>BPSK</td>
<td>Binary Phase Shift Keying</td>
</tr>
<tr>
<td>BS</td>
<td>Base Station</td>
</tr>
<tr>
<td>CA</td>
<td>Collision Avoidance</td>
</tr>
<tr>
<td>CBR</td>
<td>Constant Bit Rate</td>
</tr>
<tr>
<td>CCA</td>
<td>Clear Channel Assessment</td>
</tr>
<tr>
<td>CCK</td>
<td>Complementary Core Keying</td>
</tr>
<tr>
<td>CDF</td>
<td>Cumulative Distribution Function</td>
</tr>
<tr>
<td>CDMA</td>
<td>Code Division Multiple Access</td>
</tr>
<tr>
<td>CDo</td>
<td>Collision Dodging (a method used in BC-MAC)</td>
</tr>
<tr>
<td>CP</td>
<td>Contention Period</td>
</tr>
<tr>
<td>CPF</td>
<td>Contention Free Period</td>
</tr>
<tr>
<td>CPHP</td>
<td>High Power Cut Propagation (a propagation model)</td>
</tr>
<tr>
<td>CPLP</td>
<td>Low Power Cut Propagation (a propagation model)</td>
</tr>
<tr>
<td>CRC</td>
<td>Cyclic Redundancy Check</td>
</tr>
<tr>
<td>CSMA</td>
<td>Carrier Sense Multiple Access</td>
</tr>
<tr>
<td>CSMA/CA</td>
<td>CSMA with collision avoidance</td>
</tr>
<tr>
<td>C-T</td>
<td>Common-Transmitter (a spreading code protocol)</td>
</tr>
<tr>
<td>DBPSK</td>
<td>Differential Binary Phase Shift Keying</td>
</tr>
<tr>
<td>DC</td>
<td>Dynamic Connections (a session model)</td>
</tr>
<tr>
<td>DCF</td>
<td>Distributed Coordination Function</td>
</tr>
<tr>
<td>DIFS</td>
<td>DCF Inter Frame Space</td>
</tr>
<tr>
<td>DS</td>
<td>Direct Sequence</td>
</tr>
<tr>
<td>DSDV</td>
<td>Destination-Sequenced Distance-Vector (a routing protocol)</td>
</tr>
<tr>
<td>DSR</td>
<td>Dynamic Source Routing</td>
</tr>
<tr>
<td>FEC</td>
<td>Forward Error Correction</td>
</tr>
<tr>
<td>FH</td>
<td>Frequency Hopping</td>
</tr>
<tr>
<td>FNC</td>
<td>Fixed Network Connection (a session model)</td>
</tr>
<tr>
<td>FPL</td>
<td>Forest Propagation Loss (a propagation model)</td>
</tr>
<tr>
<td>FSL</td>
<td>Free Space Loss (a propagation model)</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communications</td>
</tr>
<tr>
<td>HW</td>
<td>Hardware</td>
</tr>
<tr>
<td>IAT</td>
<td>Interarrival Time</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Full Form</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LPD</td>
<td>Low Probability of Detection</td>
</tr>
<tr>
<td>LPI</td>
<td>Low Probability of Interception</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium Access Control</td>
</tr>
<tr>
<td>MACA</td>
<td>Multiple Access with Collision Avoidance</td>
</tr>
<tr>
<td>MAI</td>
<td>Multiple Access Interference</td>
</tr>
<tr>
<td>MANET</td>
<td>Mobile Ad Hoc NETwork</td>
</tr>
<tr>
<td>MC-CDMA</td>
<td>Multi-Carrier Code Division Multiple Access</td>
</tr>
<tr>
<td>MF</td>
<td>Matched Filter</td>
</tr>
<tr>
<td>MIMO</td>
<td>Multiple Input, Multiple Output</td>
</tr>
<tr>
<td>MLV</td>
<td>Mobility, Light Vehicular (a mobility model)</td>
</tr>
<tr>
<td>MOS</td>
<td>Mean Opinion Score</td>
</tr>
<tr>
<td>MP</td>
<td>Mobility, Pedestrian (a mobility model)</td>
</tr>
<tr>
<td>MPDU</td>
<td>MAC Protocol Data Unit</td>
</tr>
<tr>
<td>MV</td>
<td>Mobility, Vehicular (a mobility model)</td>
</tr>
<tr>
<td>NAV</td>
<td>Network Allocation Vector</td>
</tr>
<tr>
<td>np-CSMA</td>
<td>non-persistent CSMA</td>
</tr>
<tr>
<td>NSCD</td>
<td>Network Layer Spreading Code Distribution</td>
</tr>
<tr>
<td>OSI</td>
<td>Open Systems Interconnection</td>
</tr>
<tr>
<td>PCF</td>
<td>Point Coordination Function</td>
</tr>
<tr>
<td>PD</td>
<td>Power Detection (a carrier sensing detection mode)</td>
</tr>
<tr>
<td>PDF</td>
<td>Probability Density Function</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol Data Unit</td>
</tr>
<tr>
<td>PHY</td>
<td>Physical layer</td>
</tr>
<tr>
<td>PLCP</td>
<td>Physical Layer Convergence Protocol (IEEE 802.11)</td>
</tr>
<tr>
<td>PSD</td>
<td>Power and Signal detection (a carrier sensing detection mode)</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>QPSK</td>
<td>Quadrature Phase Shift Keying</td>
</tr>
<tr>
<td>RERR</td>
<td>Route Error (AODV control packet format)</td>
</tr>
<tr>
<td>reTx</td>
<td>Retransmission</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RREP</td>
<td>Route Reply (AODV control packet format)</td>
</tr>
<tr>
<td>RREQ</td>
<td>Route Request (AODV control packet format)</td>
</tr>
<tr>
<td>RWM</td>
<td>Random Waypoint Mobility</td>
</tr>
<tr>
<td>Rx</td>
<td>Reception/Receiver</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>SC</td>
<td>Static Connections (a session model)</td>
</tr>
<tr>
<td>SD</td>
<td>Signal Detection (a carrier sensing detection mode)</td>
</tr>
<tr>
<td>SE</td>
<td>Standard Error</td>
</tr>
<tr>
<td>SFD</td>
<td>Start Frame Delimiter</td>
</tr>
<tr>
<td>SIFS</td>
<td>Short Inter Frame Space</td>
</tr>
<tr>
<td>SINR</td>
<td>Signal to Interference and Noise Ratio</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal to Noise Ratio</td>
</tr>
<tr>
<td>SRMA</td>
<td>Split-channel Reservation Multiple Access</td>
</tr>
<tr>
<td>SS</td>
<td>Spread Spectrum</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>TCP</td>
<td>Transport Control Protocol</td>
</tr>
<tr>
<td>Tx</td>
<td>Transmission/Transmitter</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>UMTS</td>
<td>Universal Mobile Telephone System</td>
</tr>
<tr>
<td>VBR</td>
<td>Variable Bit Rate</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voice over IP</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless LAN</td>
</tr>
<tr>
<td>WMAN</td>
<td>Wireless Metropolitan Area Network</td>
</tr>
</tbody>
</table>
Table of contents

Abstract
Tiivistelmä
Preface
A list of symbols and abbreviations
Table of contents

1 Introduction
1.1 The nature of ad hoc networking
1.1.1 Centralized network structure
1.1.2 Structureless network
1.2 Author’s contributions to the field and motivation
1.3 Thesis outline

2 Ad Hoc networking protocols and problems
2.1 Routing protocols
2.1.1 Ad Hoc On-demand Distance Vector Routing
2.2 Link and physical layer protocols and methods
2.2.1 Channel access: ALOHA and CSMA
2.2.2 Acknowledgements
2.2.3 Collision avoidance with handshaking
2.2.4 IEEE 802.11
2.3 The problem of charge
2.4 The connectivity problem
2.4.1 A tool for analyzing connectivity
2.5 Lower layer functionalities and ad hoc network performance
2.5.1 Operation on a common contention channel

3 BCCA and BC-MAC
3.1 General functionality of BCCA
3.1.1 Spreading code protocols
3.1.2 Functionality of a BCCA based receiver
3.2 Network layer spreading code distribution
3.3 BC-MAC
3.3.1 Dodging collisions with BC-MAC
3.3.2 Cross-layer operation
3.4 Theoretical throughputs of BC-MAC vs. IEEE 802.11
3.5 Related work

4 Performance study
4.1 Modeling the network ................................................................. 75
  4.1.1 Application layer ................................................................. 76
  4.1.2 Session and transport layers .................................................. 77
  4.1.3 Network layer ................................................................. 78
  4.1.4 Link layer (MAC) ............................................................. 79
  4.1.5 Physical layer ................................................................. 80
  4.1.6 Mobility modeling .............................................................. 84
  4.1.7 How to simulate? ............................................................... 85
4.2 Performance metrics ................................................................. 94
4.3 Operation in a conventional access point network scenario .......... 97
4.4 Operation in a special 16-node random grid scenario ................. 102
  4.4.1 A single-hop scenario ......................................................... 103
  4.4.2 A multi-hop scenario ......................................................... 107
4.5 Operation in a typical 20-node ad hoc network scenario ............ 110
  4.5.1 A stationary scenario ......................................................... 110
  4.5.2 A mobile scenario .............................................................. 124
  4.5.3 Detailed performance evaluation in the time domain ........... 131
4.6 The effect of the AART parameter and mobility on the performance ................................................................. 140
4.7 BC-MAC’s dependency on the spreading factor ....................... 145
  4.7.1 Simulation results and analysis ........................................... 146
4.8 Radio channel modeling and its effects on the performance .......... 151
  4.8.1 The effect of radio channel propagation modeling ............... 152
  4.8.2 The effect of fading ............................................................ 164
4.9 Ad hoc network performance under different traffic flows .......... 176
  4.9.1 The effect of data packet size ............................................. 176
  4.9.2 The effect of traffic burstiness .......................................... 184
  4.9.3 The effect of session models .............................................. 190
  4.9.4 Ad hoc network performance under VoIP traffic ............... 196

5 Implementation issues .................................................................. 205
5.1 NSCD ....................................................................................... 205
  5.1.1 Dependencies ..................................................................... 205
  5.1.2 Implementation requirements ............................................ 206
5.2 BC-MAC .................................................................................. 207
  5.2.1 Dependencies ..................................................................... 207
  5.2.2 Implementation Requirements ........................................... 207
5.3 BCCA ..................................................................................... 208
1 Introduction

This thesis considers quite widely all three of the lowest communication layers: the physical layer (PHY), the link layer, and the network layer. Therefore, the first part of this thesis (Chapter 1 and partially Chapter 2) gives an introduction to the methods and basic concepts in these layers relevant to this thesis.

1.1 The nature of ad hoc networking

In this section, the special nature of wireless ad hoc networks, as compared to the traditional wireless networks, is discussed briefly.

1.1.1 Centralized network structure

The centralized network structure is the typical structure in wireless networks. In this, a base station (BS) (or access point, AP) serves the wireless network users in its cell area (Fig. 1). Naturally, these kinds of networks have their own design principles and problems (Tabbane 2000), but they also have several advantages. The BS controls the radio resources, so there is no need for contention, and, hence, there will be no data collisions and the network access is robust. The controlled resource allocation allows easy Quality of Service (QoS) management in the network, enabling, e.g., fixed bandwidth allocations for users.

![Fig. 1. A traditional base station network.](image)

Tabbane 2000
In cellular networks, like GSM (Global System for Mobile Communications) and 3G/UMTS (Third Generation, Universal Mobile Telephone System), the radio resource control is strict and the network side has full control over the radio access (Halonen et al. 2002, Holma & Toskala 2002). The users access the network via a common shared control channel, after which the BS can allocate dedicated physical channels for data (or voice) communications. In other words, even in the case of dynamic data communications (e.g., web browsing), fixed resources are typically allocated to a single user (fixed channel access). Of course, this is not always the case, like, e.g., in HSDPA (High Speed Downlink Packet Access) the user is in a shared physical channel, but still the channel needs to be set up prior to communications, and in that way, the BS has control over the access. Also, the cellular systems incorporate full duplex communications, which makes the network operation very stable even under a heavy bi-directional traffic load. With duplexing, it is not possible that the both communicating ends (in this case, the BS and the user station) would try to transmit simultaneously in the same channel (Tx/Tx collision). Full mobility is also easily supported with efficient mobility management and handover algorithms, since the network side has typically full control over all the base stations.

In WLAN (Wireless Local Area Network) systems, the situation is different as there is typically only a single communication channel in a single cell. Also, there is no duplexing. As a consequence, only one station can transmit in a cell area at a given time. The network access can still be controlled quite well by the AP. An example of a WLAN radio resource control method is the Point Coordination Function (PCF) used in IEEE 802.11 (Santamaría & López-Hernández 2001). In PCF, the AP handles the resource allocation by polling the stations inside its cell, allowing the terminals to access the channel without contention. This also enables a possibility for QoS classification, since the AP can give periodical access to some terminal(s) to enable, e.g., a CBR (Constant Bit Rate) service. PCF is an example of a polling based channel access method. In the IEEE 802.11 (later referred to as 802.11i) original standard, inter-cell mobility is not directly supported, but handovers are possible by performing normal disconnect and join processes sequentially. This, of course, is not very seamless, but almost the only option, since there are no separated control channels. At present, many vendors have their own variations of link level solutions for enhancing the handovers. There are also standardized ways to perform handovers like IEEE 802.11f, which uses Inter-Access Point Protocol (IEEE 2003). Quite recently this has been basi-
cally replaced by IEEE 802.11r, which defines how to enable fast roaming between WLANs (IEEE 2008).

If the traffic load is low, fixed channel access or even polling is inefficient, since there will be idle periods in the channel, wasting the capacity. This is the case especially in modern data communications, where the traffic is usually bursty in nature (Willinger et al. 1997). Hence, random access methods are needed (Tobagi 1980). Basic examples of random access methods are ALOHA and CSMA (Carrier Sense Multiple Access). In pure random access, all the terminals in the network, including the AP, compete for the channel access. This form of operation is efficient in low traffic load cases as there are no wasted fixed resources or additional polling overhead. Furthermore, if there are only two active communicating terminals in the network, they can freely use the whole available bandwidth, which is not typically possible, e.g., in cellular networks. However, the problem of random access is that the collisions will quickly become a limiting factor of performance when the traffic load and/or number of users increase. Also, as there is no actual control over the network, there are no guarantees whatsoever of packet delivery or delays and, hence, no prerequisites for QoS. Only the statistical average behavior of this kind of network can be moderately estimated when the average traffic load is known.

1.1.2 Structureless network

Even though the pure random access case is already difficult in the network management sense, the centralized WLAN structure eases the situation. The traffic is centralized to the AP, since even the traffic from one user to another, both residing in the same cell area, is routed via the AP, enabling some control on the AP side. Various good random channel access and MAC (Medium Access Control) protocols have been developed to be used in WLANs. One example is DCF (Distributed Coordination Function), which is used in 802.11 (Santamaría & López-Hernández 2001). Interestingly, DCF is typically used in current WLAN AP network implementations, even though PCF would provide various advantages over DCF, especially in crowded networks.

A completely different kind of network is an ad hoc network, which is structureless without APs or any centralized control. In an ad hoc network, no predetermined or stable topology exists, and the nodes may be mobile, dynamically joining and leaving the network. When mobility is included, this kind of network is often called a MANET (Mobile Ad Hoc NETwork). The idea of ad hoc net-
works is that they are self-configurable with distributed control, and can be formed quickly and accessed easily. This kind of network is attractive especially in military communications because of its dynamic nature and the absence of vulnerable master nodes.

The advantages also come with several disadvantages. As stated, random access communications is always difficult, but in an ad hoc environment this is even more difficult, since, e.g., the network traffic pattern is likely to be relatively random rather than concentrated to some central node. Hence, the access methods designed for centralized networks do not necessarily work well in ad hoc networks. Another interesting thing is that, since there is no fixed topology, there are also no guarantees that all the nodes will be within the radio range of each other (i.e., neighboring nodes). Hence, in a pure ad hoc network, all the nodes must act as traffic routers to other nodes in the network, which is quite challenging in such a dynamic environment.

Ad hoc networks can be divided into two main groups: flat and hierarchical ad hoc networks. Fig. 2 shows an example of a flat ad hoc network topology. In this, all the nodes are considered to be somewhat equivalent, which means in practice that the routes can be formed via any of the nodes in the network. In a hierarchical ad hoc network topology, the network is divided into clusters. Inside a cluster, direct routing can be used, but between different clusters, routing is done between cluster heads that might be similar to the other nodes, or they might be nodes with enhanced capabilities. The hierarchical structure relieves the complex routing problem in larger networks and may also help in organizing the network structure. However, the hierarchical structure also has several drawbacks as there will be extra overhead from organizing and maintaining the hierarchical structure, and, e.g., routing can easily be suboptimal and also inefficient when the topology is highly variable. (Haas & Tabrizi 1998), (Haas 1999)

A hierarchical structure is useful in a large network (>> 50 nodes), where flat routing becomes complex (Zou, et al. 2002). A hierarchical structure is also the obvious choice if the network contains nodes with different capabilities. Consider, e.g., a large military scenario, where squad members may have small radios to communicate with each other, while the communications between squads and platoons are handled by high power radios, and at the battalion level there could even be a wired high capacity connection (Kuosmanen 2004).

Since the division of hierarchical vs. flat topology is mainly a network level routing problem rather than a lower layer problem, only flat ad hoc networks are considered in this work.
1.2 Author’s contributions to the field and motivation

As the background of the author is in telecommunications, link and physical layers in particular but also traffic modeling, it was quite obvious that ad hoc networks based on IEEE 802.11 cannot work properly under higher traffic loads. To investigate and prove this, paper (Prokkola J, et al. 2003) was written. This paper is a straightforward simulation based study, where the performance of a 802.11 and AODV based ad hoc network was studied with existing models. The performance was studied as a function of the normalized offered network traffic load, and hence it was easy to show that with an increasing traffic load the performance collapsed even surprisingly quickly. The majority of existing ad hoc network related studies at that time were carried out as a function of mobility, network size, etc., where it is not that easy (or even possible) to see the effect of the traffic itself, even though some notations about poor performance under a high traffic load had been reported. The paper (Prokkola J, et al. 2003) also compared the effects of CBR and VBR (Variable Bit Rate) traffic source models and different session models, which were implemented in the simulation models.
The paper (Prokkola J, et al. 2003) used quite traditional traffic models, and hence another paper was written with the same basic idea but with different models. The new traffic models required some additions to the simulation models. In (Leppänen, et al. 2003), the effect of traffic to the ad hoc network performance was studied with Pareto distributed traffic, which correlates better to modern Internet traffic. However, the length of the conference paper (Leppänen, et al. 2003) was very limited, and not all of the intended ideas could be brought up. Thus, we decided to write yet another paper to conclude this traffic related study. As a result, the paper (Prokkola et al. 2004) was accepted as a part of a book released by the Finnish National Defense College.

Next, it was time to think about whether the poor performance of ad hoc networking could be somehow enhanced. As was found in (Prokkola J, et al. 2003), (Leppänen, et al. 2003), and (Prokkola et al. 2004), the main problems are not in the routing protocol but at the lower layer functionalities. Hence, a new channel access mechanism designed especially for ad hoc networking was developed and implemented by the author to the OPNET simulator. This Bi-Code Channel Access (BCCA) method was presented in (Prokkola & Bräysy 2004). In that paper, BCCA was tested in a small 10-node ad hoc network operating in an area of 400 m × 400 m. The used MAC was still 802.11, but BCCA was used at the physical layer. Despite the small network and the fact that 802.11 MAC is far from the optimal MAC solution for BCCA, the results were promising and showed that there is a lot of potential for enhancing the ad hoc network performance by lower layer design.

It then became clear that BCCA would need a special MAC solution in order to better exploit its potentiality. Thus, BC-MAC, a MAC solution designed for BCCA and for the needs of ad hoc networking, was developed and implemented by the author with OPNET. BC-MAC was presented in (Prokkola & Bräysy 2004b), where a network of 20 nodes operating in an area of 500 m × 500 m, was used as a basic scenario for studying the performance. It was shown that BC-MAC with BCCA boosted the ad hoc network performance to a new level when compared to 802.11.

BCCA and BC-MAC were then presented, but the drawback was that studies were carried out with simplified physical layer models, which are typically used in ad hoc networking. This kind of modeling can easily lead to errors, and we needed to prove that the developed methods work also with more realistic physical layer models. Hence, the paper (Prokkola, et al. 2005) was written. In that paper, the performances of BC-MAC and 802.11 based ad hoc networks were
studied with different propagation loss models. The used scenario was a 1500 m × 300 m area with 50 nodes. The simulation required the implementation of new propagation loss models to OPNET, and also some of OPNET’s default physical layer calculation methods needed to be further developed in order to make the simulation more realistic. While this important study showed that BC-MAC outperforms 802.11 regardless of the propagation loss modeling, it also showed surprisingly big differences in performance under different propagation models, thus verifying the assumption that simplified modeling can lead to inaccurate results. Accurate modeling is important, since the idea of this work has been from the beginning to be near to a real situation. The goals of being close to reality and designing feasible methods are also the main reasons why this study is not carried out by using theoretical analysis.

After (Prokkola, et al. 2005), BC-MAC and BCCA were presented and studied in different publications, but a concluding paper going into the details of the performance behavior was still missing. Also, the pure BCCA principle entails the drawback that the spreading code of the receiving node needs to be known. To relieve BCCA of this drawback, a novel method for distributing the spreading codes at the network layer was developed and implemented by the author to OPNET. This method, NSCD (Network layer Spreading Code Distribution), with detailed analysis of BCCA and BC-MAC vs. 802.11, were presented in (Prokkola & Bräysy 2007).

To be sure that BC-MAC functions as it is supposed to in various other situations and conditions, it become clear that a study considerably wider than a conference or journal paper is needed to handle most of the probable cases. Also, it would be desirable for the reader to have all the work in one package. This, in addition to implementation considerations, is what this thesis concerns.

1.3 Thesis outline

Chapter 2 presents common protocols and methods that are typically used in ad hoc networking and that are relevant to this work. In addition, some problems of ad hoc networking are brought up, and in Section 2.5, the problem to which this thesis proposes a solution is presented. Chapter 3 presents the developed new methods and the simulation models. Also, the simulation principles are discussed. Chapter 4 is the main results chapter, where the studied scenarios, results, and analysis are provided. The structure of Chapter 4 is based on studied scenarios, which are written so as to function as independent entities if the reader is familiar
with the basis of this work. After the results, Chapter 5 discusses the topic of implementation. As stated, the goal is to reach feasible methods, and thus that chapter considers what would be needed in implementing the methods presented in this work. Finally, the thesis is concluded with a summary and discussion, including ideas for future work.
2 Ad Hoc networking protocols and problems

In this chapter, some protocols and methods that are typically used and are relevant to this work are presented. Also, some of the problems of ad hoc networking are brought up, leading to the main topic of this work.

In this work, spread spectrum (SS) systems are considered. Thus, a brief introduction to the SS principle and Code Division Multiple Access (CDMA) is provided in Appendix 1 for those who are interested.

2.1 Routing protocols

Routing protocols in fixed networks have been used successfully for decades (Anttila 2001). However, these protocols do not perform well in the challenging dynamic ad hoc network environment, and this has caused a rush for ad hoc network routing protocol research (Perkins 2001). The intense ad hoc routing protocol research that began in the mid-90’s has created the impression that also ad hoc networking is quite a new research area. This is, however, not the case, since the idea of temporary structureless wireless communications is almost as old as the history of communications. Ad hoc networks, as they are understood today, have been studied for decades as (multi-hop) packet radio networks. In the beginning, research focused on lower layers (e.g., (Tobagi 1980, Tobagi 1987)), and the applications were mainly military orientated (Burchfiel, et al. 1975, Jubin & Tornow 1987)). Routing, however, was not an issue. The name ad hoc networking became common when research spread from the military to the academic world and started to focus on network layer issues.

In the early days of multi-hop packet radio networks, very simplified routing, like flooding (broadcast routing) (Gitman, et al. 1976), was used. The idea of flooding is simple: If a node receives packets that are not addressed to it, the node will just broadcast the packets to the neighborhood. Hence, routes are not formed, but the packets are just flooded in the network. As can be easily noticed, flooding is not suitable for large networks and, in addition, cannot handle high traffic loads. However, flooding has its place in small networks and in cases where packets need to be forwarded to their destination quickly and with high probability in an unknown network topology. Also, many modern routing protocols use flooding in the initial route discovery phase.

Nowadays, dozens of different good routing protocols exist for ad hoc networking. The principles usually originate from those used in the fixed network.
Ad hoc routing protocols can be categorized as table driven, on-demand, or hybrid routing protocols, as, e.g., in (Leppänen 2001). There are also other ways to categorize routing protocols, examples of which are flat, hierarchical, geographical, power aware, QoS aware, multicast, and geological multicast.

Some principles and key features can be identified from each category of protocols. In table driven protocols (proactive), the nodes continuously update a database of routes (routing table) including all possible destination nodes in the network. Routing is fast, since the routes are always waiting to be used. However, as the routing information must be updated periodically, the routing overhead increases rapidly with an increasing network size. This becomes inefficient, especially if the data traffic load is low, since most of the time the functional routes are not used.

On-demand routing protocols (reactive) form the routes only when needed and delete the routes that are not used. This reduces the overhead considerably, and, hence, these protocols are well suited also for larger networks and have good adaptation capabilities to topological changes. The drawback is the delay in route establishments, since the routes must be formed before communications can take place. Some hybrid protocols are also presented, and they aim to utilize the best features of table driven and on-demand routing protocols.


2.1.1 Ad Hoc On-demand Distance Vector Routing

The AODV (Ad hoc On-demand Distance Vector) routing protocol is the one used in this study. Thus, understanding its basic principle is important considering this work.

AODV is probably one of the most popular routing protocols in ad hoc networking. The popularity and the feasibility of AODV have been noted also by the Internet Engineering Task Force (IETF) (Perkins, et al. 2003). However, it is not argued here that AODV would be the best routing protocol or even that it would definitely be the best one for the purposes of this work. AODV is chosen for its known properties to function generally well in different kinds of flat ad hoc network environments. Also, the implementation of the proposed methods is straightforward with AODV.
AODV is based on the DSDV (Destination-Sequenced Distance-Vector, (Perkins & Bhagwat 1994)) routing protocol. Because of the table driven nature of DSDV, it is ineffective in large ad hoc networks. AODV is a pure on-demand route acquisition protocol, where routes are formed only when needed. Also, unused routes become obsolete with a timer. There are no periodic routing table updates, which makes it suitable also for large networks. However, local hello messages may be used to increase the awareness of the local connectivity. Alternatively, AODV can rely on MAC for maintaining local connectivity if the MAC is intelligent enough. In principle, AODV is independent from lower layer functionalities as long as basic connectivity is provided (i.e., nodes can communicate with each other). In basic AODV, bi-directional links are assumed, but additions have been also made to allow the operation over directional links. The protocol is loop free, but as work for this thesis has taught, careful attention must be paid to the implementation to guarantee loop freedom in reality, especially in the case of a congested network.

The operation of AODV has a distributed nature, as the intermediate nodes within a route only know the next hop node. This kind of operation requires less control information than, e.g., source routing protocols, like DSR (Dynamic Source Routing), where all route information is carried in packets (Johnson & Maltz 1996). Minimizing the control load is preferable in ad hoc networks, since even in the simplest cases the amount of control overhead can be problematic. To avoid loops, the route information maintenance system in AODV needs to be somewhat more complex than in source routing protocols. Monotonically increasing route sequence numbers are used to supersede stale cached routes.

The basic operation of AODV is as follows. When a packet arrives from the upper layer, AODV checks the routing table. If a valid route to the destination node is found, the packet is forwarded to the next hop node of that route, but if not, route discovery is initiated. In route discovery, a route request (RREQ) packet is flooded to the network. When receiving a RREQ, a node checks its routing table. If a fresh enough route to the desired destination is not found, RREQ is forwarded to the neighbors. If a route to the desired destination is found (or the node is the destination), the node initiates a route reply (RREP) packet back to the source. When receiving a RREQ, a node learns a reverse path to the source. Hence, the route is in fact already formed when the RREQ reaches its destination. However, the reverse path is only temporal, and the RREP packet is needed to confirm the created forward path. Naturally, the destination might receive multiple RREQ packets via different routes. In basic AODV, the first arrived RREQ is
chosen and replied with RREP, while the latter ones are forgotten. Hence, AODV stores only one route for each destination node. The first RREQ is a good guess about the fastest route, regardless of the hop count. There are also some suggestions for AODV route caching, which might give some gain in some situations (Barua & Agarwal 2002), but these are not used here.

In a dynamic ad hoc network, the topology changes and links might be lost. If such a link failure is noticed by an intermediate node while forwarding a packet within an active route, the node might launch a route repair functionality. In this, the intermediate node launches a route discovery for the destination. If the route is repaired, but the new one is longer than the original one, the intermediate node is responsible for informing the source node about this. If the route repair fails, a route error (RERR) packet is sent back to the source. The purpose of RERR is to invalidate the route, and it is up to the source node whether to launch a new discovery or not.

Now the basic functionality of AODV is presented. The actual implementation of the protocol is, however, of course far more complex, and there are several special situations which need to be solved, but there is no point to go through those here. Also, AODV includes various parameters, which affect the performance. One interesting parameter is the AODV active route timeout parameter (AART), which determines the lifetime of an inactive route.

More details on AODV can be extracted, e.g., from (Perkins, et al. 2003).

2.2 Link and physical layer protocols and methods

Some common protocols and methods used in the link and physical layers, important when considering this work, are presented next.

2.2.1 Channel access: ALOHA and CSMA

The ALOHA system can be thought to be a pioneer of packet radio networking (Abramson 1970). The ALOHA channel access method allows any station in the network freely to transmit at any time. If the network traffic load is low, this method functions well, but is quickly corrupted by collisions with an increasing traffic load.

CSMA is a method to improve the performance by reducing the collisions by monitoring the channel before transmission. The receiver senses the channel, and if it is detected to be free, the transmission can be made. If not, there are several
options of what to do, depending on the variation of CSMA. In np-CSMA (non-persistent CSMA), a random timer is set to check the channel again in the future. The same procedure is repeated until the channel is detected to be free. In p-persistent CSMA, if the terminal detects that the channel is busy, it can either transmit immediately after the channel becomes idle with probability $p$ or use the same principle as in np-CSMA. In 1p-CSMA, the terminals will always transmit immediately after a busy period if they have packets waiting in a queue. There are also several variations of these basic methods.

The throughput performance of np-CSMA as a function of offered traffic can be calculated as (Tobagi 1980)

$$S = \frac{Ge^{-ag}}{G(1+2a)+e^{-ag}},$$

where $a$ is the normalized propagation delay (propagation delay divided by the packet transmission time). Parameter $a$ has a great impact on the performance, since an increasing propagation delay leads to a higher probability of a carrier sensing failure. In Fig. 3 (Kleinrock & Tobagi 1975), the normalized throughput ($S$) performance of ALOHA and several CSMA variants in a single-hop network are presented as a function of a normalized offered traffic load $G$ ($a = 0.01$). The normalization is done to the network transmission capacity, which in this case equals to the channel bit rate of the physical layer, as no control overhead is assumed.

It must be kept in mind that the presented graphs give a good overview but are theoretical. For example, it is assumed that in the case of collision, all the colliding transmissions are always lost, even though the collision is only partial. Obviously, this is not generally true due to the capture effect in the radio channel: If the colliding transmissions arrive with different power levels, it might be possible to receive correctly the one with the highest power. In fact, the reception might not be lost even in the case of a full collision (full overlap of transmissions as a function of time) if the difference in power levels is big enough. This is especially true in the case of SS systems. Also, forward error correction (FEC) methods ease the effect of collisions. The capture effect narrows the difference between slotted and non-slotted protocols. Another thing is that CSMA was originally meant for wired networks, where signal sensing is easy. In radio networks, sensing is more difficult and also uncertain, which can decrease the performance from the ideal case. In wireless networks, also the sensing range of CSMA plays an important role.
A good tutorial for ALOHA, CSMA & variants, and other channel access schemes is provided in (Tobagi 1980).

Fig. 3. The theoretical throughput performance of different ALOHA and CSMA schemes. Figure reprinted from (Kleinrock & Tobagi 1975), © 2008 IEEE.

2.2.2 Acknowledgements

Wireless communications always introduces an unreliable link level. Hence, some sort of confirmation of data transmission is required for reliable point-to-point transmission.

Already the ALOHA network used a simple ARQ (Automatic Repeat re-Quest) method, where every transmitted data packet was acknowledged by the receiving end. If the acknowledgement packet (ACK) is not received within a certain time, the transmitting node assumes that the sent data packet has been lost and initiates a retransmission (reTx). This kind of protocol is called a stop-and-wait ARQ, and it is still very popular today (e.g., 802.11 uses a stop-and-wait based ARQ method). Different kinds of variants can be made by having different calculation methods, e.g., for reTx timing.

Although stop-and-wait ARQ is very simple, it is effective in channels where data errors are common (wireless channels typically). However, in the case of a
reliable channel, it wastes resources as there is no point in acknowledging every single packet. Hence, more efficient methods (e.g., sliding window and go back n type protocols) are developed. Good tutorials for basic ARQ methods are provided, e.g., in (Tanenbaum 2003) and (Bersekas & Galleger 1992).

### 2.2.3 Collision avoidance with handshaking

Consider a situation, presented in Fig. 4., where both nodes \( n_2 \) and \( n_3 \) are transmitting to \( n_1 \). The problem is that \( n_2 \) and \( n_3 \) can communicate with \( n_1 \), but they do not hear each other (i.e., they are outside the sensing range of each other). Hence, CSMA does not work, and the channel access acts effectively as ALOHA, giving high probability for collisions. This situation is called the hidden terminal problem.

To overcome this, the idea of MACA (Multiple Access with Collision Avoidance) protocol was composed in (Karn 1992). In this, a handshaking procedure is carried out prior to transmission of the actual data. Consider, e.g., the network in Fig. 4: If \( n_2 \) desires to send data to \( n_1 \), it first sends a small Request-to-Send (RTS) packet. When \( n_1 \) receives the RTS packet, it replies with a Clear-to-Send (CTS) packet if it is ready for data reception. If not, it may ignore the received RTS packet, and \( n_2 \) may try again later after noticing (timer expires) that the CTS packet has not been received. The CTS packet is an indication that the data transmission can take place. Hence, in our example, \( n_2 \) is then free to send the data. The hidden node \( n_3 \) does not receive the RTS, but it does receive the CTS. Hence, it will be notified for the upcoming transmission and is, therefore, able to defer its transmission to the end of the transmission \( n_2 \rightarrow n_3 \). Both packets RTS and CTS contain information about the duration of the possible upcoming data transmission, so that all the nodes receiving these packets are able to defer their transmission and, thus, avoid collisions.

The MACA principle has been popular, and also 802.11 specifies a method which is mainly the same as the basic MACA principle. MACA is efficient in reducing collisions, but the drawback is naturally the handshaking overhead. The idea is that RTS and CTS packets are significantly smaller than the upcoming data packet. If not, the MACA principle produces a considerable amount of control overhead and wastes resources. Also, RTS and CTS packets are still vulnerable to collisions, but if they are small compared to the data packet, the collision probability is also smaller. Hence, this advantage is also lost in the case of small data packets. This is the reason why in many MACA protocol implementations (e.g.,
in 802.11) there is a data packet size threshold for enabling the handshaking procedure.

The MACA protocol performance has been analyzed in single-hop and multi-hop environments, e.g., in (Chen et al. 2003).

Fig. 4. The hidden terminal problem.

2.2.4 IEEE 802.11

The original 802.11 standard specifies FHSS and DSSS physical layers. We are interested in the latter one. Some details of it, and of the 802.11 MAC layer, are given here, because 802.11 is used as the point of comparison in this study. More exact information is provided, e.g., in (IEEE 1999) and (Santamaría & López-Hernández 2001).

The basic 802.11 operates in an unlicensed 2.4 GHz ISM frequency band (2.4000–2.4835 GHz in Europe). The DSSS mode specifies 1 and 2 Mbit/s channel bit rates using BPSK (Binary Phase Shift Keying) and QPSK (Quadrature Phase Shift Keying), respectively, while the symbol rate is always 1 Msymbol/s. The chip rate is 11 Mchip/s (giving a spreading factor of 11 with 1 Mbit/s), and a unique Barker sequence is used for spreading in all implementations. The later versions of 802.11, namely IEEE 802.11b, IEEE 802.11a, IEEE 802.11g, and 802.11n, introduce new modulation mechanisms and features, allowing considerably higher data rates, but we focus only on basic 802.11, since the absolute data rate is not the main point of this study. Moreover, we are interested in meth-
ods incorporating SS techniques, while 802.11a, 802.11g, and 802.11n can no longer be considered to be SS techniques.

The MAC operation of 802.11 is based on np-CSMA, and the PHY is responsible for making the decision of clear channel assessment (CCA). There are two main methods for this: energy detection in the channel, and carrier (valid DSSS signal) detection. Combinations of both of them can also be used. In this study, energy detection will be mainly used for CCA, but also carrier detection will be tested.

The PHY adds some overhead in the form of PLCP (Physical Layer Convergence Protocol). The PLCP frame of DSSS is shown in Fig. 5. PLCP is divided into two parts, the first of which is a preamble that is to allow some time for the receiver to synchronize (SYNC) to the transmission. Start Frame Delimiter (SFD) is a 16-bit word (HEX: F3A0), which is used to indicate the end of the SYNC part and beginning of the header. In the second part (header), the Signal field is used to indicate the modulation (and also the data rate). The Service field is reserved for future usage, The length specifies the transmission time of MPDU (MAC Protocol Data Unit) in μs, and a UI-R CRC 16 polynomial generator (Cyclic Redundancy Check) is used to detect errors in the PLCP frame.

![Fig. 5. IEEE 802.11 PLCP format (DSSS mode).](image)

**IEEE 802.11 MAC**

As stated, 802.11 MAC has two different operation modes: PCF and DCF. In this work, the latter is of interest, and, thus, it is introduced briefly.

DCF has two modes: CSMA with collision avoidance (CSMA/CA), and RTS/CTS handshaking. CSMA/CA is based on np-CSMA. If the medium is found to be busy, the contenting station first defers DIFS period (DCF Inter-Frame Space (= 50 μs for the DSSS)) and then initiates the exponential back-off algorithm. The timeline is divided into slots, where the slot time in the case of DSSS is 20 μs. The number of backoff slots to be waited before the transmission is calculated with the equation
where Floor(\(\cdot\)) rounds the random number (Rand(\(\cdot\)) gives a random number from \([0, x]\)) downwards to the nearest integer value. \(N_{\text{bo,max}}\) stands for the maximum number of slots, and takes the initial value of \(N_{\text{CW,min}} (= 31 \text{ for DSSS})\) after the first unsuccessful try for channel access. After this, the maximum number of slots is increased as \(N_{\text{bo,max}} = 2N_{\text{bo,max}} + 1\) for every unsuccessful frame transmission. This is continued until the contention window limit \(N_{\text{CW,max}} (= 1023 \text{ for DSSS})\) is reached, and \(N_{\text{bo,max}} = N_{\text{CW,max}}\) for the remaining tries. The back-off timer is reduced only when the channel is detected to be idle. \(N_{\text{bo,max}}\) is reset for every new frame.

The purpose of the exponential back-off algorithm is to reduce the possibility of sequential collisions in the case where several terminals are competing for channel access. Overall, the channel access method of 802.11 promotes fairness as every station must contend for channel access for every new frame. Nevertheless, it is found that fairness could still be improved (e.g., (Vaidya, et al. 2000)).

Fig. 6 depicts the timing diagram of a successful data transmission with CSMA/CA. The station intending to transmit must first sense that the channel is free for the DIFS duration. If the channel is still free after this, the node can transmit its data frame immediately. Other nodes hearing the transmission can extract the information considering the length of the transmission from the frame header, and update their network allocation vector (NAV). Stations consider the channel to be busy when NAV is set, so it acts as virtual carrier sensing to allow the ongoing transmission to be free of collisions. After data reception, the destination node waits for SIFS (Short Inter-Frame Space (in DSSS = 10 \(\mu\)s)) before sending an ACK. SIFS is the shortest interval defined in 802.11, and it is used to give priority in situations requiring quick response. After ACK transmission, the channel is considered free for contention after a DIFS.

The finite propagation delay (\(t_{\text{prop}}\)), which is always present in reality, delays the carrier sensing and causes timing errors, as seen in Fig. 6. However, in a WLAN area, transmission ranges are typically only some tens or hundreds of meters, giving propagation delays of the order of 0.1, 1 \(\mu\)s, which does not yet have major effects on the performance.
In the RTS/CTS handshaking mode, CSMA/CA is used as a basis, but RTS and CTS packets are exchanged prior to data transmission, following the MACA principle. The timing diagram of this mode is seen in Fig. 7. As seen, the other nodes can update their NAV also from the RTS and CTS packets, enhancing the virtual carrier sensing beyond the CSMA/CA. In the case of RTS failure (no CTS packet received), the same procedure is followed as in the case of data packet failure. The only difference is that the RTS packet retry limit is usually different from the limit for data packets. When RTS/CTS is enabled, *Long Retry Limit* (default = 4) specifies the retry limit for RTS and the following data packet. Without RTS/CTS, *Short Retry Limit* (default = 7) specifies the retry limit for data.

The stations can choose not to use RTS/CTS, use always RTS/CTS, or use RTS/CTS whenever the MSDU length exceeds the determined threshold value. Because of its nature, RTS/CTS operation mode is sometimes also called *four-way handshaking*. 

![IEEE 802.11 basic CSMA/CA timing diagram of a successful data transmission.](image1)

![IEEE 802.11 basic RTS/CTS timing diagram of a successful data transmission.](image2)
If the upper layer data packet exceeds the maximum data frame size of 2346 bytes, fragmentation is executed at 802.11 MAC. After the contention for the channel, the fragments are just sent sequentially acknowledging all the fragments and waiting a SIFS between fragments and ACKs. However, fragmentation is hardly ever needed, since 802.11 is typically used to carrying IP (Internet Protocol) packets, and thus the possible fragmentation is usually done already at the IP level to fit packets to Ethernet frames (the fragmentation limit for Ethernet is 1500 B).

The general 802.11 MAC frame contains 34 B of control information (addresses, CRC, etc.) and 0–2312 B of data. The RTS packet length is 20 B, CTS is 14 B, and ACK is 14 B.

2.3 The problem of charge

The idea of ad hoc networking is based on the assumption that all the nodes in the network act as routers for other nodes, and hence no base stations are needed. In a controlled environment, e.g., military or public authority, it is easy to require and execute such a principle. However, in a civilian environment, satisfying this requirement will be an issue.

Consider, e.g., a weak connectivity network case with a single bottleneck link. It is quite obvious that the nodes forming the bottleneck link will have to route clearly more traffic than the other nodes in the network. Thus, it is likely that most of the data traffic they receive and transmit belong to other nodes in the network. So the question is, why would they be eager to consume the valuable resource of battery energy for routing data traffic generated by others? Also, because of the extra traffic, the nodes near the bottleneck link would suffer decreased QoS just because of their unfavorable location in the topology.

Another evident problematic case would be that only a single node in an ad hoc network would have a connection to a fixed network. As a result, this edge node would be routing all the network traffic between the wireless ad hoc network and fixed network. The edge node might have a power connection, which would solve the power consumption problems. However, if the edge node had to pay for its fixed network connection (which is the usual case in civilian fixed network connections), and in the worst case, pay per transferred amount of data, acting as an edge router would not sound very tempting. Also, the fixed network connection bandwidth might be quite limited, which would cause performance
problems in getting the edge node’s own traffic through while routing massive amounts of other nodes’ traffic.

This *problem of charge* cannot be solved easily, if at all. There are only a few approaches proposed for solving it. E.g., in (Buttyán & Hubaux 2003) an idea of virtual currency, *nuglets*, is proposed. In this, the devices in the network would have a hard-coded counter for the nuglets. Nuglets would be spent when transmitting a node’s own traffic and gained when forwarding data of other nodes. In this way, the nodes would be forced to act also as routers if they themselves wish to send data continuously. As can be easily determined, this nice idea still has several unsolved issues, e.g., what happens when the nuglets run out, who controls the nuglets, how to take the needed consumed transmission power into account, how would this system be implemented in reality for the numerous different wireless devices of different vendors, and how to solve the basic problem that nodes lying in different parts of the network will most definitely have different intensities of offered routing traffic load?

The current proposals are mainly of interest from the academic and theoretical point of view, and the problem of charge remains unsolved also after this thesis. The problem of charge might be one of the main reasons why ad hoc networking is still restricted mainly to military and public authority environments.

### 2.4 The connectivity problem

Its dynamic nature and lack of topology are the greatest strengths of ad hoc networking, but these are also its greatest weaknesses.

Ad hoc networks suffer from *connectivity* problems, since topologies are random and there are no guarantees of network coverage. Hence, it is quite possible that, e.g., some node in the network is separated and has no radio connection to other nodes. The network might even be physically separated into clusters that do not have possibilities to communicate with each other. An example of this kind of topology is shown in Fig. 8. Connectivity study is its own form of art, and there are lots of publications on this topic (e.g., Philips, *et al.* 1989, Desai & Manjunath 2002, Shu *et al.* 2003, Madsen, *et al.* 2005). The problem of connectivity turns out to be quite mathematical, and theories on it easily stray too far away from communications. In particular, it is often not straightforward, if at all possible, to evaluate from the calculated connectivity results the effect of connectivity on the actual data communications. Solving the problem of connectivity is not the aim of
this work, but this problem needs to be discussed, since it always affects the performance when dealing with ad hoc networks.

Connectivity can be understood in many ways. In this work, a connected network means that every node in the network belongs to the same graph (i.e., there is a path between every node in the network). Connectivity can also be understood as the degree of connectivity, which in practice means how many neighbors the nodes in the network have on average. The latter is also very important, since there might be situations where the network has several clusters that are not fully separated but have only weak connections between them. A weak connection means a clear bottleneck connection. Fig. 9 shows an example of a situation where two quite strongly connected (several routes between nodes) clusters have only a single link between them. It is quite likely that in this kind of network, the cluster connecting link will be a bottleneck and will decrease the overall performance of the network (assuming that the communication pattern between nodes is somewhat random and that all links use the same radio interface). This situation is called weak connectivity.

As easily realized, there are no ways to tackle the physical connectivity problem by MAC or upper layer design. If better connectivity is desired, the only way is to increase the transmission range. This can be done by increasing transmission power, decreasing the data rate, or using directional antennas, but since none of these are considered in this work, the connectivity will set the lower bound for the achievable performance.
In the scenarios of this thesis, a finite number of nodes are placed randomly onto a finite area. In such a random topology, 100% topological connectivity cannot be guaranteed if the effective transmission radius is smaller than the longest dimension of the operational area.

![Image of weak connectivity ad hoc network topology](image)

**Fig. 9. An example of a weak connectivity ad hoc network topology.**

### 2.4.1 A tool for analyzing connectivity

Since connectivity sets the minimum bound for the performance, its effect must be considered to separate it from the system level performance. For that purpose, an external simulator has been developed by Mr. Kai Lieska (University of Oulu, 2003). This simulator works from the geometrical point of view and ignores the effects of communication layers. It uses a strict radio range for successful transmission (connection) and calculates the probability of network connectivity by using Monte Carlo simulations of uniformly randomly distributed nodes in a predetermined network area. The simulator also calculates the frequencies of the separated network segment sizes in the case of a disconnected network. This network segment distribution is needed when calculating the effect of connectivity to the packet loss. Segment means a connected cluster of nodes that are separated from the other nodes in the network.

A screenshot from the connectivity analysis tool is presented in Fig. 10. Basic inputs for the tool are: size of the area (length, height), transmission range, and number of nodes. One can also set the number of averaging iterations, which should be quite large in order to get reliable results (> 1000).

As an output, the tool simply calculates the probability of a connected network (number of non-connected networks / number of iterations). As additional information, the tool can also calculate the average shortest path (in hops), the
coverage area of the nodes, and the network segment distribution. It is interesting to note that the coverage area is not necessarily related to the ad hoc connectivity. This can be easily seen in the visualization example of Fig. 10, which shows a disconnected ad hoc network topology where the coverage is still 100% of the studied area. Coverage is of interest in the cellular network design, where it is assumed that the nodes (base stations) will have a way to connect to the backbone network. As regards ad hoc networks, however, 100% coverage does not mean much as such.

![Fig. 10. A screenshot of the connectivity analysis tool.](image)

**Connectivity analysis**

If we consider the average packet loss caused only by connectivity, not much information is gained from a simple probability of a connected network. Generally, the overall packet loss ratio in a network with $N_n$ nodes can be calculated simply as
\[ P_{\text{loss}} = \sum_{n} F_{S_n} p_{\text{loss}} \]  

where \( F_{S_n} \) is node \( n \)'s fraction of the total generated traffic in the network, and \( p_{\text{loss}} \) is the packet loss probability of the traffic generated by node \( n \). Let us now consider a case (Case 1) where the nodes communicate randomly with each other in the network, but a single node is left out of connection. By assuming a similar traffic process for all the nodes in the network, the separated node’s fraction of the total generated traffic is \( 1/ N_n \). The number of possible destination nodes for all the nodes is, naturally, \( N_n - 1 \). For the separated node, all the destination nodes are out of reach, and, thus, all the packets will be lost. For other nodes in the network, the probability of choosing the destination node for communications to be the separated one (probability of packet loss) is \( 1/ (N_n - 1) \). Hence, by using equation 3, it can be calculated for this case that the average packet loss due to connectivity in the whole network is

\[ p_{l_c} = \frac{1}{N_n} \left( \frac{N_n - 1}{N_n - 1} + \frac{N_n - 1}{N_n} \right) = \frac{2}{N_n}. \]  

Let \( N_n \to \infty \), which leads to \( p_{l_c} \to 0 \), as the single separated node becomes meaningless.

Next, we consider a case (Case 2), where the network is split into two equally sized segments. In this, both segments have \( N_n / 2 \) nodes, and both generate half of the total traffic. There will be packet loss if the chosen destination node is located in the segment other than where the source node is. Thus, the packet loss becomes

\[ p_{l_c} = \frac{1}{2} \left( \frac{N_n}{2} - 1 + \frac{N_n}{2} - 1 \right) = \frac{N_n}{2N_n - 2}. \]  

If now \( N_n \to \infty \), it will cause \( p_{l_c} \to \frac{1}{2} \). As seen, even though the network is in both of the example cases (Case 1 and Case 2) split into two segments, the average packet loss differs completely. Therefore, deeper information about the connectivity is required in order to calculate its effect on the packet loss: Network segment distribution is needed.

Network segment distribution gives the probabilities for different kinds of disconnected network cases, and it is the key for calculating the probability of packet loss due to connectivity with random sessions. Let \( n_m \) be the number of
nodes in the network segment \( m \). The probability that a node belonging to network segment \( m \) randomly chooses to communicate with a node lying in another segment is

\[
p_{\text{to}} = \frac{N_n - n_m}{N_n - 1}.
\]

(6)

Given the probability \((n_m / N_n)\) that a node belongs to the set of \( n_m \) nodes and by summing over all \( N_n \) network segments, we get the packet loss in a single disconnected case of

\[
p_{\text{to, single, case}} = \sum_{n_m=1}^{N_n} \frac{n_m}{N_n} \frac{N_n - n_m}{N_n - 1}.
\]

(7)

Finally, including the probabilities \( p_k \) for \( N_k \) possible network subdivisions, we get the total probability for packet loss due to connectivity of

\[
p_{\text{to}} = \sum_{k=1}^{N_k} \left( p_k \sum_{n_m=1}^{N_n} \frac{n_m}{N_n} \frac{N_n - n_m}{N_n - 1} \right),
\]

(8)

where \( n_{km} \) is the number of nodes in the \( m \)th segment of the \( k \)th subdivision case.

**Connectivity examples**

The derived equations do not help much if information on the network segment distribution, subdivision probabilities, etc. is not given. For this, the connectivity simulator is used. It must be noted that as simulations are used in addition to the analytical approach, the obtained results are not absolute. However, using large number of iterations (random network topologies), the results will be accurate enough for the purposes of this work. Solving the exact network segment distribution analytically would be gratuitously complex, and an overkill when considering the focus of this work.
Table 1. The network segment distribution of a scenario (10 nodes with a 250 m effective radio range in an area of 600 m × 600 m) with 1,000,000 simulated random topologies.

<table>
<thead>
<tr>
<th>Occurrences</th>
<th>Nodes per network part ((h_{\text{par}}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>473330</td>
<td>10</td>
</tr>
<tr>
<td>181249</td>
<td>9</td>
</tr>
<tr>
<td>84183</td>
<td>8</td>
</tr>
<tr>
<td>51357</td>
<td>7</td>
</tr>
<tr>
<td>40303</td>
<td>6</td>
</tr>
<tr>
<td>18888</td>
<td>5</td>
</tr>
<tr>
<td>29114</td>
<td>8</td>
</tr>
<tr>
<td>31854</td>
<td>7</td>
</tr>
<tr>
<td>21253</td>
<td>6</td>
</tr>
<tr>
<td>8945</td>
<td>6</td>
</tr>
<tr>
<td>17026</td>
<td>5</td>
</tr>
<tr>
<td>12800</td>
<td>5</td>
</tr>
<tr>
<td>5691</td>
<td>4</td>
</tr>
<tr>
<td>4876</td>
<td>4</td>
</tr>
<tr>
<td>2077</td>
<td>7</td>
</tr>
<tr>
<td>3477</td>
<td>6</td>
</tr>
<tr>
<td>2990</td>
<td>5</td>
</tr>
<tr>
<td>2354</td>
<td>5</td>
</tr>
<tr>
<td>1355</td>
<td>4</td>
</tr>
<tr>
<td>4071</td>
<td>4</td>
</tr>
<tr>
<td>586</td>
<td>4</td>
</tr>
<tr>
<td>637</td>
<td>3</td>
</tr>
<tr>
<td>763</td>
<td>3</td>
</tr>
<tr>
<td>84</td>
<td>6</td>
</tr>
<tr>
<td>205</td>
<td>5</td>
</tr>
<tr>
<td>137</td>
<td>4</td>
</tr>
<tr>
<td>162</td>
<td>4</td>
</tr>
<tr>
<td>158</td>
<td>3</td>
</tr>
<tr>
<td>55</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Let us consider a scenario of 10 nodes with a radio range of 250 m located randomly in an area of 600 m × 600 m. Using the connectivity simulator, after 1,000,000 iterations we get the network segment distribution shown in Table 1.
As seen, this kind of scenario is not a very favorable ad hoc network scenario, since the probability of full connectivity is only about 47% (473,330 cases out of 1,000,000). If disconnected, the network is split mainly into two or three segments, but also divisions into four or five segments are possible, and even some topologies of six segments are observed. Based on this network segment distribution, and using equation (8), we arrive at the conclusion that the average probability of packet loss due to connectivity problems is as high as ~ 21%. In this, random sessions were assumed, so the situation might be even worse if, for example, one of the nodes would be a server for some application that the other nodes are using. Also, it must be noted that this way of analysis does not consider additional problems caused by weak connectivity.

This was a nice example of ad hoc network failure and the need for connectivity analysis. Even though the presented scenario sounds at first quite reasonable, it is practically unusable. It can be concluded that in an area of 600 m × 600 m, there should be either more nodes or the nodes should have higher transmission power to increase the effective radio transmission range in order to enhance connectivity to a reasonable level.

![Fig. 11. Average packet loss ratios of 400 simulated random ad hoc scenarios with 10 nodes in an area of 600 m × 600 m (~ 250 m radio range).](image)

To prove the presented connectivity analysis, the previous scenario was tested with a network simulator (OPNET). The 10 nodes are equipped with BC-MAC and AODV (the setup is very similar to the basic setup presented in Chapter 4.5
with the exception of area size and node count). The traffic type and load will be such that it does not cause extra difficulties in the network (low normalized load of 0.01), i.e., the connectivity should be the dominating and practically the only cause for packet loss. 400 iterations were run, and the average packet losses as a function of iterations are shown in Fig. 11. The scenarios with connectivity problems can be quite easily seen in the figure, as the packet losses are high, while practically zero packet loss is got otherwise. By observing the behavior of calculated mean packet loss, we can note that the packet loss seems to correlate quite nicely to the calculated connectivity packet loss of ~ 21%. As seen, the packet loss settles near 21% after about 70 iterations. Naturally, the simulated value might not reach the theoretical one exactly, even if iterations $\rightarrow \infty$, since simulations present the “real” case without simplification assumptions as in the theoretical calculations (also, it must be noted that the theoretical limit is in fact semi-theoretical because of the simulated network segment distribution). Nevertheless, very good results were achieved, and, thereby, the correctness of the connectivity analysis tool and the derived connectivity packet loss equations are verified.

Connectivity properties of the studied scenarios

As was shown, scenarios should be designed carefully. In this thesis, scenarios of high connectivity will be used, and next we will consider the connectivity properties of these scenarios. Using the presented methods, the results in Table 2 were obtained. As seen, the connectivity properties of the proposed scenarios are very good, and, hence, this allows us to focus on topics other than connectivity. It must be noted that, even though the packet loss ratios due to connectivity are yet not insignificant, they are long term average values. If, for example, we consider the 20-node scenario, the calculated probability for connectivity (~ 98.96%) means that only about 1 topology out of 100 would be disconnected on average. Hence, it is likely that in the network simulations, where typically less than 100 averaging iterations are run, there will not be disconnected topologies in every set, and the possible disconnected ones can be identified quite easily on the basis of a sufficiently high packet loss ratio as compared to the other similar runs.

In the stationary network simulations, the study is restricted (if not especially mentioned) to full connectivity cases, because the connectivity problem is not the main issue in this work. In practice, this means that if the simulated random ad hoc scenario has a disconnected topology iteration, the results for that iteration are not included in the study. It should be pointed out that this refers only to cases
of disconnected networks; weak connectivity cases are normally included. On the other hand, in a mobile network case such a division into connected and disconnected networks is not made, since possible disconnected network situations are not permanent. Thus, in mobile scenarios, the connectivity problem is present.

Table 2. The connectivity properties of the studied scenarios.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>$P_{\text{connected}}$</th>
<th>$P_{\text{l_c}}$</th>
<th>Node density</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 nodes, 250 m range, 500 m × 500 m</td>
<td>98.96%</td>
<td>0.156%</td>
<td>80</td>
</tr>
<tr>
<td>50 nodes, 250 m range, 1500 m × 300 m</td>
<td>98.61%</td>
<td>0.446%</td>
<td>111.11</td>
</tr>
<tr>
<td>50 nodes, 250 m range, 500 m × 500 m</td>
<td>99.998%</td>
<td>$8 \cdot 10^{-6}$%</td>
<td>200</td>
</tr>
</tbody>
</table>

2.5 Lower layer functionalities and ad hoc network performance

When considering ad hoc networks, the focus is usually on the network layer and, more specifically, in the routing protocol. One of the main reasons for this is that IEEE 802.11 and its successors have reached a de facto position, defining MAC and PHY to be used with ad hoc networking. However, the lower layers set the basic limits on the achievable performance, and not much can be done by modifying higher layers to overcome these limits.

Consider, e.g., the limits set by 802.11 on a single link (Jun et al. 2003). These limits cannot be overcome by routing protocol design. Maybe the only aid at the network layer could be to design routing mechanisms that perform load balancing, i.e., try to balance the routing load more or less evenly in the network, and thus avoid individual links from overloading (Lee & Gerla (2001). Load balancing, however, does not help in small networks, and it does not solve the limiting factor: the performance of lower layers. It should also be noted that in a multi-hop environment, all the problems of links will be cumulated. Hence, if a single link has only fair performance, the performance over multiple hops will be poor (see, e.g., (Desilva & Das 2000)). This makes the lower layer design extremely important for ad hoc networking. In addition to the individual link performance, some general principles of the lower layer functionalities are such that they limit the overall network performance.
2.5.1 Operation on a common contention channel

To be able to allow easy access to the network, ad hoc networks typically operate on a common contention channel. In this kind of channel, all the communicating nodes compete for the use of the common media. Hence, collisions tend to limit the performance, and several methods have been developed for collision avoidance. In a centralized random access network environment, efficient CA methods are needed, since most of the communications involve the central node, which will be heavily loaded. However, in the case of an ad hoc network (and especially a flat ad hoc network), the situation is completely different. For example, it is likely that the traffic pattern is more or less random rather than concentrated on some central node. Even if there are some server nodes or edge routers in an ad hoc network, the multi-hop environment will distribute the traffic pattern. The result is that even in a small area there is likely to be needs for several independent transmissions of different sessions considering different source and destination nodes.

Let us now examine what will happen when a common channel with efficient CA methods is used in an ad hoc networking environment. Consider the situation in Fig. 12, where a transmission from node $n_5$ to node $n_4$ takes place. In the case of CSMA, nodes around the carrier sensing range (in this case, about the same as the effective transmission range) of $n_5$ (namely, $n_3$, $n_7$, $n_8$) would halt their operation and become blocked as they sense that the channel is busy. With CA handshaking methods (like RTS/CTS), also the neighboring nodes of $n_4$ would become blocked. As a result, a large area around the communicating nodes will be reserved for a single transmission, meaning that the nodes inside this area are not able to communicate during the ongoing transmission. In a centralized network case, this kind of operation is needed, but in an ad hoc network case, especially if SS systems are used, potential capacity is wasted, since all the possibly successful parallel transmissions in the reserved area are prevented. Even if CSMA is not used, the nodes around the transmitting node will be blocked, because they all track the common channel and, thus, will automatically try to receive all the transmissions. Of course, when MAC reads the address fields of the received frame and notices that the message is not intended for that node, it discards it (unless a promiscuous mode is used). Thus, even though the user will not see all the messages, PHY will receive them, blocking the node.
As an ad hoc network has a multi-hop distributed structure, in addition to the temporal link capacity, the spatial capacity is also very important. The performance of the whole network should be considered instead of a single link. Blocked nodes can denote unused network capacity and lead to reduced network throughput. Naturally, blocked nodes cannot always be considered to be wasted capacity, since in many cases blocking is required in order to prevent harmful collisions that would intercept the ongoing transmission. In the example (Fig. 12), in addition to the ongoing transmission it should be possible to make simultaneous transmissions (like \( n_1 \rightarrow n_2 \) or \( n_8 \rightarrow n_7 \)) without causing too much interference to the ongoing transmission, depending of course on the accurate link budget.

The problem described is also called the exposed terminal problem, and it is a limitation that cannot be solved properly by network layer methods. It is a major limiting factor of ad hoc networking, causing the performance to collapse quickly as a function of an increasing offered traffic load. By increasing the PHY data rate, the throughput performance might improve in short routes, but the basic problem remains. Unlike the problem of connectivity, this problem can be solved or at least eased by lower layer design without using directional antennas, etc. The key is to allow co-located simultaneous transmissions. However, the main difficulty is to know, in reality, when there can be more transmissions in the same area without corrupting the ongoing transmission(s). This leads to a dilemma:
How can possibly harmful collisions be avoided with distributed control while allowing collisions that do not cause transmission interruptions?

Another, more straightforward, dilemma is how to perform simultaneous transmissions in practice, since this is not doable with the common channel approach. This can be solved by introducing some form of channelization in order to separate the co-located simultaneous transmissions. Spread spectrum systems (like 802.11), which are often used in ad hoc networking, offer processing gain against low-correlating transmissions, and, thus, channelization could be done by CDMA. In DSSS systems, the possibility of simultaneous co-located successful transmissions is dependent on the link budget, which is defined by the received signal power versus the noise and interfering powers (e.g., other users). The use of SS techniques is attractive, because simultaneous transmissions can be done at the same time, frequency, and space, which is exactly what is desired in ad hoc networking. Also, this can be done quite easily by only using different spreading codes. With CDMA, the complex frequency and time slot allocations of FDMA and TDMA, respectively, are not needed.

The use of channelization is illustrated in Fig. 13, where DS-CDMA is used instead of the single common channel as in the previous example. The situation is very different: The performance is limited by Multiple Access Interference (MAI) rather than the basic principle of the channel access. As seen in this simple example, the temporary capacity of this network can be multiplied with channelization.

The challenge is how to achieve channelization while retaining the basic properties of ad hoc networking. In addition, the solution should be quite simple, without putting too much complexity in the transceiver hardware (HW) or making modifications to the communication protocols. One can always find excellent, indefinitely complex solutions, but in this work the interest is in solutions that are simple enough to be even economically feasible.
Fig. 13. An ad hoc network operating with dedicated data transmission channels allows multiple simultaneous co-located transmissions. Figure reprinted from (Prokkola & Bräysy 2007), © 2008 Elsevier.
3 BCCA and BC-MAC

Bi-Code Channel Access (BCCA) is a method designed especially for ad hoc networking to enable a straightforward way to use CDMA without forgetting the special needs of ad hoc networking. This chapter first presents the BCCA method itself, followed by the spreading code distribution method and a dedicated MAC solution for the use of BCCA.

3.1 General functionality of BCCA

3.1.1 Spreading code protocols

There are several ways to use SS and CDMA, and they all have some pros and cons. This way of usage is called a spreading code protocol.

Fig. 14 presents the basic spreading code protocols and their vulnerabilities for collisions (Sousa & Silvester 1988). In C-code protocol, all the nodes use the same code for reception and transmission (cf. common contention channel, 802.11). The channel is, therefore, completely a contention based random access channel, where performance is radically weakened by collisions. The advantage is simplicity, since the nodes only need to be able to transmit and receive with a single code. This fits directly the fundamental idea of ad hoc networking, where access to the network should be easy.

With receiver based spreading codes (R-codes), every node has its unique code for reception. The receiver still does not have to be able to track more than a single code, and at the transmitter side, it is only required that the spreading code must be able to be changed on a per frame basis. Hence, even though the transceiver complexity is almost the same as in the C-code case, successful co-located simultaneous transmissions can be done. A disadvantage is the knowledge requirement for the receiver code of the intended destination node. Also, there is still a possibility for collisions, since multiple nodes transmitting to the same destination node will use the same code.

With transmitter based codes (T-codes), the nodes will have a unique transmission code. Clearly, the performance is the best of the presented spreading code protocols, since no primary collisions can occur. The major disadvantage is complexity, since the receiver must be able to track the codes of all the possible transmitting nodes simultaneously. This means, in practice, that the receiver must
have several parallel channels, while only a single channel is needed for R- and C-code protocols. T-Codes are used, e.g., in cellular systems, as there is no problem for the BS to have extra HW for serving several users. Also, the served users are always known, since before the actual data communications they will register to the BS. However, in the idea of ad hoc networking, there should not be an assumption of knowing all the nodes in the network, or even how many possible neighbors a node will have. Hence, it will be difficult to know how many channels are needed at the receiver. If the extra HW would not be a problem, the next challenge would be to obtain knowledge of the possible transmitting neighboring nodes and to be able to track their codes in the changing network topology without centralized registration.

The problems of T-codes can be alleviated by the use of hybrid methods, where the data portion of the message is sent with a T-code and header either with a C-code (C-T protocol) or an R-code (R-T protocol). The used T-code is informed in the PHY frame header, so the receiver can change the reception of the data to the appropriate T-code after the header is received. Thus, the complex multi-channel receiver structure is avoided, but the drawback is that the header part will now be vulnerable to collisions. In an ad hoc network environment, the use of the sufficiently easily implementable C-T protocol would also mean that the header part transmission would block the neighboring nodes, leading back to the starting point.

![Diagram of spreading code protocols](image)

**Fig. 14.** The most common spreading code protocols (the darker the color, the more vulnerable that part is to collisions).
3.1.2 Functionality of a BCCA based receiver

From the fundamental idea of ad hoc networking, it is clear that a common channel known to all nodes should exist, and for that, C-code protocol is the only practical choice. On the other hand, the use of CDMA would increase the performance considerably. However, T-code protocol is not an option, since the goal was to design a relatively simple method. Thus, to allow CDMA, R-code protocol is chosen.

As a result, a method where a two-channel receiver capable of tracking and receiving transmissions with two different spreading codes is proposed. One channel tracks the C-code and is used as a common channel, and the other channel is a dedicated receiver based code channel. At this stage, it should be pointed out that the dedicated channel here does not mean dedicated in the sense in which it is understood, e.g., in the case of cellular networks. Rather, the R-code channel is dedicated to direct transmission to a single node, but the channel is still a shared channel in the sense that it can be accessed freely without registrations. The common channel is used to broadcast traffic and control information, such as route search, topology and connectivity maintenance, while the R-code channel is used for all directed traffic (e.g., data) transmissions. This method is called bi-code channel access (BCCA), and it was introduced in (Prokkola & Bräysy 2004). Fig. 15 clarifies the idea of using BCCA.

Fig. 15. A time-frequency-code division view of the proposed BCCA method. Figure reprinted from (Prokkola & Bräysy 2004), © 2008 IEEE.
One option would have been to use C-T protocol, as it would provide both easy access and CDMA channeling, and only a single channel at the receiver would be needed. However, in an ad hoc network, and especially in MANET, there is a lot of control traffic in the network (Prokkola et al. 2004, Lee, et al. 1999), and, hence, there is a need for a channel which is dedicated to control and broadcast traffic. Another thing is that the goal is to enable the use of CDMA in such way that a data transmission is not interrupted or is not even seen by nodes other than the intended destination (see Fig. 13). If C-T protocol were used, all the neighboring nodes of the transmitting one would still always see the header part and be interrupted for that time. Hence, the undesired exposed terminal problem would still remain even though it would be somewhat eased. With R-codes, only the intended destination will lock to the transmission, allowing the other neighboring nodes to communicate simultaneously. The dual channel structure will also allow reception with both channels simultaneously, increasing the efficiency of the channel usage.

The R-code channel is still vulnerable to collisions. Because of this, R-code protocol is not at its best in a centralized network structure; also, R-code protocol might not be reasonable if the network load is concentrated on certain single nodes in the network. However, BCCA is designed especially for flat ad hoc networks, where the traffic is typically distributed in nature, and, hence, R-code collisions are rare. Even though there were only a few important server nodes in the network, the multi-hop environment would distribute the traffic throughout the network, and, hence, BCCA would increase the performance also in those situations, though of course not as much as in a completely random session network (this will be seen later in the session model study part). Nevertheless, because R-code collisions are possible, some intelligence is required from MAC to minimize their effect.

The idea of BCCA could be, in fact, implemented in any domain, namely the time, frequency, space, and code domain. Yet, in the case of ad hoc networking, the use of code domain is the most straightforward. DS spreading is preferred in order to avoid extra complexity in the RF/IF blocks of the transceiver. On the transmitter side, only minor modifications are required to allow DS-CDMA as compared to the common channel DSSS. The only requirement is that the used spreading code can be changed on a frame basis, in contrast, e.g., to 802.11, where the spreading code is fixed. With DS, this is straightforward, but with FH, it would be more complex.
At the receiver side, two channels are needed. An idea-level structure of the possible receiver is depicted in Fig. 16. The implementation of BCCA will require an extra branch (R-code branch) at the receiver after the common RF/IF and ADC (Analog to Digital Conversion) blocks, containing a matched filter (MF) (or a correlator), demodulator (if not implemented in the RF/IF-block), detector, and a code synchronization unit. Received channel bits are buffered in order to separate messages received from different channels (codes), since common and dedicated messages can arrive simultaneously. The rest of the receiver is common to both channels and usually includes channel decoding, de-interleaving, etc. Buffering could introduce some minor delay if the traffic to both channels is high, because the arrival of the whole frame has to be waited for before releasing the received data further. This delay and the need for a buffer can be avoided if both separated branches contain all the PHY functionalities after RF/IF. This requires extra HW/SW, but the frames would not need any extra buffering before MAC, in which the processing capacity can be assumed to be sufficient for negligible delay. More aspects of possible implementation are given in Section 5.3.

3.2 Network layer spreading code distribution

It is possible to realize BCCA completely at PHY and MAC, but there are some interesting advantages if the network layer is involved also to the design.

One can assume that the spreading codes are known a priori, which is a fair assumption in, e.g., a military scenario (Prokkola & Bräysy 2004b). However, in commercial and civilian usage, this is a restriction. To release this restriction, a network layer spreading code distribution (NSCD) method operating on top of the routing protocol is proposed.

NSCD was originally presented in (Prokkola & Bräysy 2007), and the idea behind it is quite straightforward. Since the only requirement in BCCA is that the spreading codes of the neighboring nodes belonging to an active route are known,
the codes can actually be distributed in the routing protocol’s route discovery phase. The easiest way to do this in AODV is to modify the RREQ and RREP packets, as shown in Fig. 17. Only one extra field containing spreading code information is required for the RREQ and RREP packets. This field indicates the spreading code of the previous hop node. The information contained in this field could be directly the used spreading code, an index referring to a predefined list of the used code-set (military systems), or perhaps the generation polynomial of the code. Therefore, the length of the field is dependent on the spreading code length and also on the way in which the code information is presented. In the implementation studied here, we distribute directly the actual spreading codes with NSCD. This is the most pessimistic assumption, since there is no compression in the spreading code information. This is a good starting point for a performance study, as we are able to see the effect of NSCD in the worst case.

\[
\begin{array}{c|c|c|c|c|c}
& ctrl\ info & hop\ cnt & broadcast\ id (RREQ_id) & dest\ addr & source\ addr \\
\hline
RREQ & & & & dest\ seq\ num & source\ seq\ num \\
spreading\ code\ info \ (n\ bits) & & & & & \\
\hline
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c}
& ctrl\ info & hop\ cnt & broadcast\ id (RREP_id) & dest\ addr & source\ addr \\
\hline
RREP & & & & dest\ seq\ num & lifetime \\
spreading\ code\ info \ (n\ bits) & & & & & \\
\hline
\end{array}
\]

\[64\text{ bits} + 192 + n\text{ bits}\]

\[64\text{ bits} + 160 + n\text{ bits}\]

NSCD functions as follows (the operation of NSCD is illustrated in Fig. 18):

- When generating RREQ or RREP, information about the generating node’s own R-type spreading code is set into the packet.
- When receiving RREQ or RREP, the information about the previous hop node’s spreading code is acquired from the packet and added to a spreading code table.
- When forwarding RREQ or RREP packets, information about the receiving node’s own R-code is replaced to the packet.

Thus, spreading code information is always checked from and set to RREQ and RREP packets. Therefore, spreading code usage optimization (i.e., neighboring
nodes would choose to use a set of codes with good cross-correlation properties) could be easily carried out above NSCD. Nevertheless, the optimization is not considered in this work. Instead, when NSCD is used, we make the pessimistic assumption that the nodes choose their spreading codes randomly from a certain set. This is straightforward, but certainly suboptimal. For example, with Gold codes the family size is only $L$ (code length) + 2 (Sarwate & Pursley 1980), so with short codes it is quite possible that the neighboring nodes would have chosen the same R-code. Such a way is chosen in order to get an idea about the realistic performance. When introducing a new method, it is not very convenient to show the performance with perfect algorithms and optimal parameters, which are unlikely in reality. The cost of using NSCD is shown as increased control traffic, since RREQ and RREP packets will be longer. However, no extra packets are needed, which would not be the case if the method was implemented at MAC.

To the best of our knowledge, NSCD is a novel method for distributing spreading codes at the network level. Code distribution methods have been studied, but the majority of them operates at the physical or link layer. The usual approach is to distribute the codes with a modified RTS/CTS type handshaking mechanism (Muqattash & Krunz 2003, Wu et al. 2000). This kind of approach does not fit the concept of this work, since it is particularly desired to avoid using complex handshaking mechanisms and additional control packets. Also, it should be noted that one should not mix NSCD with the code assignment or re-coding schemes, which are used for optimizing the spreading code usage (Hu 1993), (Gupta 2001). Rather, NSCD is a method for performing the actual delivery of the spreading codes, and the possible optimization methods can operate above it.

When using BCCA, each node must maintain a spreading code table. The table contains the destination nodes and their R-codes, and, in NSCD, it is initialized to have C-code for all nodes by default. In the assumption of known codes, the table is filled beforehand, and location of the table is not that important. When using NSCD, the C-codes are replaced when information about the dedicated R-codes is obtained, and, thus, it is convenient to keep the table in the network layer. In the implementation here, the spreading code table is maintained by the AODV routing protocol, which guides MAC regarding the spreading code usage on a per packet basis. A separate table is used, but the spreading code information could also be added to the routing table.
Fig. 18. Example: In the AODV initial route discovery phase, \( n_1 \) is searching for a route to \( n_6 \). From the received RREQ, \( n_2 \) learns the spreading code of the \( n_1 \) and adds it to the spreading code table (fig. a). When \( n_2 \) eventually receives RREP, it learns also the code of \( n_4 \), which is needed in the formed active route (fig. b). After the route search, all the nodes know the R-codes of their neighbors belonging to the active route, and, hereafter, the route can operate completely in dedicated channels. In fact, already the RREP’s can be transmitted with R-codes. Figure reprinted from (Prokkola & Bräysy 2007), © 2008 Elsevier.
3.3 BC-MAC

BCCA is a channel access method, and thus it can be used with different higher level MACs, e.g., the one used in 802.11. In order to do this, one replaces the channel access part of the MAC protocol with BCCA, i.e., the Barker C-code system is replaced by BCCA’s bi-code channel system. However, to get most out of BCCA, a MAC designed particularly for BCCA is needed.

Due to the partial random access nature of BCCA, there is a need for CA. But, as the idea of BCCA intrinsically reduces the likelihood of collisions, CA methods can be more loose and simplified when compared to 802.11. This creates prerequisites for fast and more efficient operation. In general, the demands for a MAC in an ad hoc network environment are very contradictory: An efficient MAC should allow as many simultaneous transmissions in a small area as successfully possible but, on the other hand, prevent harmful collisions.

Bi-Code MAC (BC-MAC) is designed especially to be used with BCCA, and it was originally presented in (Prokkola & Bräysy 2004b). The operation of BC-MAC is based on np-CSMA, but the implementation is more straightforward than in 802.11. If the channel is detected to be free, transmission is allowed immediately. No exponential back-off algorithm is used, but only a random deferment period is set when the channel is detected to be busy. The exponential back-off algorithm in 802.11 is grounded on the idea that the majority of packet losses are caused by collisions, and also that when a collision occurs all the colliding packets are lost. However, when CDMA is used, this assumption does not hold. The random deferment period of BC-MAC is of the form

$$t_{\text{defer}} = \text{Unif} \left( \frac{l_{\text{prev_pk}}}{r_d} \right),$$

(9)

where $l_{\text{prev_pk}}$ is the total length of the previously sent PDU, $r_d$ is the physical layer data rate, and Unif($x$) stands for uniformly distributed random variable from [0, $x$]. From the equation, it can be seen that if a long data packet was just sent, the next deferment period is likely to be longer, which gives priority to nodes sending shorter packets to access the channel. The purpose of this feature is to maintain fairness in channel access and increase the randomness of the access method. If the channel is still busy after the deferment period, a new deferment period is set. This process is repeated until the channel is detected to be idle after the deferment period, and, then, transmission is allowed immediately. To promote fairness fur-
ther, a deferment period is always set after transmission to allow other nodes a chance to access the channel.

To increase the efficiency of BCCA, simultaneous transmissions are encouraged. Complex handshaking methods, such as RTS/CTS, are not used. Also, NAV, which in 802.11 is used as a part of the virtual carrier sensing, is not applied to BC-MAC.

Because random access schemes with wireless communications always introduce unreliable links, an ARQ mechanism is needed. In BC-MAC, a stop-and-wait ARQ is used, and the ACKs are sent immediately after data packet reception regardless of the channel state (busy or idle). If, on the other hand, the channel state is checked after a reception and if there is a colliding transmission, the receiver will lock to that after it has finished the ongoing reception. Hence, the ACK for the possibly successfully received packet would be delayed, causing an ACK timeout and retransmission. As a result, this methodology would fail to benefit from the capture effect. The 802.11 standard (IEEE 1999) also defines similar immediate ACK usage, where the channel is not checked before the transmission of an ACK. 802.11 waits for an SIFS before ACK transmission, while BC-MAC sends the ACK immediately, but the principle is the same. Of course, even without any IFS, there is small delay caused by the time required to switch from reception to transmission (Rx/Tx switch time).

IFSs, such as in 802.11, are not used by default. However, packet priority issues, if needed, e.g., for QoS extensions, can be handled by modifying the calculation of the deferment period. Packets with lower priority could be set to have, e.g., a constant deferment period part in addition to the random part, giving high priority packets an advantage in channel contention. Fast ACK response and the lack of exponential back-off and NAV motivate the use of more reTx tries before making a decision on a link failure. 20 reTx tries are used by default. A strict ACK timeout is calculated as

\[ t_{\text{ACK\_to\_s}} = t_{\text{data}} + t_{\text{ACK}} + 2\hat{t}_{\text{prop}} + \hat{r}_{\text{Rx/Tx}} + \hat{t}_{\text{pros}} \]

where \( t_{\text{data}} \) and \( t_{\text{ACK}} \) stand for transmit durations of data and ACK packets, respectively, \( \hat{t}_{\text{prop}} \) is a conservative (link distances are typically shorter) propagation delay estimate, \( \hat{r}_{\text{Rx/Tx}} \) is the average Rx/Tx switch time, and \( \hat{t}_{\text{pros}} \) is an estimate of the overall processing time required to process the received data packet and prepare an ACK for transmission. To increase the randomness (e.g., to randomize the possible colliding sequential transmissions), a random time is added to the strict

\[ t_{\text{ACK\_pros\_to\_s}} = t_{\text{data}} + t_{\text{ACK}} + 2\hat{t}_{\text{prop}} + \hat{r}_{\text{Rx/Tx}} + \hat{t}_{\text{pros}} + \hat{t}_{\text{random}} \]

where \( \hat{t}_{\text{random}} \) is the added random time.
timeout. The random time is calculated as a fraction $\tau_a$ of the strict timeout. Finally, an ACK timeout is therefore given by

$$t_{ACK_{st}} = t_{ACK_{st,s}} + \text{Unif}(\tau_a t_{ACK_{st,s}}).$$  \hspace{1cm} (11)

A randomness of $5\%$ ($\tau_a = 0.05$) of the strict ACK timeout is found with test simulations to function generally well with the studied setups. Thus, this will be used, but it should be noted that $\tau_a = 0.05$ is not an optimized value.

**Fig. 19. General state-chart description of BC-MAC.**

As seen in BC-MAC, the randomness of the back-off process is a result of both deferment period and an ACK timeout, in contrast to 802.11, where an ACK timeout is constant and the randomness is achieved by the exponential back-off algorithm. Fig. 19 depicts a general state-chart presentation of BC-MAC. It presents only the major functionalities, while the real implementation is naturally considerably more complex (e.g., “a need for packet Tx” can mean upper layer packet arrival, end of a deferment period with the condition that Tx queue is not empty, etc.).
When BC-MAC is later discussed, especially in the simulation result analysis part, it means BC-MAC with BCCA as opposed to the pure MAC mechanism, if not otherwise mentioned.

### 3.3.1 Dodging collisions with BC-MAC

By investigating the functionality of BC-MAC and equation (11), it is easy to notice that in the case of a hidden terminal, the time gap between sequential transmission tries is typically not long enough (depending on the packet sizes) to avoid continuous collisions. In 802.11, the effect of exponential back-off is that with an increasing number of reTx tries, the period for the possible transmission expands, decreasing the possibility of collisions. The drawback of this method is that when a packet is lost for a reason other than a hidden terminal (e.g., temporary bad channel), the back-off procedure delays the transmission for nothing. In BC-MAC, such a procedure is, generally, not necessarily required, since BC-MAC is used with wideband SS systems, where the capture effect is strong, relieving the hidden terminal problem.

Anyway, there might be situations involving a strong near-far problem, where the processing gain just is not enough to allow error free reception in the case of a collision (for the node in the worse position). Fig. 20 presents a near-far problem scenario, where one transmitting node is near to the receiving node, giving a strong signal and dominating over a weak signal coming from a node near the limits of the effective radio range. We determine the effective radio range as the range where the signal is just receivable without errors, on average, with only background noise included. If the near-far problem sums with a hidden terminal problem, as in Fig. 20, the scenario becomes unfair to the node further away. The near-far problem can be tackled with transmission power control (see, e.g., (El-Osery & Abdallah 2000), but as this is not included in our study and a MAC cannot only rely to the physical layer in this, a collision dodging (CDo) method is designed for BC-MAC. In this, the aim is to dodge possible continuous collisions by introducing occasionally longer deferment periods in order to give the possible hidden terminal time to transmit its packet during this dodging period.

The method is very straightforward. Normally, ACK timeout is calculated with equation (11), but with CDo there will be a dodging period with probability $p_{CDo}$ after 2 unsuccessful transmissions. The dodging period is simply $3t_{ACK,sw}$. This originates from the fact that in the worst case (assuming equally long transmissions, continuous transmission tries, and two hidden nodes far away from the
receiver), a colliding transmission begins just after the beginning of the transmis-

sion of the hidden terminal, and, hence, both transmissions are destroyed since the
colliding transmission almost completely overlaps the hidden terminal transmis-
sion (see Fig. 21). After this, the hidden terminal initiates a retransmission, but
this will also be destroyed in the worst case, because the beginning of the trans-
mision still overlaps with the end of the colliding transmission. After this, the
transmission of the hidden terminal can be successful only if the colliding termi-
nal dodges that period. Thus, a dodging terminal must wait for its own transmis-
sion time, and two times the retransmission time of the hidden terminal (including
ACK waiting times, etc). The following picture clarifies the worst case situation.

![Fig. 20. The near-far problem.](image)

Naturally, the cost of using CDo is an increased delay, but it is compensated as
there is no need for as many retransmissions as without it. The efficiency of CDo
can be evaluated with a simple calculus. Let there be two contenting hidden ter-
minals, and let us assume that both stations have carried out the 2 first unsuccess-
ful transmissions, and, after this, if the other station is dodging and the other is
not, dodging will be successful. The probability for a successful dodging is, there-
fore,

\[ P_{\text{dodge}} = 2p_{\text{CDo}}(1 - p_{\text{CDo}}), \]  

(12)
where $p_{CDo}$ is the probability of dodging for a single terminal. Hence, after $N_{reTX}$ retransmissions, the probability of dodging failure becomes

$$p_{CDo,\text{fail}} = (1 - 2p_{CDo}(1 - p_{CDo}))^{N_{reTX}}.$$  

Now, with the chosen $p_{CDo} = 0.3$, it can be calculated that after 10 retransmissions the probability of collision dodging failure is as low as $\sim 0.0043$. In this calculation, it was assumed that the packet lengths of the terminals are about equal. If this assumption is loosened, the dodging failure probability will increase. The exact probability is more or less complicated, but in the worst case the probability of dodging failure will be the same as equation (13) without the factor 2. This is because, in the worst case, the dodging of a node with a smaller packet will not be able to prevent the collision. However, even in the worst case, the dodging system is quite efficient.

The used $p_{CDo}$ value is probably not the optimal one, since no exact research on it has been done. Naturally, giving $p_{CDo} = 0.5$ would give the highest probability of dodging success. The situation is, however, not so straightforward, since not all of the reTx failures are due to a hidden terminal, and when not, dodging has only a negative effect of increasing delays. In addition, if there are more than two hidden nodes contending the channel access, $p_{CDo} = 0.5$ is no longer the optimal value even in theory. Hence, a smaller dodging probability is chosen.

The need for CDo is rare in the studied ad hoc cases. The near-far problem probability is dependent on the processing gain, which makes CDo increasingly important with a decreasing spreading factor. Even in the strong near-far case, the need for CDo is not always apparent. Consider, e.g., the case in Fig. 20. If both nodes $n_1$ and $n_3$ transmit simultaneously to $n_2$, the transmission of $n_3$ will be lost.
because of the strong signal of $n_1$. However, the transmission of $n_1$ is likely to be received without problems. As a result, $n_1$ has successfully transmitted its frame, and the next retransmission of $n_3$ is free of collisions. Only if $n_1$ has several packets to be transmitted will the transmission of $n_3$ fail continuously, and only in the case of a high continuous traffic stream from $n_1$ would the transmission of $n_3$ fail completely. In any case, node $n_3$ will suffer from the near-far situation, and this will mainly be seen as increased delay, decreasing fairness. Thus, to promote fairness, the use of CD0 is preferred.

Another situation in which CD0 will help is a case where two nodes hidden from each other are both situated near to the limits of the effective radio range from the intended destination node. In this case, if both nodes transmit at the same time, the MAI will be strong enough to destroy both transmissions. This situation, however, is rare and, again, dependent on the processing gain. Anyway, it is clear that CD0 will enhance the performance in this kind of situation, and especially when dealing with narrow band systems. As a conclusion, CD0 is always used with BC-MAC by default.

### 3.3.2 Cross-layer operation

Basically, BCCA can operate not only with different MACs, but also with different routing protocols. AODV is chosen as the routing protocol because of its on-demand nature and capability to adapt to different conditions (Perkins, et al. 2003). In order to use BCCA, the spreading code information must be stored somewhere. There are basically three options for this: the network, the link, or the physical layer. PHY is not preferred, since it is not practical when considering spreading code management. MAC is possible, but the network layer is chosen because it is the most straightforward when using NSCD.

Since the spreading codes are handled at the network layer, cross-layer functionality is required. The network layer must guide MAC for the spreading code usage, which in turn guides PHY to perform the actual transmission. The spreading code information is stored at the network layer by the routing protocol. The guidance can be implemented by direct inter-layer control messaging or, e.g., so that there is a virtual field of spreading code information in the PDU (see Fig. 22). A virtual field is used only for carrying inter-layer control information and is not sent. The network layer sets the virtual field to the packet, while MAC handles this whole packet as any packet, and hands it to PHY after possible queuing. PHY
rips off the virtual field of code information, tunes the transmitter to a given code, and sends the frame forward.

It must be noted that cross-layer operation does not mean that the idea of layering becomes obsolete. Rather, it only means that the layers will cooperate and communicate to achieve better performance, while the internal layer functionalities are still independent and layer protocols can be changed without making changes to other layers.

![Cross-layer operation in BCCA](image)

**Dynamic ACKs and inter-layer flow control**

Cross-layer operation is also used in controlling link layer ACKs, since the network layer (routing protocol, in practice) can tell with inter-layer control messages which packets need to be acknowledged and which do not. For instance, Broadcast packets (e.g., RREQ) do not need ACKs, so the routing protocol does not ask for an ACK for these. Thus, MAC does not have to have the intelligence to decide on the ACK usage, nor does it have to understand the contents of the upper layer PDU.

The dynamic ACK system also uses a flow control mechanism between MAC and the network layer. Normally, the network layer would operate completely independently from the lower layers, and hence it would push packets without knowing anything about the success of the actual transmissions. In the system here (in both 802.11 and BC-MAC), the network layer operation is tied to the
MAC’s knowledge on transmissions. Thus, the network layer does not push new packets to lower layers until MAC informs that the transmission of the previous packet was successful. This includes also address knowledge, i.e., if, e.g., the network layer has given a packet addressed to node \( n \) to MAC, it must wait for MAC’s indication of the result of the transmission before giving the next packet addressed to node \( n \). However, while waiting for the transmission information of node \( n \), the network layer is free to give packets targeted to nodes other than \( n \). If the transmission is not successful (i.e., the maximum number of reTx has been tried without an ACK), MAC informs the network layer, and the routing protocol will perform the needed actions.

The network layer stores a copy of the packet under transmission. Therefore, a lower layer failure does not lose packets permanently, and the network layer is free to decide how to operate; e.g., it can initiate a route re-establishment operation.

This described flow control operation is beneficial in an ad hoc environment, where rapid topological changes cause frequent link failures. Imagine if the routing protocol just pushed data packets directly to the MAC after route establishment: If a link failed, all the packets at MAC would be permanently lost. However, with the inter-layer flow control, the routing protocol has a chance to find a new route for the buffered packets. This is considerably faster than just relying on transport layer delivery services like TCP (Transport Control Protocol), which typically is too slow for real-time applications.

### 3.4 Theoretical throughputs of BC-MAC vs. IEEE 802.11

In this section, a simple calculation for the theoretical maximum throughputs of 802.11 and BC-MAC in an ideal case is performed. The purpose is to illustrate the limits of the accessible throughput of the studied random access schemes. Throughput limits of 802.11 have been studied also before (e.g., in (Jun et al. 2003), there is a good study indicating the limits of 802.11).

We assume that the data is transmitted correctly and that the upper layer data is not fragmented. We also assume that carrier sensing is ideal. In addition, we make the calculation for a single communications pair (two nodes), and not for a network. Thus, this handles basically only the performance of BC-MAC while there are no actual benefits now from BCCA.

The time that it takes to transmit a packet composed of \( N_{\text{data}} \) data bits and \( N_{\text{overhead}} \) overhead bits is calculated as
From the physical distance $d_i$ between transmitter and receiver, we get signal propagation delay of

$$t_{\text{prop}} = \frac{d_i}{c}.$$  \hfill (15)

Fig. 6 presents a data transmission scheduling diagram for 802.11 without RTS/CTS. From this, we can calculate the total delay of sending a data packet with 802.11 to be

$$t_{802.11} = t_{\text{DIFS}} + t_{\text{data}} + 2t_{\text{prop}} + t_{\text{SIFS}} + t_{\text{ACK}}.$$  \hfill (16)

The actual data throughput (in bits/s) is obtained with the total delay $t_{\text{tot}}$ as

$$r_b = \frac{N_{\text{data}}}{t_{\text{tot}}}.$$  \hfill (17)

Hence, with this equation, the throughput of 802.11 can be easily calculated by substituting $t_{\text{tot}} = t_{802.11}$. With RTS/CTS, the calculation is similar, but the total delay is of the form (see Fig. 7)

$$t_{802.11\text{RTS/CTS}} = t_{\text{DIFS}} + t_{\text{RTS}} + t_{\text{CTS}} + t_{\text{data}} + 2t_{\text{prop}} + 3t_{\text{SIFS}} + t_{\text{ACK}}.$$  \hfill (18)

As expected, the delay structure with RTS/CTS contains several extra terms, which will be seen as decreased data throughput.

BC-MAC, on the other hand, has no additional frame spaces, but only the Rx/Tx switch time must be waited for between the reception of data and transmission of an ACK, so the delay of successful data packet transmission is of the form:

$$t_{\text{BC-MAC}} = t_{\text{data}} + 2t_{\text{prop}} + t_{\text{Rx/Tx}} + t_{\text{ACK}}.$$  \hfill (19)

With the above equations, we can calculate the achievable throughputs as a function of packet size. The normalized throughputs are depicted in Fig. 23. As can be assumed, BC-MAC and 802.11 have almost equal performance, but the complex RTS/CTS mechanism pays for its enhanced hidden terminal prevention ability. This decreased throughput of RTS/CTS will easily become a limiting fac-
tor in a multi-hop network path, where the problems of single links are cumu-
lated. Nevertheless, the overall trend is the same with all setups: The efficiency is
poor with small data packets, but increases as a function of packet size. This is
because of the ARQ system and the overhead in packets. Based on the results, it
can be assumed that BC-MAC will perform similarly to 802.11 in an ideal single-
hop WLAN network, where all the stations can hear each other. However, when
carrier sensing errors, MAI, etc. are included, the situation will be different in
favor of BC-MAC, while the real benefits of BC-MAC, and BCCA in particular,
will be seen in a multi-hop ad hoc network.

![Theoretical throughputs](image)

**Fig. 23.** Theoretical throughputs (normalized to the channel bit rate) as a function of data packet size.

### 3.5 Related work

Early radio access methods were single channel systems, as is the popular IEEE
802.11 today, but the general idea of using separate control and data channels in
radio networks was proposed as early as in 1976 in (Tobagi & Kleinrock 1976).
Even earlier propositions of the principle exist, but they are usually not computer
network related. In the proposed split-channel reservation multiple access
(SRMA), control and data channels are separated in the frequency domain. The
proposed method includes a dynamic data channel reservation mechanism similar
to MACA. Since then, several variations of the split-channel multiple access
based methods with and without data channel reservation have been proposed (e.g., Fitzek et al. 2005). Overall, numerous approaches for different situations have been taken, and, e.g., (Pomalaza-Raez & Alam 2000) evaluates the performance of different channel access protocols for multimedia mobile terminals in a multi-hop network.

In this work, the interests, in particular, are in methods incorporating CDMA. The idea that CDMA could provide an efficient way of communications for wireless LAN’s by allowing multiple simultaneous transmissions did not escape the attention of researchers (Lee & Cho 1995), but CDMA was still not included in the 802.11 standard. Earlier, for the packet radio networks, R-code based CDMA has been studied and analyzed in several publications (e.g., Dill & Silvester 1988, Sousa & Silvester 1988). In (Dill & Silvester 1988), for example, minimum hop routing was assumed with an idealized slotted ALOHA channel access protocol without incorporating the effect of MAI. The benefits of CDMA can be exploited also in a slightly different way: (Fitzek et al. 1998) introduces a method where multiple code channels are used per single connection in parallel for enabling fast packet retransmission.

As regards actual CDMA-based ad hoc network MACs, most of the presented ones incorporate some sort of virtual carrier sensing to minimize the number of collisions. In (Joa-Ng & Lu 1999), for example, an MACA/C-T protocol is proposed, aiming to solve the hidden and exposed terminal problems by exchanging RTS/CTS packets between the source and the destination on a common code while T-codes are used for data transmissions. Dual receiver architecture for DS-CDMA networks was proposed in (Lal & Sousa 1999), where a common code is used for a distributed resource allocation mechanism aimed for power control and data rate adjusting. Another channel is tuned to a unique R-code for point-to-point data transmission. The MAC operation is based on an RTS/CTS mechanism, and the study is restricted to single-hop systems. In a similar fashion, the adaptive acquisition collision avoidance multiple access (AACAMA) protocol presented in (Qiang, et al. 2001) uses a common channel for network organization, code acquisition, and other management tasks, but receiver (or transmitter) codes for data traffic with an RTS/CTS mechanism. (Qiang, et al. 2001) also provides a good review of related channel access protocols. A slightly different principle is presented in (Raghavan & Baum 2000), where a method is proposed for making frequency channel reservations for directed (point-to-point) data traffic using combinations of common and receiver codes in a reserved control channel frequency. However, multi-hop systems are not considered. In (Muqattash & Krunz 2003), a
frequency divided dual channel CDMA ad hoc network MAC solution is presented. RTS/CTS handshaking on a common channel is used for power control and collision avoidance, while terminal specific spreading codes are used for data transmission in the data channel.

An interesting approach is given in (Stemerding 2007). The spatial capacity loss caused by aggressive CA is noticed as in this thesis. The solution, however, is not CDMA based, but instead the study proposes that simultaneous co-located transmissions can take place using multi-channel radios. The idea is in a way similar to what a frequency division version of BCCA would be. However, in (Stemerding 2007), there are no separated control and data channels, which causes connectivity problems if there are too many different channels used in the network. Anyway, with the developed simulator, it is shown that also this approach has great potential.

BCCA and BC-MAC are good examples of the benefits of cross-layered design. Cross-layered design of networking protocol stacks for wireless networks has been considered, e.g., in (Zhang 2002, Chiang 2004, and Conti et al. 2004). Cross-layering is still quite a new research area as such, even though steps towards it (e.g., protocol layer interaction) have been taken earlier. Also, control over multiple layers has been proposed in several studies. For example, (Cooklev 2003) proposes that a QoS manager, having control from source coding to channel coding in addition to channel access, would be used in enhancing QoS. Hence, the term “cross-layering” can mean different things to different people. The layered approach works, and it enables the possibility to change a protocol in one layer without modifications to other layers, which is mandatory in modern communications where networks and devices are very complex, containing a lot of SW and HW provided by different vendors. Thus, the idea of cross-layering should not be to crush the idea of the functional communication layer structure, but rather to provide potentially a standard way of how to convey information between different layers in order to enhance performance and allow flexible design principles for specific systems.
4 Performance study

In this chapter, the performance of BCCA, BC-MAC, and NSCD are studied in various scenarios under different conditions and parameters. The basic point of comparison is the commonly used 802.11. It should be noted that the idea of this work is not to criticize 802.11 itself, but rather just to show that its performance is very far from optimal in an environment it was not designed for. IEEE 802.11 suits AP network scenarios very well but is not at its best in an ad hoc networking environment. It will be shown that in an ad hoc environment, there is a lot of unused potential in the lower layer design.

The study will be mainly carried out by simulations, since building a real mobile ad hoc network of dozens of nodes is not practical at this stage. Simulations are done with the OPNET 10.0 network simulator (OPNET 2008)). A pure theoretical approach, in contrast, would be very complex and would require too many assumptions, calling in question the rationality of the whole study. For instance, even the capacity calculation of a wireless network is a difficult task despite the various simplifications (Gupta & Kumar 2000).

In this performance study, several different situations/cases are run. These different cases are all organized so that the scenario and expectations are first presented. Second, the actual results are shown and analyzed, and, third, discussion and possibly a brief summary are given in the last paragraphs.

4.1 Modeling the network

Before going into the actual simulations the used simulation modeling is discussed. Fig. 24 shows the general OPNET node model that is used in the simulations, including the ISO OSI (Open Systems Interconnection) reference model and the correspondence between them. As seen, the simulation model suits quite well also the reference model, and some layers are modeled more accurately, while others have practically no meaning in this work. Next, the basic modeling principles are described. The described simulation model is the default model that is used in most of the scenarios if not otherwise stated.
4.1.1 Application layer

The functionality of the application layer is modeled by the developed application process module in OPNET. It produces traffic streams with given properties (e.g., interarrival time and packet length distributions). Statistical traffic modeling is used to describe application layer characteristics. Hence, the generated application traffic is well-controlled and does not cause surprises in the simulation.

The overall system behavior is affected by the burstiness of the traffic model, where modern data traffic is usually self-similar, exhibiting burstiness over a wide range of time scales (Leland, et al. 1994). Therefore, heavy tailed source traffic models are preferred in modeling. However, as the main focus of this study is not on the effects of bursty traffic models (see, e.g., (Willinger et al. 1997, Huebner et al. 1998, Prokkola 2001, Prokkola et al. 2004) for this), a more conventional traffic model is chosen for most of the cases. However, since the system must also be able to handle modern data traffic, a section of this work studies the effects of other traffic models.
The default traffic model is as follows: Data packets have a constant length of 4,096 bits (512 B), which is a compromise between a short data packet (high relative control load, small delays, and good fairness) and a long data packet (low relative control load, high delays, and reduced fairness). Interarrival times follow exponential distribution, so the traffic model is a variable bit rate Poisson process (VBR-M). Exponential distribution is also a good compromise, as it gives a good model for the overall traffic, unlike, e.g., the constant bit rate (CBR) model, which is used in several studies (e.g., (Gupta & Das 2001, Perkins, et al. 2001)). The traffic produced by the VBR-M model is bursty but not self-similar, which is good in the sense that self-similarity would bring its own effects to the results, making the analysis of the effects of lower layer methods more confusing.

In the simulation model, the functionality of the presentation layer is represented practically only by the stream between the application module and the application manager module (“app_manager”).

### 4.1.2 Session and transport layers

In the used model, there is no actual session protocol, but, instead, the application manager will make the decisions considering the sessions. In practice, a session means that the application manager chooses the destination node to which the packet stream will be sent according to the given parameters. By default, sessions follow the model of dynamic connections (DC), where a data source node chooses its destination node randomly in the beginning of each session, and the sessions will be changed during the simulation iteration. Session duration is an exponential random variable with an expectation value of 100 packets per session. The DC model is used for most of the study, but there are also some scenarios where different kinds of session models are tested. DC is a fair model for studying the overall network performance, mixing well the traffic streams in the network and giving continuous exertion to routing protocol even in a stationary network scenario. However, the model of static connections (SC) is often used, especially in early ad hoc network simulation studies (e.g., (Perkins, et al. 2001)). In the SC model, the destination node remains the same for the whole simulation iteration, which can result in too optimistic a view of the performance, especially when considering the network control load.

In order to keep the focus on lower layer functionalities, it is not desired that the upper layers will add complexity and more parameters to the currently already quite large set of them. Thus, transport layer functionalities are minimized.
particular, the connection orientated TCP would bring too much effects of its own to the network behavior (consider, e.g., congestion control algorithms), and it is also known that TCP has problems when working over wireless links (Holland & Vaidya 1999). Thus, real transport protocols are not implemented in the simulation model. However, the transport layer can be assumed to exist in the simulation model as UDP (User Datagram Protocol) or UDP-lite (Lightweight User Datagram Protocol, (Larson, et al. 2004)), if one assumes that the UDP fields exist in the data packet. Since CRC (Cyclic Redundancy Check) is not modeled, the assumption of UDP-lite is closer to reality than UDP, but the results should be applicable to both quite well. However, the possible UDP headers are not considered as overhead from the performance point of view, but instead all the information above the network layer is considered to be data.

The developed application manager simulation model is very important also in the statistical sense, since it is responsible for calculating several end-to-end application level statistics: offered load, throughput, delay, jitter, packet loss, etc.

4.1.3 Network layer

Network layer functionalities are also kept simple enough to emphasize the performance characteristics of the lower layers. The network layer is based on AODV routing protocol, and is IP-like, but, e.g., Address Resolution Protocol (ARP) is not used, but instead a common address space with MAC and network layers is assumed. The network layer data packet header contains a 32-bit IP-like source and destination addresses, Type (8 bits), Hop count (8 bits), and Datagram length (16 bits) fields.

Since ad hoc networks are currently most often used in a military environment, we will take this into account when planning scenarios and evaluating results. Thus, to avoid extra control traffic, AODV hello messages are not used. Also, since this thesis is a study on the effects of lower layer protocols, the role of MAC in detecting the status of the links is emphasized. In addition, it can be argued that the use of hello messages goes against the basic idea of on-demand routing. We are particularly interested in the on-demand nature of AODV, but when hello messages are used, the routing has also a proactive nature, as there is continuous information exchange on the local connectivity.

Setting the AART parameter correctly is very important. The purpose of this parameter is to remove potentially obsolete routes from the route tables. In a highly mobile network, this prevents nodes from trying to use routes that no long-
er exist (broken routes). The problem, however, is that when this prediction fails, e.g., the parameter value is too small compared to the mobility, functional routes are unnecessarily deleted and new route establishments will be needed, leading to decreased performance. One example is a completely stationary network, where the route timeout functionality only ruins the performance of the network, since the topology does not change. By default, this parameter will be set to infinity, letting MAC handle the detection of broken links. However, there is also a study on the effects of this parameter and mobility.

The used OPNET model of AODV was originally developed by the Wireless Communications Technologies Group (WCTG) at the National Institute of Standards and Technology (NIST) (Guamari 2001). Some bug fixes to this model have been made (e.g., in the original model, the active route usage updating system did not work correctly). Separated send buffers the size of 64 packets for each destination node are used, while the buffer is often common to all transmissions. Also, some modifications have been made in order to enable the AODV model to be used with the other models. The implementation of NSCD, in particular, needed some programming, but this was more like an addition, while the AODV protocol functionality itself remained unchanged. Moreover, new statistic collection functionalities have been added.

4.1.4 Link layer (MAC)

The link layer, which in practice means MAC in this case, will be BC-MAC or 802.11 MAC. Several different cases will be tested, e.g., 802.11 will be driven in a pure CSMA/CA mode and an RTS/CTS handshaking mode. BC-MAC will also be tested with various parameters and modes. Interesting comparison is also made with enhanced 802.11 with a BCCA functionality.

The OPNET model of 802.11b is based on the version provided with the OPNET 8.0 simulator. Several modifications to this have been made in order to enable it to be used with AODV. Also, the original model did not take into account the effects of a wideband channel, but this has been corrected. PLCP modeling is included, and some bugs have also been corrected. In addition, 802.11 BCCA capability is added to the model.

In order to make fair comparison between BC-MAC and 802.11, the frame formats are set to be equal according to 802.11 (224-bit MAC frame (ad hoc mode) and 184-bit PLCP).
4.1.5 Physical layer

The physical layer is based on 2.4 GHz DSSS with properties much similar to those of 802.11, but, of course, BCCA brings some differences, as described earlier.

Since ad hoc networks are mainly used in a military environment, physical layer parameters are partially chosen on this basis. The channel bit rate is set to 1 Mbit/s. The use of 1 Mbit/s is straightforward, since it uses the well analyzed DBPSK (Differential Binary Phase Shift Keying) modulation. Also, in 802.11b the control part is always sent at this rate, making the control and data part transmission equal, easing the interpretation of the results and analysis. To allow potential for interference suppression, and to enhance LPD (Low Probability of Detection) and LPI (Low Probability of Interception) properties, a sufficient spreading factor is needed. A spreading code length of $N_c = 63$ chip is chosen. With bitwise spreading, processing gain $G_p$ is directly 63 (18.0 dB), giving the chip rate of 63 Mchip/s. Of course, the actual processing gain is dependent on the properties of the spreading codes, but for simulation simplification, transmissions with other spreading codes are considered Gaussian noise from the perspective of the signal under reception, and no specific code family is assumed here. It is further assumed that the cross-correlation between the codes at all phases is $1/N_c$. Not much generality of the results is lost with this approximation, because there are several code families with well-controlled cross-correlation properties (e.g., Gold-codes (Sarwate & Pursley 1980)).

To make fair comparisons between BC-MAC and 802.11, the same spreading factor is also set to 802.11 instead of the default 11. Since the spreading factor will naturally affect the performance, a special case study is also made on the effects of the spreading factor.

One might wonder about the relatively low channel bit rate of 1 Mbit/s, since modern WLAN systems are capable of data rates of $\gg 10$ Mbit/s. The chosen data rate does not, however, restrict the applicability of the results to higher data rates also, since the study is mainly performed as a function of a normalized offered traffic load, and some of the performance metrics are also normalized. The higher data rate means that there is less energy per bit, which in effect will be shown as a reduced effective transmission range. Also, the spreading factor will become smaller with an increasing data rate (if the chip rate is kept constant). Hence, the simulated results can be, in some sense, applied to higher data rates also, if the ratio between the data rate, transmission range, and spreading factor is...
kept constant. Naturally, one must also take into account the packet lengths, since the packets will be sent faster with higher data rates. Thus, higher data rates could also have been chosen for this study, and the idea of BCCA itself is not restricted to the chosen data rate case. However, one should also bear in mind that the modern WLAN standards use the available bandwidth quite thoroughly for maximizing a single link data rate. Thus, in those there is practically “no room” for a CDMA-like operation, and BCCA cannot be made in the code domain. Modern WLAN standards like 802.11g and 802.11n have high maximum data rates, but since there are no mechanisms to aid multi-hop environment operation, the performance will most likely behave similarly to that of the basic 802.11: The performance deteriorates quickly as the hop count increases. In IEEE 802.11 standards, there is also 802.11s, which is an amendment to 802.11 for mesh networking, but since this is still in draft stage, it is not included in the study. IEEE 802.11s has several good ideas which ease the practical setup and operation of mesh/ad hoc networks. However, in its basic form, there is not much new on the channel access side, meaning limited performance in a multi-hop environment, but there are optional features, which allow, e.g., the use of several frequency channels and several transceivers, easing the multi-hop situation (Walke, et al. 2006). One also has to remember that this is a work for presenting and proving the basis of the BCCA functionality, while on later stage, the proposed systems can be further developed. In particular, there are no limits why e.g., adaptive modulation could not be used with BCCA as well. Thus, when there is available bandwidth and not many users in the network, BC-MAC could also use higher order modulations or less spreading to maximize the data rates. Moreover, BCCA can be built on top of a MC-CDMA (Multi-Carrier CDMA) PHY enabling easier dynamic data rate management.

In the original OPNET models, only temporary received signal power is provided by the OPNET’s receiver module. For modeling power based CSMA accurately, the total channel power (a cumulative value) is needed. For this purpose, we also developed a module (“received_power” in the simulation node model of Fig. 24), which gives the total power level seen in the channel by the node.

**Radio channel modeling**

A typical feature of ad hoc network simulations is that the physical layer and, in particular, radio channel models are highly simplified. An often applied model is a cut propagation model, where the signal propagates according to some simple
propagation model to a predefined distance, where the signal is cut (e.g., (Iwata et al. 1999, Perkins, et al. 2001)). This kind of propagation model is also referred to as the unit disk graph (UDG) model (Stojmenovic, et al. 2005). Real propagation models make the simulation considerably heavier, since the link budget must always be calculated from each transmitting node to all the other nodes in the scenario instead of just the neighboring nodes defined by the absolute propagation range.

According to the free space propagation loss (FSL) model, received signal power is of the form (Sounders 1999)

\[ P_R = P_T G_R G_T \left( \frac{\lambda}{4\pi} \right)^2 d_l^{-2}, \]  

(20)

where \( P_T \) is the transmission power, \( \lambda \) is the wavelength of the carrier, \( d_l \) is the path length, and \( G_R \) and \( G_T \) are the antenna gains of the receiver and the transmitter, respectively. From the equation, it can be easily seen that if the path length is doubled, the signal strength drops only about 6 dB. This has a major effect in ad hoc networks in increasing the interference level and, thus, it should not be ignored. In this work, the FSL model will be mainly used. By default, fading is not used, but its effects are studied in a special scenario. FSL is not the most realistic model for ground communications, but it is a good model for comparing different setups because of its fairly pessimistic MAI properties. Also, a case with different propagation models is examined, and it will be seen that cut propagation modeling leads to far too optimistic results.

The used modulation method is DBPSK, in which the bit error probability can be calculated as (Proakis 2001)

\[ BER_{DBPSK} = \frac{1}{2} e^{-\gamma_b}, \]  

(21)

where \( \gamma_b \) stands for signal-to-noise ratio (SNR) per bit. SNR can be calculated by link budget calculation with a path loss model when the noise power (background noise plus possible interference) is known. If interfering signals are present, SNR is usually referred to as SINR (signal-to-noise and interference ratio). By assuming MAI as uncorrelated noise, the total interference power observed by node \( k \) can be calculated simply as
\[
P_{\text{rk}} = \sum_{i=1, i \neq k}^{N} P_{ri},
\]
where \( P_{ni} \) is the received power from the node \( i \) (= 0 if the node is not transmitting). The above definition assumes *Gaussian approximation*, which is valid for long pseudorandom codes, and gives quite good results for small SINR values (Lehnert & Pursley 1987, Sousa 1990). Background noise power is calculated as
\[
P_{n} = kTB,
\]
where \( B \) is the receiver bandwidth and \( T \) is the receiver effective noise temperature (assumed to be 290 K) and \( k \) is the Boltzmann’s constant \( (1.3805 \times 10^{-23} \text{ J/K}) \).

Let us now consider how to set the effective radio range \( d_{rr} \). By taking into account the assumed error correcting capability of the error correcting code (in this study, target bit error ratio \( p_{\text{ber}} = 5\% \) is used) and processing gain \( (63) \), we get that the received signal power should be
\[
P_{r} \geq \frac{P_{n} \gamma_{b}}{G_{p}},
\]
where \( \gamma_{b} \) is the corresponding target SNR for the \( p_{\text{ber}} \). By substituting (20), we get the required transmission power for error free communications (average) to be of the form
\[
P_{t} \geq \frac{P_{n} \gamma_{b}}{G_{p} G_{R} G_{t}} \left( \frac{\lambda}{4\pi} \right)^{2} d_{1}^{-2} \geq \frac{\ln(2 p_{\text{ber}}) kTBd_{1}^{2}}{G_{p} G_{R} G_{t}} \left( \frac{\lambda}{4\pi} \right)^{2}.
\]

The effective radio range is set in this study by scaling the transmission power (alternatively, it could be set by scaling the channel bit rate). The desired \( d_{rr} = 250 \) m, so we get the transmit power to be as low as 6 \( \mu \text{W} \) (antenna gains are set to 0 dB) with (25).

With the presented setup, the signal reception threshold is \(-92.3 \text{ dBm} \) after despreading, meaning that the radio channel power level threshold for the reception is \(-110.3 \text{ dBm} \). The carrier sensing threshold is set equal to this, and, hence, the sensing range equals to \( d_{H} \). This is a fairly realistic assumption, but often the sensing range is assumed to be longer than the signal reception range (e.g., in
Having a much lower carrier sensing threshold than reception threshold is sometimes referred to as aggressive carrier sensing.

### 4.1.6 Mobility modeling

Sensor networks are typical examples of stationary ad hoc networks where no mobility is included (Akyildiz et al. 2002). However, in most cases of ad hoc networking, mobility is present. Mobility is one of the things that make ad hoc networking challenging, since the topology will be changing continuously. Mobility is a problem, which must be mainly solved by network layer methods (routing protocol, in particular). Nevertheless, lower layer solutions also have effects on the performance under mobility, and, most importantly, the lower layers must not have any restrictions on the mobility.

The mobility model used in this study is based on the random waypoint mobility (RWM) model used in several other studies (e.g., (Royer & Toh 1999, Das et al. 1998)). In this model, nodes are at first placed randomly in a defined area. When the operation begins, a node randomly selects a new position inside the defined area, and then starts to move towards that position at a randomly chosen speed. At the destination point, a node waits for the duration of pause time before its next movement. Thus, in RWM, pause time and speed distribution define mobility properties. In the actual OPNET implementation, the mobility simulation module is located at each network node (see Fig. 24).

In this thesis, two cases will be used: no mobility (pause time ≥ simulation iteration time) and continuous mobility (pause time is 0 s). It must be noted that the initial positions are chosen randomly also in a stationary network if not otherwise stated. The mobility model has been rebuilt on the basis of the original mobility model by NIST. With RWM, random speed is typically chosen from 0 to $v_{\text{max}}$ m/s, but in this work, there is a limit for minimum speed ($v_{\text{min}}$ m/s) to avoid inconsequential movement during simulations. Consider the following: According to the model, the node moves to the randomly chosen position at the randomly chosen speed. If the speed range starts from 0 m/s, a node might get a speed very near to 0, in which case it can take the whole simulation time for the node to move to the destination. Also, if the node chooses a higher speed, it reaches the destination faster, where it again can choose a speed near to 0. Hence, in a long simulation, it becomes probable that several nodes will be crawling slowly forward, thus almost disabling the mobility of those nodes. Hence, the mobility model cannot be considered very stable, and a lower limit is needed. To back up this reasoning, this
phenomenon of the RWM has been also proven in (Yoon, et al. 2003), which also proves that setting the lower limit to \( > 0 \) stabilizes the model.

In this study, \( v_{\text{min}} = 0.2 \). The upper limit is dependent on the desired modeling case. In most of the cases, \( v_{\text{max}} = 6.0 \text{ m/s} \), presumably representing pedestrian and light vehicular (MLV) traffic.

The RWM model has been criticized (e.g., in (Yoon, et al. 2003) and (Chu & Nikolaidis 2004)), and there are also models seeking more realistic mobility (e.g., (Bettstetter 2001)). It is known that the distribution of the nodes using this model is not uniformly random anymore; instead, the node density tends to increase towards the center of the operational area (Chu & Nikolaidis 2004). In our study, this does not really matter, since the uniform node distribution is not a requirement. Of course, RWM might give a somewhat more optimistic view of the performance when compared to the uniform node distribution case, since the average path length of the routes will decrease, thus easing the task of the routing protocol. This will have a difference and should be taken into account when comparing the performances in stationary and mobile scenarios. In addition, if the absolute performance of a certain scenario is of interest, mobility models accurately suiting that particular scenario should be considered. However, when comparing the performances of different setups (e.g., MACs) in a mobile test scenario, the mobility model is the same for both, making an equal environment for comparison. Thus, the author believes that the used mobility model works well in testing and comparing different systems.

4.1.7 How to simulate?

Simulation principles

The network performance can be studied in several ways, as described in Table 3. For example, one has to choose whether to study the overall average performance or the performance of a single node in the network as a function of simulation time. In the latter case, more exact information is obtained from the single node point of view, and time varying performance characteristics are also revealed. However, this does not give a good insight considering the performance of the whole network, since the QoS from a single node point of view can differ a lot between nodes depending on the network topology. Also, handling and analysis of the results becomes difficult when the effect of some parameter (e.g., offered traf-
fic load) needs to be examined, since the number of simulated results (individual graphs as a function of simulation time) will explode.

Table 3. A comparison of some different ways of studying network performance.

<table>
<thead>
<tr>
<th>Way of studying</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>The performance of a single node as a function of simulation time</td>
<td>Temporal view of the node's performance is gained</td>
<td>Overall network performance cannot be studied</td>
</tr>
<tr>
<td></td>
<td>Accurate analysis of a single node can be done</td>
<td>The effect of different parameters is difficult to analyze</td>
</tr>
<tr>
<td></td>
<td>Simulation is simple and fast</td>
<td>Statistical verification cannot be made</td>
</tr>
<tr>
<td>The averaged performance over all nodes in the network as a function of simulation time</td>
<td>Temporal view of the network performance is gained</td>
<td>Accurate analysis of a single node cannot be done</td>
</tr>
<tr>
<td></td>
<td>An idea about the overall network performance is given</td>
<td>The effect of different parameters is difficult to analyze</td>
</tr>
<tr>
<td></td>
<td>Simulation is relatively fast</td>
<td>Statistical verification is difficult</td>
</tr>
<tr>
<td>The averaged performance of the whole network over simulation iteration time as a function of parameter of interest</td>
<td>The performance of the network can be analyzed directly from some parameter point of view</td>
<td>Temporal analysis cannot be made</td>
</tr>
<tr>
<td></td>
<td>Overall view of the network performance is gained</td>
<td>The information of single node performance is lost</td>
</tr>
<tr>
<td></td>
<td>Statistical verification can be done</td>
<td>Simulation is slow</td>
</tr>
</tbody>
</table>

If one chooses to study the overall performance, the way of simulating and examining the results is still not straightforward. By averaging the simulated results over all the nodes in the network, the overall performance can be seen as a function of simulation time. However, the analysis of the effects of a certain parameter is again difficult. Hence, the usual way to get an idea about the overall performance as a function of an interesting parameter is to make sets of simulations by changing the parameter and collecting only the time average values of the results. As a result, graphs with the performance metric of interest can be drawn as a function of the desired parameter, making the presentation illustrative. In addition, the effect of secondary parameters can be plotted as different curves. The drawback of this kind of analysis is that the temporal performance of the network or single node behavior can no longer be evaluated.

Many of the ad hoc network studies are carried out as a function of mobility (in particular, as a function of RWM pause time, (Boukerche 2001), (Broch et al. 1998)). The reason for this is that there are a lot of differences in how the routing protocols will perform under varying mobility. Hence, mobility is an important
parameter, but in the author’s opinion, the most important parameter is the traffic load, since the whole idea of a data network is to convey application data traffic. Mobility can be added to the study as a secondary parameter.

As a result, in this work performance is mainly studied as a function of a mean offered data traffic load (typically measured from the application layer), which is set by varying the average interarrival time. Traffic is normalized to the transmitter channel bit rate, meaning that value 1 corresponds to a situation where the network’s overall application layer offered traffic load equals the channel bit rate.

A decision about the basic simulation principle is now made, but how to actually perform this is the next thing to consider. Let’s assume that we would like to study the network performance with $G$ in the range of $0.01 – 1$. Next, we select the points (single values of $G$) to be simulated as, e.g., 0.01, 0.03, 0.1, 0.3, and 1. Then, since we have decided mainly to use a fixed packet size, we will calculate the average interarrival times that will give the desired $G$. If we assume the given parameters (1 Mbit/s channel bit rate, 4096 bit packet size) and 20 active nodes (active node = generates data traffic), we get the interarrival times as presented in Table 4.

| Table 4. An example of calculating average offered traffic loads of a 20-node network. |
|---------------------------------|-----------------|-----------------|-----------------|-----------------|
| Normalized offered traffic load (G), simulated point | Offered traffic load [bit/s] | Offered traffic load (single node) [bit/s] | Packet interarrival time (single node) [s] |
| 0.01  | 10,000  | 500  | 8.192  |
| 0.03  | 30,000  | 1,500  | 2.7307  |
| 0.1  | 100,000  | 5,000  | 0.8192  |
| 0.3  | 300,000  | 15,000  | 0.2737  |
| 1  | 1,000,000  | 50,000  | 0.08192  |

As we are studying the averaged performance over the network, a single simulated iteration represents only a single ad hoc topology (with a single set of other random variables) of the infinite possible ones. Hence, the results of a single value of $G$ must be averaged over different topologies. In practice, this is done by performing $N_{\text{iter}}$ simulation iterations per point with the same parameters by modifying only the random generator seed value of the simulator. Therefore, each simulated point will be an average of the results over several iterations.

Next we must decide how to set the simulation length. The usual way is to set a constant length for the simulation. However, since we now have a different in-
terarrival time at each point, constant simulation length would mean that the number of simulated packets would be different for each simulated point. This is not desired, since the accuracy of the results would also vary as a function of offered traffic. From the statistical point of view, this could be compensated for by giving more averaging iterations to the ones with fewer packets. This kind of calculus, based on the previous example, is shown in Table 5. In this example, a simulation length of 1,000 s was set, and about 2,500,000 packets are desired for each simulated point.

Table 5. An example of calculating simulation parameters with a constant simulation length (1,000 s) of an ad hoc network with 20 nodes.

<table>
<thead>
<tr>
<th>Normalized offered traffic (G)</th>
<th>Interarrival time (single node) [s]</th>
<th>Simulated packets per iteration</th>
<th>Averaging iterations N_{iter}</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>8.192</td>
<td>2,441</td>
<td>1,000</td>
</tr>
<tr>
<td>0.03</td>
<td>2.7307</td>
<td>7,324</td>
<td>333</td>
</tr>
<tr>
<td>0.1</td>
<td>0.8192</td>
<td>24,414</td>
<td>100</td>
</tr>
<tr>
<td>0.3</td>
<td>0.2737</td>
<td>73,242</td>
<td>33</td>
</tr>
<tr>
<td>1</td>
<td>0.08192</td>
<td>244,140</td>
<td>10</td>
</tr>
</tbody>
</table>

As seen from Table 5, with G of 1, only 10 iterations need to be run, but the number of simulated packets per iteration (N_{iter, pk}) is very high (244,140). The situation under a small traffic load is reversed. We can now speculate what kind of view of the system performance this way of simulation will give. Consider, e.g., the point 0.01, where 2,441 packets are simulated, giving only 122 packets per node. Recalling the nature of ad hoc networking, 122 simulated packets per node is not enough, since the initial route search will dominate in the beginning of the communications. At the other end, we have a simulation with a large number of packets (244,140), which should be well enough for revealing the average network behavior. However, in this case we have only 10 averaging iterations, which is not much for giving averaging over different scenarios. This is the case especially if a stationary network is simulated, since there will be no topological averaging during the simulation run. As a result, with different values of G, there will be unequal averaging, and the initial route search will affect differently, making the analysis challenging.

On the other hand, if we consider mobility, the nodes will be able to move the same distance (on average), since the simulation length is constant in all simulated points. Thus, this way of simulating is fair in relation to mobility. However,
there are still different numbers of averaging iterations, which, naturally, will result in different accuracies of the different points in terms of mobility.

These drawbacks can be eased by increasing the number of simulated packets and the number of averaging iterations until all the requirements are at a satisfactory level. In practice, this is not possible, since the required computational power is not reasonable, and the simulations would last impractically long. For instance, simulation iteration initialization and result collection will take some time regardless of the iteration duration. Hence, simulating thousands of iterations per point will be very slow even though the iteration itself was short. At the other end, simulating hundreds of thousands of packets per iterations is itself slow under a high traffic load (a lot of collisions and other events to be calculated).

There is also another option of setting the scenarios: keep $N_{iter, pk}$ and $N_{iter}$ constant over the simulation. This will result in the simulation iteration length being different in each point, as shown in Table 6, where the number of simulated packets per iteration was chosen to be about 74,000 and number of iterations to be 33. Now we have constant accuracy for the traffic in all the points, and no unnecessarily long simulations or large number of iterations need to be to run. When considering mobility, we have also an equal number of scenarios to run for each point. Nevertheless, there is the drawback that, as the iteration length will vary, also the duration of mobility, i.e., the average moved distance, will be different in each point.

<table>
<thead>
<tr>
<th>Table 6. An example of calculating simulation parameters with variable simulation length of a network with 20 nodes ($N_{iter, pk} = 73242$, $N_{iter} = 33$).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normalized Offered traffic load (G)</td>
</tr>
<tr>
<td>0.01</td>
</tr>
<tr>
<td>0.03</td>
</tr>
<tr>
<td>0.1</td>
</tr>
<tr>
<td>0.3</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

Both methods have their advantages and disadvantages. A choice needs to be made about whether to simulate mobility behavior accurately and lose accuracy in traffic or accept some loss of accuracy in mobility and simulate traffic accurately. From the point of view of this study, the latter method of having a variable simulation iteration length is more favorable, since traffic is considered to be more important than mobility. And, in fact, the drawback of mobility inaccuracy is not
as severe as it seems. If we consider typical network behavior, a functional network has no problems in handling low traffic loads. Under a low traffic load, the simulation iteration is longer, and, hence, the mobility modeling will be more accurate. Thus, the effects of mobility will be accurately simulated under a low traffic load, just where it is needed, since it will be the main reason for decreasing performance. With an increasing traffic load, the traffic itself will increasingly dominate the performance, finally leading to a situation where the traffic congests the network regardless of the mobility. Hence, the need for mobility accuracy diminishes with an increasing traffic load, which suits exactly this simulation principle.

It should be noted that the chosen point of normalization for the offered traffic load, the channel bit rate, is not typically equal to the network capacity. This is especially the case in a random ad hoc network, where the operational area is larger than the transmission range of a single device. Network capacity was not chosen, because it is very hard to define accurately in an ad hoc network. Since there can be several transmissions ongoing in the network simultaneously, it is basically possible that overall throughputs over 1 would occur with the chosen normalization. However, due to multiple hops and the nature of an ad hoc network, this is very unlikely in a larger random network even with BCCA. However, ideally thinking, if a network with \( N \) nodes would form communicating pairs, in which the other node sends traffic with a maximum bit rate to the other node which only listens, it is possible (theoretically) to reach a throughput of \( \frac{N}{2} \). This would of course require that the communicating pairs did not interfere with each other, that there was no packet loss, no overhead (headers) or ARQ, and that the paired nodes were within the radio range of each other (no multi-hopping). There are no limits in the amount of offered traffic load, since it is measured at the application layer, who can push as much traffic to lower layer as the computational power allows, nor is there any flow control between the application layer and the layer below (in this work). However, obviously, a single node’s PHY layer offered traffic load has a maximum of 1, and, thus, the network’s overall maximum PHY layer offered traffic load would, theoretically, be \( N \) without CSMA. If an application in a single node generates traffic at a rate of \( G > 1 \), the PHY layer cannot transmit it all, and the excess traffic will be queued in the lower layers and finally dropped when the buffer space is exceeded.
Simulation scenarios

As the simulation principle is now chosen, the next question is: What is the required accuracy, i.e., how many packets need to be simulated per iteration, and how many iterations per simulated point are needed? The answer is not very simple and depends on many factors, one of which is the distributions of the performance metrics in the studied cases. Naturally, the distributions are unknown, since highly dynamic and complex systems are being studied, so something less accurate has to be settled for. Another restriction is the computational power, which limits the simulation time.

First, we will decide the simulation iteration length. Because of the initial route establishments and other differences in the beginning of the simulation, iteration should contain at least several thousands of packets per node. The next requirement comes from the desired accuracy of the performance metrics, among which the packet loss ratio is perhaps the most important when considered from the statistical point of view. This is because packet loss has a direct relation to the number of simulated packets \( N_{\text{sim_pk}} \), since the resolution of packet loss is naturally

\[
P_{\text{loss_res}} = \frac{1}{N_{\text{sim_pk}}}.
\]

For this study, it is desirable for the packet loss resolution of a single run to be the order of \( 10^{-5} \), so about 100,000 packets are needed per iteration. Slightly less will be enough, since the averaging iterations will increase the overall packet loss resolution further.

The next step is to determine the required number of averaging iterations to achieve reasonable accuracy. For this, we can use basic statistical methods like the standard error (SE) (Wonnacott & Wonnacott 1990)

\[
SE = \frac{\sigma}{\sqrt{n}},
\]

where \( \sigma \) is the standard deviation of the process under investigation, and \( n \) is the number of samples. As the process of the performance metrics is not known, not to mention the standard deviation, we must use the estimator \( s \) of \( \sigma \), i.e., calculate the standard deviation from samples. With SE, we can calculate the typically used 95% confidence intervals for the process as (Wonnacott & Wonnacott 1990)
\[
\left[ \bar{X} - t_{0.025} \frac{s}{\sqrt{n}}, \bar{X} + t_{0.025} \frac{s}{\sqrt{n}} \right],
\]

where \( \bar{X} \) stands for the calculated sample mean, and \( t_{0.025} \) is the tail of the Student’s \( t \) distribution with \( n - 1 \) degrees of freedom. Normal distribution could be used if \( \sigma \) was known and, also, if \( n \) was large (> 100), since the \( t \) distribution approaches normal distribution when \( n \to \infty \). (Wonnacott & Wonnacott 1990)

Next, we will test how the packet loss ratio is averaged with an increasing number of iterations in an ad hoc network of 20 nodes with BC-MAC and AODV (the network used in Chapter 4.5). Fig. 25 shows the effect of increasing averaging iterations to the average packet loss ratio with \( G = 0.4 \). In the figure, individual samples, the sample mean, the sample standard deviation, standard errors, and 95% confidence interval bars are shown. As seen, the mean value is very uncertain with a low number of iterations, SE is high, and the confidence interval is wide. Yet the mean packet loss ratio starts to settle quite quickly with an increasing number of iterations, and after about 15 iterations, there are no more significant variations. However, the confidence interval narrows quite slowly after the high initial undulation and remains at the level of \( 10^{-4} \), but due to the reasonably steady behavior of the sample mean, we are satisfied with the accuracy of about 20 iterations. The situation is about the same in other cases (e.g., different traffic loads, 802.11) also (not shown).

Fig. 25. An example of the effect of averaging iterations on the average packet loss ratio with \( G = 0.4 \) (20-node ad hoc network).
Based on the above discussion, we will use 32 iterations, which should be enough for reasonable accuracy, while still keeping the simulation time at a reasonable level. The number of simulated packets per iteration is chosen to be 93,750 on average (this is chosen instead of, e.g., 100,000 in order to be able to give rational values to the simulation iteration length and interarrival times). Hence, the $P_{\text{loss res}} = \frac{1}{2812500} = 3.6 \cdot 10^{-7}$, which is sufficient for this study.

The simulation principles and parameters are now about chosen. Table 7 presents a list of the most commonly used parameters (i.e., the default parameters). Thus, if not mentioned otherwise, these parameters will be used in the simulations.

Table 7. The default set of the most important parameters used in the simulations.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall simulation parameters:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mobility model(s)</td>
<td>no mobility, RWM</td>
<td></td>
</tr>
<tr>
<td>Node speed</td>
<td>0.2–6.0 m/s</td>
<td>MLV</td>
</tr>
<tr>
<td>Simulation iteration length</td>
<td>~ 93,750 packets</td>
<td></td>
</tr>
<tr>
<td>Averaging</td>
<td>32 runs per simulated point</td>
<td></td>
</tr>
<tr>
<td>Traffic parameters:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Traffic model</td>
<td>VBR-M</td>
<td></td>
</tr>
<tr>
<td>Interarrival time</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td>Data packet length</td>
<td>4,096 bits</td>
<td></td>
</tr>
<tr>
<td>Session model</td>
<td>DC</td>
<td></td>
</tr>
<tr>
<td>Routing parameters:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Routing Protocol</td>
<td>AODV</td>
<td></td>
</tr>
<tr>
<td>Node traversal time</td>
<td>0.08 s</td>
<td></td>
</tr>
<tr>
<td>Hello interval</td>
<td>Infinity</td>
<td>Not used</td>
</tr>
<tr>
<td>AART</td>
<td>Infinity</td>
<td>Not used</td>
</tr>
<tr>
<td>MAC parameters:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAC(s)</td>
<td>BC-MAC, 802.11</td>
<td></td>
</tr>
<tr>
<td>Carrier sensing basis</td>
<td>RF power detection</td>
<td></td>
</tr>
<tr>
<td>Carrier sensing range</td>
<td>~ 250 m</td>
<td></td>
</tr>
<tr>
<td>BC-MAC reTx tries</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>PHY parameters:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tx power</td>
<td>6 μW</td>
<td></td>
</tr>
<tr>
<td>Modulation</td>
<td>DBPSK</td>
<td></td>
</tr>
<tr>
<td>Processing gain</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>SS method</td>
<td>DSSS</td>
<td></td>
</tr>
<tr>
<td>Error correcting</td>
<td>5%</td>
<td>channel coding</td>
</tr>
<tr>
<td>Frequency band</td>
<td>2.4 GHz ISM</td>
<td></td>
</tr>
<tr>
<td>Radio channel characteristics:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radio Range</td>
<td>~ 250 m</td>
<td>an effective value, not a cut signal</td>
</tr>
<tr>
<td>Channel model</td>
<td>FSL</td>
<td></td>
</tr>
<tr>
<td>MAI calculation</td>
<td>true link budget</td>
<td></td>
</tr>
</tbody>
</table>
The study includes various scenarios. In the first one, a simple access point scenario is studied. After this, a step towards real random ad hoc networks is taken with a stationary random grid scenario of 16 nodes. Real random ad hoc networks include a 20-node basic network, a 50-node oblong network (high average hop count), and a 50-node dense network. In addition to basic performance study, various special cases are investigated. These include a traffic scenario study including an internet call scenario, a radio channel model study, a spreading factor study, and an AODV parameter study. In addition, in some scenarios, special situations are considered, e.g., examining the performance behavior of individual nodes as a function of simulation time (accurate node behavior).

### 4.2 Performance metrics

In this part, the various performance metrics used in this study are presented, starting from the actual performance metrics:

- **Data packet delay [s]** is measured from data packet generation at the application layer to the reception of the packet at the target node’s application. In this way, the delay is end-to-end and covers everything the packet experiences in the network (routting, queuing, propagation…).

- **Jitter [s]** (Data packet delay standard deviation) measures the variation of the data packet delay. There are several ways to calculate jitter. One way is to calculate the delay difference of consecutive packets, which is referred to as absolute jitter. However, in this study, data packet delay standard deviation will be used. When dealing with average values, both of these give very similar information.

- **Data throughput** is the amount of data received correctly by the destination nodes (end-to-end) per time unit. This is normalized to the channel bit rate.

- **Total network control load [bit/bit]** gives a realistic picture of the total control overhead in the network. All other than the (application) data (in bits) transferred in the network, including MAC and physical layer overhead, are measured and normalized to the correctly received data (bits).

- **Routing overhead [bit/bit]** gives a realistic picture of the total routing overhead in the network. All routing overhead (bits) is measured and normalized to the correctly received data (bits).

- **Routing overhead [pk/pk]** is similar to the previous metric, but only routing control packets are calculated, and normalization is done to the data packets.
It gives an idea about the extra transmissions required by routing. In several studies, this performance metric is favored instead of the previous one (e.g., (Broch et al. 1998, Perkins, et al. 2001)). However, the author does not see here any reason for trying to find the superior of these metrics, but instead both are considered very useful, giving different views of the routing overhead.

- **Packet loss ratio** is the ratio of the lost application level data packets (i.e., 1 – packets delivered to the destination application / generated application packets). With delay, this is the most important QoS performance metric.

Also, some informative performance metrics are needed in examining the behavior of the systems:

- **Number of collisions per packet** is averaged over all correctly received physical layer packets (data, control, etc.). It should be noted that simultaneous receptions with different codes are also calculated as collisions by this performance metric, even though they are MAI rather than collisions.
- **Number of retransmissions** is the overall number of MAC-level retransmissions.
- **Number of retransmission failures** is the overall number of failed MAC-level re-transmissions (the maximum number of retransmissions was tried without successful reception of an ACK).
- **Buffer overflows** tell directly the number of packets that are thrown away due to buffer fulfillments at the network layer.

The study does not include an energy consumption metric as such, since energy efficiency issues are an art form of their own, and including this would be going too far from the main focus of this work. However, metrics like total network control load help also in comparing the energy consumption of the nodes, since, naturally, the amount of sent traffic is directly comparable to the energy consumption caused by Rx/Tx operations.

As regards performance, there are no common limits to define acceptable performance, since the requirements for delay, packet loss, and also to other QoS metrics are dependent on the used application. For example, data applications can typically tolerate high delays, but the packet loss must be practically zero, while, in contrast, real-time audio conversation can tolerate fairly high packet losses (depending on the codec, of course) but not delay. Some common recommendations by ITU are listed in Table 8 and Table 9 for multimedia and data applica-
tions, respectively (ITU-T 2001). These given QoS recommendations are end-to-end type values seen by the user (application layer), so they can also be used as a reference point in this study.

Table 8. ITU-T G.1010 QoS recommendations for audio and video applications.

<table>
<thead>
<tr>
<th>Medium</th>
<th>Application</th>
<th>Degree of symmetry</th>
<th>Typical data rates</th>
<th>Key performance parameters and target values</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>Conversational voice</td>
<td>Two-way</td>
<td>4–64 kbit/s</td>
<td>&lt; 150 ms preferred &lt; 400 ms limit</td>
<td>&lt; 1 ms &lt; 3% packet loss ratio (PLR)</td>
</tr>
<tr>
<td>Audio</td>
<td>Voice messaging</td>
<td>Primarily one-way</td>
<td>4–32 kbit/s</td>
<td>&lt; 1 s for playback &lt; 2 s for record</td>
<td>&lt; 1 ms &lt; 3% PLR</td>
</tr>
<tr>
<td>Audio</td>
<td>High quality</td>
<td>Primarily one-way</td>
<td>16–128 kbit/s</td>
<td>&lt; 10 s &lt;&lt; 1 ms</td>
<td>&lt; 1% PLR</td>
</tr>
<tr>
<td>Video</td>
<td>Videophone</td>
<td>Two-way</td>
<td>16–384 kbit/s</td>
<td>&lt; 150 ms preferred &lt; 400 ms limit</td>
<td>&lt; 1% PLR Lip-synch: &lt; 80 ms</td>
</tr>
<tr>
<td>Video</td>
<td>One-way</td>
<td>One-way</td>
<td>16–384 kbit/s</td>
<td>&lt; 10 s</td>
<td>&lt; 1% PLR</td>
</tr>
</tbody>
</table>

1Assumes adequate echo control.

2Exact values depend on specific codec, but assume the use of a packet loss concealment algorithm to minimize effect of packet loss.

3Quality is very dependent on codec type and bit-rate.

4These values are to be considered as long-term target values which may not be met by current technology.
<table>
<thead>
<tr>
<th>Medium</th>
<th>Application</th>
<th>Degree of symmetry</th>
<th>Typical amount of data</th>
<th>Key performance parameters and target values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>One-way delay (Note)</td>
</tr>
<tr>
<td>Data</td>
<td>Web-browsing – HTML</td>
<td>Primarily one-way</td>
<td>~10 KB</td>
<td>Preferred &lt; 2 s /page</td>
</tr>
<tr>
<td>Data</td>
<td>Bulk data transfer/retrieval</td>
<td>Primarily one-way</td>
<td>10 KB–10 MB</td>
<td>Preferred &lt; 15 s /page</td>
</tr>
<tr>
<td>Data</td>
<td>Transaction services – high priority, e.g., e-commerce, ATM</td>
<td>Two-way</td>
<td>&lt; 10 KB</td>
<td>Preferred &lt; 2 s</td>
</tr>
<tr>
<td>Data</td>
<td>Command/control</td>
<td>Two-way</td>
<td>~ 1 KB</td>
<td>&lt; 250 ms</td>
</tr>
<tr>
<td>Data</td>
<td>Still image</td>
<td>One-way</td>
<td>&lt; 100 KB</td>
<td>Preferred &lt; 15 s</td>
</tr>
<tr>
<td>Data</td>
<td>Interactive games</td>
<td>Two-way</td>
<td>&lt; 1 KB</td>
<td>&lt; 200 ms</td>
</tr>
<tr>
<td>Data</td>
<td>Telnet</td>
<td>Two-way</td>
<td>&lt; 1 KB</td>
<td>&lt; 200 ms</td>
</tr>
<tr>
<td>Data</td>
<td>E-mail (server access)</td>
<td>Primarily one-way</td>
<td>&lt; 10 KB</td>
<td>Preferred &lt; 2 s</td>
</tr>
<tr>
<td>Data</td>
<td>E-mail (server to server transfer)</td>
<td>Primarily one-way</td>
<td>&lt; 10 KB</td>
<td>Can be several minutes</td>
</tr>
<tr>
<td>Data</td>
<td>Fax (&quot;real-time&quot;)</td>
<td>Primarily one-way</td>
<td>~ 10 KB</td>
<td>&lt; 30 s /page</td>
</tr>
<tr>
<td>Data</td>
<td>Fax (store &amp; forward)</td>
<td>Primarily one-way</td>
<td>~ 10 KB</td>
<td>Can be several minutes</td>
</tr>
<tr>
<td>Data</td>
<td>Low priority transactions</td>
<td>Primarily one-way</td>
<td>&lt; 10 KB</td>
<td>&lt; 30 s</td>
</tr>
<tr>
<td>Data</td>
<td>Usenet</td>
<td>Primarily one-way</td>
<td>Can be 1 MB or more</td>
<td>Can be several minutes</td>
</tr>
</tbody>
</table>

Note: In some cases, it may be more appropriate to consider these values as response times.

### 4.3 Operation in a conventional access point network scenario

As a first simulation study, a conventional network scenario with an access point is taken. This scenario has one AP located at the center of a circle of ten stationary nodes communicating to the AP (Fig. 26). The circle has a radius of about 210 m. Hence, there will be serious hidden terminal problems, as the nodes at the opposite sides do not hear each other ($d_r ≈ 250$ m). Nodes communicate only to the fixed network via the access point and not to each other. Also, the communication is one-way in the sense that there is no data traffic towards the wireless nodes (only ACKs). Fixed routing is used in this scenario. The next table summarizes some special parameters of this scenario.
Let us first study a special case with a chip rate of only 4 Mchip/s. This implies that the processing gain is only 4 (or ~ 6 dB). By using equations (20), (21), and (22), we can calculate that two simultaneous transmissions will lead to an SINR of about 2.6 dB at the access point, which is not enough for error free transmission. Hence, a collision is likely to destroy every colliding transmission.

Fig. 27 presents the average delay as a function of $G$. The delay figure is illustrative, since it is easy to see when the network functions well (low, steady delay) and when it starts to get congested (fast rise in the delay with an increasing traffic load). The extra delay is caused by queuing, and when the rise settles, it is an indication of a congested network: The buffers are full and packets will be lost (see Fig. 28 for packet loss). The point where the network starts to get congested is referred to as the point of congestion in this work.
As seen, 802.11 RTS/CTS performs very well, which is not a surprise, since the four-way handshaking was particularly made for this purpose: avoiding the hidden terminal problem. The cost of RTS/CTS handshaking is seen as a minor delay increase under steady conditions. 802.11 without RTS/CTS also works fine. However, BC-MAC is not at its best in this kind of scenario. See, e.g., that 802.11 RTS/CTS is able to handle traffic loads of about 0.6 before the delay explodes, while BC-MAC is only able to handle about a tenth of that. This is simply because no major effort is put into solving the hidden terminal problem. Also, there is no additional benefit of using BCCA, since the data is transmitted only to the access point with the access point’s own R-code, and, hence, transmissions with R-code and C-code are equally vulnerable to collisions.

In the packet loss figure (Fig. 28), BC-MAC is shown with and without CDo, and the benefit of using CDo is seen immediately. It considerably boosts the performance of BC-MAC under a low traffic load, since in this scenario collisions are the main cause of packet loss. In fact, with BC-MAC CDo, one can get the best packet loss performance of the setups under comparison under a low traffic load. However, CDo can only handle simple collisions, but the performance is rapidly decreased if there are several hidden terminals competing for access. This is seen as a quick rise in packet loss. From the delay figure (Fig. 27), we can see that CDo does not help network congestion: The point of congestion is practically the same with and without CDo. In 802.11, the exponential back-off algorithm...
can tackle multiple collisions under a heavy traffic load, leading to better performance than BC-MAC.

![Graph](image)

**Fig. 28. Packet loss ratio (access point scenario, \( G_p = 4 \)).**

Next, the chip rate is set to 11 Mchip/s (similar to the IEEE 802.11). The situation is totally different as seen in the delay behavior in Fig. 29. The differences in performance between the setups have clearly narrowed, and, in fact, BC-MAC now functions better than 802.11 and almost reaches the performance of 802.11 RTS/CTS. Moreover, if we take a look at the packet loss ratio (Fig. 30), we see that mostly BC-MAC has the lowest loss ratio.

The reason for this behavior change relates to the capture effect. In the case of two simultaneous transmissions, the calculated SINR is 4.1 dB, which implies that a single collision is not enough for destroying the ongoing transmission. Hence, BC-MAC functions efficiently, since there is no need to consume resources for collision avoidance. Naturally, at a very high traffic load, collisions of several packets can corrupt the ongoing transmission, but this situation is rare and will dominate only under a high traffic load. This is seen in Fig. 30, where the packet loss of BC-MAC rises rapidly and exceeds that of 802.11 RTS/CTS at \( G \approx 0.5 \). However, at this point the packet loss ratio is already too high (> 2%) for most of the applications, and thus, it does not matter much what setup performs better beyond this point. In many studies, it is pessimistically assumed that a collision always leads to the failure of all of the colliding transmissions. In spread spectrum systems (like IEEE 802.11), this is not the case.
Fig. 29. Data packet delay (access point scenario, $G_p = 11$).

With wider bandwidth, collisions are not as dominating cause for packet loss as with the narrower band in the earlier case. Hence, CDo in this case does not provide a great improvement on the packet loss behavior, as seen in Fig. 30. The improvement gained with CDo is almost marginal with higher bandwidth, but it did not bring any negative effects either.
As a conclusion, it can be stated that BC-MAC is not at its best in this kind of AP scenario, but it does already show some favorable properties. Interestingly, 802.11 based setups seem to give practically the same performance regardless of the spreading factor. Hence, they work fine also in narrow band cases, unlike BC-MAC, but, in contrast, they fail to harness the potential of spread spectrum techniques.

4.4 Operation in a special 16-node random grid scenario

From an AP network, we go towards a more distributed network with no central nodes. A 16-node random grid topology, shown in Fig. 31, is studied. The topology has random components but is still somewhat controlled, because the random placements of the nodes lie on a grid. The network size is 400 m × 400 m, and each node has its own 100 m × 100 m square, where they are placed randomly. The traffic scenarios are now following the default model of DC. No mobility is yet included, so this kind of scenario presents an intermediate form of a centralized topology and a real ad hoc topology.

![Fig. 31. A 16-node random grid scenario.](image-url)
4.4.1 A single-hop scenario

First, we will test a single-hop scenario, where, obviously, no routing protocol is needed. The transmission power is set to 35 μW, so the effective transmission range will be about 605 m, which is enough for covering the whole network area (the longest possible link is \(\sqrt{400^2 + 400^2} = 566\) m). Table 11 summarizes the special simulation parameters used.

Table 11. Simulation parameters of a single-hop 16-node random grid scenario.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC (w/o CDo), 802.11 RTS/CTS, 802.11</td>
</tr>
<tr>
<td>BC-MAC channel sensing</td>
<td>PD, SD, PSD</td>
</tr>
<tr>
<td>Area</td>
<td>400 m × 400 m (random grid)</td>
</tr>
<tr>
<td>Transmission power</td>
<td>35 μW</td>
</tr>
<tr>
<td>Routing protocol</td>
<td>no routing needed</td>
</tr>
<tr>
<td>Mobility model(s)</td>
<td>no mobility</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>16 active</td>
</tr>
</tbody>
</table>

Fig. 32 presents the simulated throughput. The first notation is immediately that, since there is no longer centralized fashion in the network, BC-MAC already gives the best performance of the studied setups. The throughputs of 802.11 and BC-MAC are almost equal as could be expected on the basis of the theoretical results shown earlier in, Fig. 23, since this scenario is much similar to the assumptions behind the theoretical results. The reached maximum throughputs are quite close to the theoretical ones, as, e.g., 802.11 seems to reach a throughput of about 0.8, while the theoretical value is about 0.83. BC-MAC is very close to the theoretical performance (0.83 vs. 0.84), but 802.11 RTS/CTS performs clearly worse than the assumed theoretical maximum (0.73). The differences are expected, as the theoretical calculations assumed ideal conditions. In the simulations, several non-idealities are already present, and, e.g., carrier sensing is not perfect.

Packet losses and retransmissions are the main reasons why the results differ from the theoretical ones, since error-free transmission was assumed in the theoretical calculation. From the reTx behavior shown in Fig. 33, it can be concluded that the number of occurred retransmissions correlate well to the differences between the theoretical and simulated throughputs. The more retransmissions there are, the bigger the difference is. Since there are no hidden terminals, the exchange
of RTS and CTS packets does not make much sense, and, thus, 802.11 RTS/CTS is not at its best in this kind of scenario.

Fig. 32. Throughput in a 16-node random grid network (single-hop).

Because of CSMA, BC-MAC performs similarly to 802.11, and no additional benefits are gained from BCCA in addition to those cases where the channel sensing fails. However, if we take another look at the retransmissions (Fig. 33), we

Fig. 33. Number of retransmissions in a 16-node random grid network (single-hop).
see that the number of retransmissions with BC-MAC is only a fraction of those of 802.11. The collision avoidance in 802.11 is more efficient than in BC-MAC. Hence, there will be more collisions with BC-MAC (not shown), but because of BCCA, packets are lost practically only due to relatively rare Tx/Tx collisions. This implies that there is unused potential in BC-MAC. BCCA enables channelization and allows multiple simultaneous successful transmissions, but CSMA prevents it.

There is a way to encourage more simultaneous transmissions, but again the problem is that there are no means for separating “the good collisions” from “the bad collisions”. By now, CSMA with power detection (PD) is used as is typically done in BC-MAC, but we can try using pure signal detection (SD), where the channel is considered busy only when a valid signal is detected regardless of the power level. Using SD with BCCA forms a very loose CSMA method in this scenario, since only the transmission meant to the node itself will be detected as there are no C-code transmissions. Hence, there is no way of knowing whether the intended destination node is transmitting or not. Another special setup is to use both signal and power detection (PSD), but with a higher power sensing threshold (shorter range). By default, the threshold has been the predicted effective transmission range, but now we test a threshold of $1.5 \cdot 10^{-13}$ W, which gives a sensing range of only about 150 m. Also, PD with a reduced detection range will be tested.

![Fig. 34. Packet loss ratio in a 16-node random grid network (single-hop) with various channel sensing methods.](image-url)
Packet loss ratios with various CSMA detection techniques are shown in Fig. 34. At first, if we consider only the BC-MAC results, we see that reducing CSMA to the level of SD does not work well and gives mainly the worst performance. Only BC-MAC with PD of a 150 m CSMA range gives a worse performance than SD. However, if we look more carefully, this is the situation merely under a low traffic load, but with an increasing offered load, BC-MAC with a PD of 150 m performs eventually even better than 802.11 and the default BC-MAC. This is seen also in the throughput curves shown in Fig. 35, where BC-MAC with a PD of 150 m reaches very high throughputs, and, in fact, higher than the theoretical maximum of the random channel access! Exceeding theoretical limits is not extraordinary, because the maximum throughput of 1 is a limit for random access schemes. BCCA is not a pure random access scheme, and when used in this way, its capabilities of allowing several simultaneous successful transmissions in a small area begin to function. Even higher throughputs are achieved with PSD. With this setup, also the packet loss ratio is practically the best in all regions. If we take, for instance, the packet loss ratio of $10^{-3}$ as a limit of acceptable behavior, we see that BC-MAC PSD is able to handle a data traffic load of about 1.1 before exceeding the limit, while 802.11 can handle a load of ~ 0.8 and BC-MAC SD only 0.5.

---

**Fig. 35. Throughput in a 16-node random grid network (single-hop) with various channel sensing methods.**

The figures also include 802.11 with a PD of 150 m. The same effect as with BC-MAC is not seen, but, instead, the performance is constantly worse than in all the
other studied setups. This could be expected on the basis of retransmission behavior (Fig. 33), which did not show as much potential for 802.11 as it did for BC-MAC. The reason is obvious: As 802.11 is a common channel pure random access scheme, its performance will be corrupted by collisions, while, in contrast, the performance of BCCA can be enhanced by allowing collisions reasonably. This scenario highlighted the potential of the BCCA method, but also the difficulty of getting the most out of distributed random CDMA access. BCCA definitely needs CSMA, but SD is a too loose method. CSMA with reduced PD sensing capability works fine under a high traffic load but is not reliable enough under a low traffic load. PSD with a reduced sensing range, on the other hand, gave very good results at least in this well-controlled single-hop study. Despite the apparent potential of PSD, in reality, parameterization will probably become a problem, since the performance is obviously heavily dependent on the sensing range.

4.4.2 A multi-hop scenario

Next, a multi-hop version of the same scenario is tested, where the effective radio range is dropped to the default ~ 250 m. Hence, routing is needed, and AODV is used. The situation is already very close to a pure random ad hoc network case.

The behavior is very different compared to single-hop cases. Consider, e.g., the behavior of delay variation in multi-hop vs. single-hop, which is shown in Fig. 36: Whereas the jitter decreases steadily as a function of decreasing $G$ in single-hop cases, it saturates to an almost constant value in multi-hop cases. The figure also includes BC-MAC jitter behavior in the AP scenario that was studied earlier, and shows that hidden terminals do not cause the saturated jitter behavior. Rather, the saturated jitter is caused by the very nature of ad hoc networking: A multi-hop environment introduces unreliable links even in the stationary network case, since with nonzero probability there will be long unreliable links due to the random placement of nodes. Unreliable links cause route breakages, which leads to route re-establishments, causing variation in the delay. Nevertheless, the level of jitter is low (< 10 ms) and can be easily compensated by buffering (e.g., with a leaky bucket algorithm, (Turner 1986), but the behavior is interesting to note. If larger ad hoc networks with longer routes are considered, the jitter level could easily become notable.
The performance between 802.11 and BC-MAC was almost the same in the single-hop case, but in the multi-hop case BC-MAC starts to reclaim its expectations: BC-MAC clearly outperforms 802.11, as seen in the packet loss behavior in Fig. 37. Table 12 shows numerical information about the supported maximum offered traffic loads with given packet loss criteria, and also the gain of BC-MAC over 802.11 is presented. As seen, the small differences in single-hop case are turned into clear distinctions in the multi-hop case. The differences become narrower towards higher packet loss criteria, but even with loss ratio of 0.1, the improvement provided by BC-MAC is over 280%.

Table 12. Maximum offered traffic loads and performance differences between BC-MAC vs. 802.11 with a given packet loss criteria in multi-hop and single-hop scenarios in a 16-node random grid network.

<table>
<thead>
<tr>
<th>Packet loss criteria</th>
<th>Maximum offered traffic load with 802.11, multi-hop, (single-hop)</th>
<th>Maximum offered traffic load with BC-MAC, multi-hop, (single-hop)</th>
<th>Performance improvement of BC-MAC over 802.11 in %, multi-hop (single-hop)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>0.210 (0.935)</td>
<td>0.800 (0.955)</td>
<td>281% (2%)</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>0.083 (0.851)</td>
<td>0.635 (0.877)</td>
<td>665% (3%)</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>0.037 (0.780)</td>
<td>0.515 (0.834)</td>
<td>1292% (7%)</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>0.011 (0.685)</td>
<td>0.432 (0.751)</td>
<td>3827% (10%)</td>
</tr>
</tbody>
</table>

By examining the packet loss behavior (Fig. 37) of 802.11 RTS/CTS, it is seen that it no longer functions well in a multi-hop case. As was predicted in the dis-
discussion earlier, the advantages of virtual handshaking are turned into disadvantages in a distributed multi-hop environment.

Fig. 37 also shows the packet loss when using PSD, which, a bit surprisingly, gives about the same performance as PD. In PSD, a reduced sensing range of about 150 m was again used. This sensing range is now obtained with a threshold of $2.571 \times 10^{-14}$ W (due to different transmission power as compared to the single-hop case). In the earlier single-hop scenario, CSMA with full range PD gave fairly good results, but it caused BC-MAC to act like a pure random access scheme, and, thus, PSD with a reduced detection range gave clearly better results. However, in this multi-hop scenario, the situation is different because the CSMA range does not cover the whole network area, and, hence, the improvement gained with PSD is not as convincing as it was in the single-hop scenario. PSD probably has potential, but, as assumed before, it is difficult to parameterize. It functions as intended, encouraging simultaneous transmissions, which can be concluded from the collision behavior (not shown), but the actual performance improvement over PD is marginal.

In the packet loss figure, also BC-MAC with SD is shown, and as it could be assumed on the basis of the single-hop scenario, it does not work well in multi-hop scenario either. As a result, only BC-MAC with PD will be used through the rest of the paper, and the optimization of PSD will be left for future work.
4.5 Operation in a typical 20-node ad hoc network scenario

Next, we turn our attention to a real multi-hop ad hoc network, where 20 active nodes are placed randomly to an area of 500 m × 500 m, which gives a node density of 80 nodes/km². The hop count varies from 1 to 3, and in some special topologies there can be even more, but on average, there are about 1.7 hops (studied with the connectivity analysis tool). In many ways, this 20-node network represents a typical situation of ad hoc networking that could be imagined for a flat structure in real life, and, hence, it will be the default ad hoc network scenario used in several tests of this thesis.

Table 13 presents the simulation parameters in use. In this scenario, in addition to typical setups there is a special setup where 802.11 is enhanced with a BCCA functionality. Also, the performance of NSCD over BC-MAC will be examined.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, BC-MAC NSCD, 802.11, 802.11 RTS/CTS, 802.11 BCCA</td>
</tr>
<tr>
<td>Area</td>
<td>500 m × 500 m</td>
</tr>
<tr>
<td>Mobility models</td>
<td>no mobility, RWM (MLV)</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20 active</td>
</tr>
</tbody>
</table>

4.5.1 A stationary scenario

First, a stationary scenario is evaluated. Fig. 38 shows the packet loss ratios. Now, since we are dealing with a real ad hoc situation, the different setups have very different performances. If, for example, a 0.1% packet loss is considered an acceptable loss for some application, 802.11 can only handle a traffic load of about 0.02, while BC-MAC can handle a traffic load of about 0.45. Thus, a BC-MAC network can handle over 20 times more traffic than a 802.11 based network. Enhancing 802.11 with a BCCA functionality improves the performance of 802.11 significantly as it allows a traffic load of about 0.23 with a 0.1% packet loss, but BC-MAC still outperforms it. 802.11 BCCA is able to reach very low packet loss ratios of the order of 10⁻⁵ under a traffic load of less than 0.05, but the packet loss with BC-MAC is already practically zero at that point. Basic 802.11 reaches the level of 10⁻⁵ only with a very low traffic load of the order of 0.001.
Fig. 38. Packet loss ratio (20 stationary nodes).

Using NSCD instead of the assumption that the spreading codes are known a priori turns out to give nearly the same performance. In fact, no conclusions can be drawn from the packet loss that NSCD would distinctly worsen the performance.

Similarly as in the multi-hop random grid scenario, it is noted that while 802.11 RTS/CTS was effective in the access point scenario, it is not at its best in a real ad hoc network. In fact, it never reaches packet losses below ~ 0.1% in this scenario. The bad performance was expected on the basis of earlier discussion on the four-way handshaking system. RTS/CTS is meant mainly for centralized networks, where the hidden terminal problem is a major problem. In the case of an ad hoc network, there are different kinds of problems, which favor more straightforward channel access methods. An efficient CA method prevents simultaneous transmissions in a large area, and, thus, through the decrease of spatial capacity, it radically decreases the total transmission capacity of the network. Moreover, delay caused by route establishments and multi-hop routing is already a problem in ad hoc networks, and this is further enhanced by a slow channel access method.

As was stated in the connectivity discussion in section 2.4, the non-connected network case iterations are removed from the analysis of the stationary networks. However, Fig. 38 shows yet one more interesting graph, which represents BC-MAC with known codes but with the connectivity problem included. As stated, non-connected cases are very rare in this scenario, and, thus, reliable connectivity results are difficult to simulate. In fact, to produce this graph, 1,000 iterations were run per traffic point instead of the default 32, and the simulation took sev-
eral weeks of computer time. As seen, connectivity has a serious effect: As assumed, it sets the lower bound for achievable packet loss. The packet loss ratio lower bound is about 0.0016, while the semi-theoretically calculated value was 0.00156, being very close to the simulated one, thus giving credibility to the performed connectivity analysis. It is seen that the connectivity only sets the lower bound, but the graph otherwise follows the fully connected BC-MAC graph. Similar results were also obtained for 802.11 (not shown), as could be expected, since this is a scenario limitation, not a MAC or a channel access limitation. Thus, BC-MAC shifts the point of congestion and clearly enhances the performance over 802.11, but it cannot tackle connectivity problems.

An adaptive antenna approach has already been taken to extend BC-MAC for easing also the connectivity problem, and very promising results have been got (Bräysy et al. 2006). Nevertheless, since this study is not about connectivity, only fully connected networks are studied in the rest of the stationary scenarios. Despite this decision, all the results presented apply to fully connected cases, thus giving a good view of the typical performance. It should be noted that even though the average packet loss in the connectivity problem graph looks difficult, the few non-connected network cases with very high packet loss dominate the performance, while the rest perform as in the fully connected BC-MAC graph.

Fig. 39 shows the packet loss ratio behavior of BC-MAC and 802.11 with detailed information, including standard deviation bars and minimum & maximum values. As seen (and could be expected on the basis of the packet loss calculation principle), the proportional variation is increased towards lower packet loss ratios. Under a low traffic load, for example, the mean packet loss and standard deviation of BC-MAC stay approximately below the level of $10^{-5}$. However, there are still some scenario(s) where the maximum values lie near $10^{-4}$. This emphasizes the importance of sufficient averaging, as discussed earlier. An ad hoc network is a highly dynamic system and, hence, has great variation depending on the topology. This figure was an example of a more accurate statistical view, but as this form of presentation is unclear in the case of several plots in the same figure, only the average values are shown in the rest of the figures. However, when planning real ad hoc networks, more exact statistical information would be beneficial, since it is important to know, e.g., confidence intervals for the packet loss.
Interestingly, the number of reTx failures shown in Fig. 40 is continuously a magnitude higher with RTS/CTS than with other setups. ReTx failures lead directly to link breakages and, thus, route breakages (shown only for BC-MAC and 802.11 in Fig. 54), since the detection of link breakages is handled by MAC ARQ. Route breakages cause packet loss and lead to increased delay due to route re-establishments. In fact, with RTS/CTS, there is an increased probability of link breakages, since, in addition to the failure of a data packet, the failure of an RTS packet will also lead to a link breakage. In other words, RTS/CTS CA in an ad hoc environment turns out to act as a link status double check. Because of this, it is more likely that a link will be falsely detected to be broken if there are several nodes competing for channel access. Also, naturally unstable links (e.g., long links, crowded links due to weak connectivity) will be easily (and continuously) detected to be broken. This is of course a positive feature if there are other routes to be used to reach the destination. However, if an unstable link is the only link to reach the destination, continuous route breakages and route re-establishments only ruin the performance and cause extra control traffic.

The benefit of using CDMA is clearly seen as, on average, BC-MAC fails only some tens of times (Fig. 40). Even if the network is congested, BC-MAC fails only about 80 re-transmissions. The difference to 802.11 is huge, as it fails several tens of thousands of times at worst. Notice also the difference in behavior:
While the number of failures with BC-MAC is reasonably controlled, the failures with 802.11 grow exponentially with an increasing traffic load until the network gets to a congested state.

![Graph](image-url)

**Fig. 40. Number of re-transmission failures at MAC (20 stationary nodes). Figure reprinted from (Prokkola & Bräysy 2007), © 2008 Elsevier.**

In addition to link failures, complex CA methods tend to cause extra overhead. This can be clearly seen in Fig. 41, which shows the total network control load. The control load is significantly high if RTS/CTS is used, and with increasing traffic load, the network suffocates rapidly to the control overhead. The increase of control traffic with BC-MAC is moderate even in the congested network. 802.11 BCCA produces more control traffic than BC-MAC, but this amount is still much less than with 802.11 (not to mention 802.11 RTS/CTS). The produced extra control load is caused by extra transmissions, which lead, e.g., to additional power consumption. The rapid raise of the control load is mainly a consequence of the high number of route breakages.

The use of NSCD affects directly the control load, since the control packets of AODV are longer when NSCD is used. Hence, the control overhead is slightly higher when NSCD is used, but the difference to the known code setup is negligible.

Let us turn our attention back to the packet loss and examine where the packets are actually (finally) lost. The network and link layers have a shared interactive flow control, i.e., the network layer does not push packets to a lower layer
until it receives an indication that the previous packet was transmitted successfully. Therefore, if the first link of a route fails, AODV still has a copy of the packet and can try finding a new route and still get the packet through. However, if the route fails at the second hop or further and the AODV route repair is not successful, the pending packets are discarded and lost in the middle of the route, causing packet loss from the higher layer perspective. On the other hand, if the lower layers are not capable of transmitting the data as fast as required, packets will flow over from the buffers at the network layer. Hence, packets are always finally lost at the network layer, but the reason for destroying packets can be due to route breakage, unsuccessful route discovery, or buffer overflow. To overcome network layer (or below) losses, end-to-end ARQ mechanisms above the network layer are needed. TCP is the typical choice, but, as known, it is not at its best in wireless ad hoc networks (Holland & Vaidya 1999).

Fig. 41. Total network control load (20 stationary nodes). Figure reprinted from (Prokkola & Bräysy 2007), © 2008 Elsevier.

Fig. 42 shows the number of all the discarded data packets at the network layer. As seen, this correlates nicely with the packet loss behavior. Under a low traffic load, closer examination (not shown) reveals that the explanation for packet loss lies in unstable links in the routes. 802.11 based systems are more vulnerable to unstable links, and the number of route breakages increases rapidly as more traffic is pushed to the network. The process starts to feed itself: Route breakages
produce RERR packets and cause route re-establishments, increasing the total traffic load. This increases MAI and the probability of collisions, weakening the reliability of the links. In real networks with real users, the situation is even worse, since when the users notice the decreased QoS level, they often start discovering the reason for this and initiate e.g., ping requests, thus generating even more traffic to the network (Ramachandran, et al. 2004).

![Graph](image)

Fig. 42. Destroyed data packets at the network layer (20 stationary nodes). Figure reprinted from (Prokkola & Bräysy 2007), © 2008 Elsevier.

Coming back to route breakages, the number of them increases until the network gets congested. Beyond network congestion, the breakage count (see the retransmission failure behavior in Fig. 40) starts to decrease, since with increasing $G$, more and more packets will be dropped already from the network layer buffer, and they never get to MAC. Buffer overflows, shown in Fig. 43, become one of the major causes of packet loss under a high traffic load. However, by examining the exact values of the buffer overflows and number of retransmission failures, an interesting observation is made: Buffer overflows quickly become the dominating reason of packet loss for BC-MAC, but for 802.11 based systems, packet losses due to a high number of route breakages remain the dominating cause of packet loss also under a high load. Naturally, at some point ($G >> 10$), buffer overflows would start to dominate the loss also for 802.11 based systems, but this does not happen yet with the studied values of $G$. BC-MAC is efficient in keeping the routes functional, and hence the packets will be just dropped because the lower
layers cannot serve higher layers as quickly as needed under congestion. In 802.11, however, the routes are lost under congestion, and the majority of the traffic after this is route establishment control traffic.

![Buffer overflows at the network layer (20 stationary nodes).](image)

Packet loss behavior did not show practically any difference between NSCD and pure BC-MAC. However, there is some difference in the maximum achievable throughput shown in Fig. 44. Nevertheless, the extra loss caused by NSCD is only a fraction of the gain that is attained with BC-MAC. For example, 802.11 reaches a throughput of about 0.2, while BC-MAC reaches significantly higher, beyond 0.7 and almost up to 0.8. Enabling BCCA boosts 802.11 clearly, giving maximum throughput of the order of 0.5.

When inspecting the throughput behavior, one must recall that when the throughput curve branches from the ideal random channel access curve, it is an indication of network congestion. Beyond this, minor differences do not matter in terms of comparison, since the packet loss is already far too high for any practical applications. Still, the behavior of maximum throughput under very high traffic load is not irrelevant, because it is an indicator of the stability of the system. Modern data traffic is highly bursty in nature, so it is important that networks recover also from temporary high traffic loads and that the throughput of the network does not collapse. Clearly, all the studied setups are very stable as they do
not cause the network to collapse even though steady traffic loads of 10 times the channel bit rate are pushed to the network.

The behavior of the delay jitter is shown in Fig. 45. High jitter is an indication of high delay variation and high temporary delays, which can be caused by buffer fulfillments due to problems at lower layers, leading finally to packet loss. BC-MAC gives the lowest jitter, but as seen in the figure, the jitter with all the setups (except with 802.11 RTS/CTS due to time-consuming handshaking) is about at the same level under a low traffic load. This level is about 6–8 ms, which is very good for a wireless and, in particular, for a multi-hop ad hoc network. For example, the jitter limit of 1 ms for multimedia applications (ITU recommendations, (ITU-T 2001)) is easily achievable with buffering, since the end-to-end delay (Fig. 55) is also small (~ 10 ms), making no restrictions to multimedia application usage. The behavior of BCCA-enabled setups is worth attention: They are able to maintain low jitter steadily up to the point of congestion. This behavior is advantageous considering the application QoS, since it is important to maintain steady performance even under a highly variable network traffic load.

Next, we can compare different ways of calculating control overhead. By comparing the total network control load (Fig. 41) and the routing control load shown in Fig. 46, it can be concluded that a great deal of the absolute control load is caused by layers below the network layer. This overhead is formed naturally by MAC
headers, PLCP, MAC ACKs and MAC retransmissions. However, lower layers still cannot be blamed for causing the majority of the control overhead, because their operation is tied to the higher layer operation. Consider, e.g., if the routing protocol lost a route and initiated a new search: It would cause several transmissions at the lower layers increasing the total overhead. This can be also seen on the basis of the similar shapes of the curves in all control load figures.

Fig. 45. Data delay standard deviation (20 stationary nodes). Figure reprinted from (Prok-kola & Bräysy 2007), © 2008 Elsevier.

The differences in routing overhead when measured in bits (Fig. 46) or packets (Fig. 47) are also interesting. As seen, BC-MAC needs clearly the least routing packets, because BC-MAC also has the smallest number of lost routes. However, the differences in control packets are quite meaningless to the total overhead, since when the routing overhead is measured in bits, the differences between different setups under a low traffic load are barely visible. Fig. 46 and Fig. 47 also verify that the extra overhead of NSCD is caused only by the longer AODV control packets, while the number of control packets remains unchanged.
As a conclusion, this real ad hoc scenario now showed the true benefits of using BCCA and BC-MAC: BC-MAC totally outperforms 802.11. 802.11 with BCCA also gives high performance when compared to pure 802.11, while the RTS/CTS mechanism is not at its best in this kind of scenario. In addition, NSCD was shown to function very well, and the control load increase was almost insignificant.
As was discovered, the RTS/CTS mechanism realizes a link status double check in ad hoc networks, and thus it has a higher probability of falsely detecting links to be broken. This results from the fact that in the RTS/CTS mechanism, one has to manage two successful transmission pairs (RTS/CTS and data/ACK) instead of directly transmitting data, and a route breakage is caused in both cases: in the failure of RTS or the following data packet. Some might say that this is in fact a positive feature, since then the routing protocol is able to find better routes with perhaps more reliable links. In reality, it not that straightforward because of the connectivity: Sometimes an unreliable link might be the only link for providing full connectivity, and, thus, continuous route searches would only decrease the performance. Naturally, in a mobile case the situation is different, because the link conditions might be only temporary. It can be further assumed that in a dense network, quick detection of unreliable links is favorable, while in a sparse network, it is preferable to hold even the unreliable links as long as possible.

Typically, it is assumed that since the RTS/CTS mechanism is more efficient in reducing collisions, it can operate with fewer reTx tries. Therefore, a reTx limit of 4 is typically used instead of 7 (which is used without RTS/CTS). But, because of the link status double check phenomenon, this assumption turns out to further increase the probability of incorrect link breakage detection. Also, it must be noted that collisions might not be the dominating cause of packet loss in ad hoc networking, questioning the reasonability of the use of RTS/CTS. Consequently, in this test, we will test what happens when both the long and short retry limits are set to 20 (cf. BC-MAC).

Fig. 48 presents the packet loss ratio behavior of 802.11 and 802.11 BCCA with a 20 reTx try limit and default limits. For 802.11 RTS/CTS, also a special version with BCCA is shown (both with a 20 reTx try limit). The reTx try limit has a major effect on the performance in an ad hoc network environment, unlike is often assumed in the case of traditional networks. Increasing the reTx limit to 20 clearly improves the packet loss performance with all setups. The most obvious observation is in the performance of 802.11 RTS/CTS, which now performs even better than 802.11 when the network is getting congested. Under a low traffic load, pure 802.11 still achieves a lower packet loss than 802.11 RTS/CTS, but the difference between the default case (shown in Fig. 38) and the 20 reTx try limit case is huge. Nevertheless, 802.11 BCCA gives the best performance, and, in fact, with 20 reTx tries, the performance becomes very good. However, BC-
MAC still performs better, and, e.g., with packet loss limit of $10^{-3}$, 802.11 can handle loads of about 0.35, while BC-MAC can handle loads of ~ 0.47 (see Fig. 38 for BC-MAC performance). It is interesting to see that the performance of RTS/CTS is not affected significantly by enabling BCCA, and 802.11 BCCA performs clearly better even without extra retransmissions. The reason for this is that RTS/CTS is a too strict method for BCCA: Possible simultaneous transmissions are prevented by the effective collision avoidance mechanism.

![Fig. 48. Packet loss ratio and the effect of an increased reTx try limit on 802.11 based set-ups (20-node stationary scenario).](image)

Even though the RTS/CTS performance has greatly improved, the link status double check phenomenon is still present. This can be observed from Fig. 49, which shows the number of route breakages. Even though the higher reTx try limit radically decreases the amount of route breakages of 802.11 RTS/CTS, breakages still remain at a higher level when compared to pure 802.11. In fact, the number of route breakages of 802.11 RTS/CTS under a low traffic load is about two times higher than without RTS/CTS. Generally, it can be stated that the reTx try limit is closely related to the capability of detecting and maintaining unreliable links.

As an overall observation, it can be stated that for 802.11 (with and without BCCA), increasing the reTx limit seems to bend the packet loss curve towards lower losses under a low traffic load, while the high traffic part remains mostly unchanged. However, for RTS/CTS, the increment of the reTx limit has a differ-
ent effect: It looks as if the whole RTS/CTS curve had been dropped towards lower packet losses. The effect of enabling BCCA for 802.11 is that it approximately shifts the whole curve towards higher traffic loads, moving the point of congestion, thus enhancing the traffic handling capability. Under RTS/CTS, this behavior is also visible, but it has a much smaller effect than without RTS/CTS.

**Fig. 49. Number of route breakages and the effect of an increased reTx try limit on 802.11 based setups (20-node stationary scenario).**

While the packet loss behavior showed great improvement when increasing the reTx try limit, there are some negative effects in the delay behavior under a high traffic load (not shown). This highlights the problem of setting the correct MAC reTx try limit, which is far from trivial; it is dependent at least on the MAC mechanism, channel access principle, physical layer characteristics, routing protocol operation, network topology, and node density. Also, the next study case shows that mobility and the reTx try limit together form a combination that affects the network performance. In addition, the situation can be completely different as regards the fading radio channel: In some cases, increasing the number of re-transmissions can be a crucial factor in decreasing the performance (Mullen & Huang 2005). A detailed analysis and optimization of the reTx try limit is out of the scope of this work, but the main effects and dependencies of it are important to understand.
4.5.2 A mobile scenario

First, in the mobile scenario test case, we examine the throughput shown in Fig. 50. As seen, bringing mobility does not affect the achieved maximum throughputs significantly. The only notable difference as compared to the stationary scenario (Fig. 44) is that 802.11 seems to perform somewhat better now. At first, this seems a bit peculiar, but the explanation is quite simple: On average, 802.11 with RWM enjoys shorter routes because of the properties of RWM (see the theory behind this in (Chu & Nikolaidis 2004)). This is proven with Fig. 51, where the average route length (hops) in both mobile and stationary scenarios is plotted for BC-MAC and 802.11.

It is interesting to note that BC-MAC does not seem to follow the theory (Chu & Nikolaidis 2004), at least under a low traffic load, as the routes seem to be even longer in the mobile than in the stationary scenario. As shown earlier, BC-MAC is relatively efficient in keeping the routes functional. Thus, BC-MAC is able to hold the functional links near to the limits of the radio range when the nodes move. 802.11, on the other hand, “gives up” earlier, forcing AODV to find new routes and keeping the routes shorter on average. The drawback is, naturally, that giving up earlier leads to an increased packet loss, as seen in Fig. 52.

![Fig. 50. Data throughput (20-node mobile scenario). Figure reprinted from (Prokkola & Bräysy 2007), © 2008 Elsevier.](image-url)
From the throughput Fig. 50, we can see that NSCD functions well also with mobile nodes, and the difference to the known code case is, again, insignificant.

In the packet loss behavior shown in Fig. 52, we can observe that the shapes of all the curves of different setups under a low traffic load are very similar. Since the
non-connected cases are not excluded in this mobile scenario, one could expect a
steady lower bound for packet loss. Interestingly, there is no steady lower bound,
and the packet loss does not decrease continuously with a decreasing traffic load,
but, rather, the loss has its minimum values just before the point of congestion,
and decreasing the traffic load from this only increases the packet loss. This be-
behavior originates now from mobility rather than from the differences between
the setups. Mobility is similar for all setups and the dominating cause of packet loss
under a low traffic load. Dealing with packet loss caused by mobility is more up
to the routing protocol and upper layers than MAC, while the main benefits of
lower layer design are in the ability to handle high traffic loads (as seen in the fact
that BC-MAC can again handle several times the traffic of 802.11).

The interesting behavior of packet loss can be explained with the relative
mobility, that is, the mobility in relation to the average interarrival time of the data
packets. Let us define the relative mobility as

$$\Omega = \frac{t_{\text{iat}}}{v}$$

where $t_{\text{iat}}$ is the (average) interarrival time between sequential generated data
packets, and $v$ is the (average) speed of nodes. Hence, $\Omega$ value 1 means that the
nodes will move, on average, the distance of the radio range in $t_{\text{iat}}$.

Fig. 53 shows the relative mobility as a function of the offered traffic load,
presenting also the effect of varying $v$. Logically, $\Omega$ increases with decreasing $G$,
which means that the network topology changes more and more between packet
arrivals, since $t_{\text{iat}}$ becomes longer with decreasing $G$. Consequently, with decreasing $G$, it becomes progressively more likely that the routes will become obsolete between packet arrivals. This also increases the probability of route repair fail-
ures, leading finally to increased packet loss.

The average number of route breakages in both mobile and stationary scenar-
ios for BC-MAC and 802.11 are shown in Fig. 54. In the mobile scenario, the
number of route breakages increases with decreasing $G$, approaching slowly the
situation where routes must be re-established for every arriving packet (~ 93750
in our case). Route breakages (as packet loss) have a certain minimum under a
certain traffic load, which is dependent on the used setup. After the minimum
(with increasing $G$), the breakage count starts to increase again, but now because
the network gets congested. In a stationary network, no breakages are caused by
topological changes, but still the point of congestion seems to be about the same
in both mobile and stationary scenarios. This highlights the network behavior: Mobility dominates the performance behavior under a low traffic load, while the effect of the data traffic itself becomes significant under a high load.

Fig. 53. Relative mobility as a function of offered traffic load.

Fig. 54. Number of route breakages (20-node mobile scenario and stationary scenarios for BC-MAC and 802.11). Figure reprinted from (Prokkola & Bräysy 2007), © 2008 Elsevier.
Another interesting phenomenon caused by $\Omega$ is that the route lengths tend to increase with a decreasing traffic load, as we saw in Fig. 51. Since we are not using AODV ART, AODV always tries to repair the broken routes before forming new ones, and thus the likelihood of the need for route repairs increases with increasing $\Omega$. With continuous mobility, this in the long run leads to the use of suboptimal routes. This is the reason why BC-MAC does not enjoy shorter routes under mobility even though predicted by the theory. The same behavior is present also with 802.11, but is less eye-catching, since the routes are clearly shorter on average, and also, BC-MAC has higher probability of route repair success, enhancing the phenomenon.

Yet one more observation from the packet loss figure is that, despite the fact that non-connected situations are now included, average packet losses below the theoretical bound are achieved. The reason for this is of course simple, since, as already stated, the node distribution under RWM is no longer uniformly random but rather, somewhat concentrated to the center of the network area, enhancing connectivity. In the connectivity analysis, uniformly random node distribution was assumed.

Keeping the routes functional for as long as possible also has the negative effect of increased latencies. The end-to-end delay behaviors of BC-MAC and 802.11 in stationary and mobile scenarios are presented in Fig. 55. As seen, the average delay (in the mobile case) of 802.11 is only about half of the delay of BC-MAC under a low traffic load. BC-MAC is by default set to try the transmission 20 times before making a decision on a failed link. Despite the faster reTx try (lack of an exponential back-off algorithm), it probably takes longer than with 802.11 to notice a broken route, causing increased latency. In addition, because of BC-MAC’s ability to be more successful in keeping the links up, BC-MAC can hold up even low quality links, which potentially need several retransmissions to get data through, increasing also the latency. In the same figure, there is also a curve of BC-MAC where the reTx limit is dropped to 7, as in 802.11. As seen, the delay with this setup is dropped to the same level (or even lower) than with 802.11. Naturally, the ability of BC-MAC to maintain very long links is decreased, and packet loss is somewhat increased, but it still is well below that of the setups with 802.11 (not shown).

It can be concluded that there is a tradeoff between the capability to maintain routes functional and the readiness to detect topological changes in the network. The number of reTx tries is strongly tied to this relationship. Since in a stationary network there are no topological changes, a higher reTx limit (within reasonable
limits) typically leads to better performance, but in a mobile network, a high reTx try limit can have also negative effects. Enabling AODV ART would pass the responsibility of topological change detection more towards the network layer, making the tradeoff more complicated.

From the delay figure, we can also examine how the delay tends to increase with a decreasing traffic load. This is a consequence of an increasing number of route re-establishments, caused by increasing $\Omega$. Similar behavior is not seen in the stationary node scenario as the routes do not become obsolete. It can be concluded that $\Omega$ has a major effect on the performance at least in on-demand type ad hoc networks. Thus, in a mobile network, also very low traffic loads can be problematic. Enabling AODV hello messages would ease the effect of high $\Omega$, at least for 1 hop routes, and also table driven routing protocols might work better, but the cost is, naturally, that a lots of extra transmissions would be needed in these both, making the network operation inefficient.

![Fig. 55. Average end-to-end data packet delay (a comparison between mobile and stationary scenarios). Figure reprinted from (Prokkola & Bräysy 2007), © 2008 Elsevier.](image)

As stated, the functionality of CA in 802.11 is more accurate than in BC-MAC. This can be seen clearly in Fig. 56, which shows the average number of collisions per correctly received packet. The graph is presented as a function of a normalized total offered traffic load at the MAC level instead of the typical application level offered data traffic load. This is more illustrative because the collision metric is calculated at the MAC level. Presenting collisions as a function of the of-
fered data traffic load would give a correct idea of the overall high-level system behavior, including correct points of congestion, etc. However, presenting collisions as a function of the MAC level offered traffic load gives a truthful view about how the system behaves when considering the actual total traffic load pushed to the channel. Hence, this kind of presentation normalizes the effect of congestion on the different studied setups and gives a realistic view on the collision avoidance performance vs. total colliding traffic.

![Graph showing average number of collisions per correctly received packet as a function of the offered traffic load measured at MAC. (20-node mobile scenario).](image)

As expected, 802.11 RTS/CTS has the smallest number of collisions. Note also that RTS/CTS has more packets to collide, so if considering only data packets, the number of collisions with RTS/CTS would be even lower. This behavior is needed in centralized networks, where most of the traffic involves some central node (e.g., an access point). In a flat ad hoc network structure, where the traffic pattern is also more or less flat, the situation is completely different. Namely, the collision behavior of 802.11 RTS/CTS also indicates the lowest number of simultaneous successful transmissions. Since the number of collisions (including MAI) with BCCA-enabled setups is clearly higher than in other cases, it can be concluded that the two-channel structure of BCCA really does promote multiple simultaneous successful transmissions, thus boosting the network efficiency. This is
of course something that we knew already based on the results shown earlier, but now it has been also verified. Also, the figure clearly shows that, at maximum, BCCA-enabled setups can deliver MAC-level traffic loads of about 2, while 802.11 is able to deliver only about 1.3 and 802.11 RTS/CTS is saturated already to 1 (the theoretical maximum for a random access scheme where every station can hear each other).

This scenario showed that BC-MAC functions also in a mobile scenario. Even though BC-MAC cannot assist much in tackling the negative effects of mobility, it can offer support for clearly higher traffic loads when compared to 802.11. This study also brought up interesting phenomena like relative mobility, which has surprisingly clear effects on the performance.

4.5.3 Detailed performance evaluation in the time domain

Let us now continue with the current scenario and take a closer look at the performance behavior as a function of time instead of just average values. To compare 802.11 and BC-MAC in a single case, a completely stationary scenario with known topology, as shown in Fig. 57, is set. Now the focus will mainly be on a single iteration with an offered traffic load of 0.1, and the session model is set to DC with constant 100 packet sessions instead of the exponential session length to allow easier evaluation of the results as a function of simulation time. In addition to overall performance, nodes 0 (an example of an edge node), 8 (central location), and 17 (typical location) are taken under closer examination. Table 14 lists the special parameters used in this scenario.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, 802.11</td>
</tr>
<tr>
<td>Session model</td>
<td>DC (100 packets per session)</td>
</tr>
<tr>
<td>Mobility models</td>
<td>no mobility (predetermined topology)</td>
</tr>
<tr>
<td>$G$</td>
<td>$0.1 (+ 0.01$ with 801.11 and 0.5 with BC-MAC)</td>
</tr>
</tbody>
</table>
Fig. 57. The 20-node network topology used for detailed vector statistic analysis.

Fig. 58 (a) presents the throughputs of nodes 0, 8, and 17 with BC-MAC. As seen, there are heavy temporal variations in the throughputs as well variations between different nodes. Node 8 must route more traffic on average than the other nodes because of its central location. The node has a lot of neighbors, and, thus, a high likelihood that several routes involve this node. Node 17, on the other hand, sometimes receives high bursts of traffic, but its average throughput is quite low, while the edge node 0 has only a reasonably steady low throughput (< 25 kbit/s) and only a few traffic bursts exceeding this. Moderate temporal throughput variation is caused by the dynamic session model. High peaks are likely to be caused by route re-establishments and the following transmission of buffered data.
Interesting comparison of BC-MAC throughput behavior can be made with 802.11 shown in Fig. 58 (b). As seen, some similarity in the behavior exists, as, e.g., the throughput of the edge node 0 is somewhat smaller compared to nodes 8 and 17. However, throughputs are clearly higher than with BC-MAC, and also the temporal variation is very high. In addition, the variation is frequent and continuous as compared to the few bursts with BC-MAC. There are several reasons for this behavior. First, 802.11 operates in a common channel only, so the traffic will be received by all the neighboring nodes of the transmitting one, while in BC-MAC only the intended destination will receive the directed traffic because of R-codes. Another reason is that there is simply more traffic to be received with 802.11. This can be verified from Fig. 59, which shows the total offered network load at the MAC level as a function of the offered traffic load (the scenario studied in section 4.5.1). As it was seen previously, with the studied offered load of 0.1, the 802.11 based network is already getting congested, and the control load of the network is high (Fig. 41), thus increasing the overall traffic load.
The high variation of the throughput seen in Fig. 58 (b) is related to the fact that 802.11 has trouble handling the offered data traffic load, causing numerous route breakages and re-establishments. This can be seen in Fig. 60 (a), which shows the distribution of route establishment times. As seen, with 802.11 there are a large number of route establishments. Most of the establishments are quite short (the median is 40 ms), but there are also several establishments taking over 1 s and even about 5 s. The distribution seems to be long tailed (or heavy tailed) as the mean value ~ 137 ms > median, and also standard deviation ~ 280 ms > mean. More precisely, a distribution is said to be heavy tailed if (Willinger et al. 1997)

\[
\lim_{x \to \infty} F^{-1}(x) = x^{-r} Y(x), \quad 0 < r < 2;
\]

where \( F^{-1}(x) \) is the inverse Cumulative Distribution Function (CDF) of random variable \( X \), and \( Y(x) \) is a slowly changing function. From the definition it is be seen that the tail decreases slowly as a function of \( x \). We do not now separate long and heavy tailed behaviors, but instead we refer both as long tailed.

Long tailed behavior is also observed in the case of BC-MAC (Fig. 60 (b)), but with quite different values: median ≈ 6 ms, mean ≈ 12 ms and standard deviation ≈ 37 ms. Also, only 115 route discoveries are made with BC-MAC, in contrast to the 19,050 discoveries with 802.11. The maximum route establishment

\[
\text{Fig. 59. Total offered load (MAC level) as a function of the offered traffic load in a 20-node random ad hoc network (stationary).}
\]
time with BC-MAC is ~ 330 ms, which is only a single value, and the other values are below 60 ms.

Fig. 60. Distribution of route establishment time with 802.11 (a) and BC-MAC (b) (G = 0.1).

The large number of route establishments with 802.11 will most certainly affect the data packet end-to-end delay. This is proved in Fig. 61, which shows the delay distributions for 802.11 and BC-MAC. The delay distribution of BC-MAC decreases sharply, but again the distribution of 802.11 seems to be long tailed. The long tailed behavior of the delay is verified in Table 15, which summarizes the statistical properties of the delay distributions. It is interesting to see that the delay behavior of BC-MAC is not long tailed, and in fact the tail is shorter than with exponential distribution (with exponential distribution, mean = standard deviation). The minimum delay with both setups is 4.6 ms. This is, naturally, achieved in a situation where there is no competing traffic (immediate transmission), the first transmission is successful, and the destination node is one hop away, since 4.6 ms is the transmission delay (+ negligible propagation delay) of a 4,096 bit data packet including MAC headers and PLCP.
802.11 experiences unacceptably high average jitter of almost 400 ms, while the jitter of BC-MAC is only ~ 11 ms, which can be easily compensated for with buffering. BC-MAC performs very well, as 95% of the time it can maintain delays below 33 ms, and 99% of the time delays are still below 51 ms, which is well enough, e.g., for real-time multimedia applications.

Table 15. Statistical properties of the end-to-end data packet delay samples of BC-MAC and 802.11 in the studied case ($G = 0.1$).

<table>
<thead>
<tr>
<th>Property</th>
<th>802.11 ($G = 0.1$)</th>
<th>BC-MAC ($G = 0.1$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean [s]</td>
<td>0.0907</td>
<td>0.0142</td>
</tr>
<tr>
<td>Standard deviation (jitter) [s]</td>
<td>0.3894</td>
<td>0.0114</td>
</tr>
<tr>
<td>Max [s]</td>
<td>28.7441</td>
<td>0.5471</td>
</tr>
<tr>
<td>Min [s]</td>
<td>0.0046</td>
<td>0.0046</td>
</tr>
<tr>
<td>Median [s]</td>
<td>0.0153</td>
<td>0.0119</td>
</tr>
<tr>
<td>Number of Samples</td>
<td>86420</td>
<td>93531</td>
</tr>
<tr>
<td>95% of the values are below</td>
<td>0.4029</td>
<td>0.0334</td>
</tr>
<tr>
<td>99% of the values are below</td>
<td>1.1563</td>
<td>0.051</td>
</tr>
</tbody>
</table>

The long tailed delay behavior observed with 802.11 is a very interesting discovery, but this kind of behavior might not be only a feature of 802.11. With $G = 0.1$, 802.11 was already in a congested state, while BC-MAC was not. Moreover,
route establishment duration time seemed to be long tailed for both setups. Hence, we will make a null hypothesis that the long tailed delay behavior is caused by network congestion rather than the setups (802.11 or BC-MAC). Thus, we make another test where we now select the offered traffic loads to be such that the network with BC-MAC will be congested but 802.11 will be not. To achieve this, $G = 0.5$ is chosen for BC-MAC, and $G = 0.01$ is chosen for 802.11. The statistical properties of the delay behavior in these cases are shown in Table 16.

Table 16. Statistical properties of the end-to-end data packet delay samples of BC-MAC ($G = 0.5$) and 802.11 ($G = 0.01$).

<table>
<thead>
<tr>
<th>Property</th>
<th>802.11 ($G = 0.01$)</th>
<th>BC-MAC ($G = 0.5$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean [s]</td>
<td>0.0106</td>
<td>0.0983</td>
</tr>
<tr>
<td>Standard deviation (jitter) [s]</td>
<td>0.0103</td>
<td>0.3985</td>
</tr>
<tr>
<td>Max [s]</td>
<td>1.4834</td>
<td>8.3383</td>
</tr>
<tr>
<td>Min [s]</td>
<td>0.0046</td>
<td>0.0046</td>
</tr>
<tr>
<td>Median [s]</td>
<td>0.0098</td>
<td>0.0283</td>
</tr>
<tr>
<td>Number of Samples</td>
<td>93700</td>
<td>94000</td>
</tr>
<tr>
<td>95% of the values are below</td>
<td>0.0205</td>
<td>0.2782</td>
</tr>
<tr>
<td>99% of the values are below</td>
<td>0.0372</td>
<td>1.5961</td>
</tr>
</tbody>
</table>

The null hypothesis is verified: Clearly, a congested BC-MAC based network also leads to long tailed delay behavior, while a non-congested 802.11 network does not. The very same properties in the delay behavior of a congested BC-MAC network are seen as were seen with congested 802.11. The maximum delays with BC-MAC are several seconds and 95% of the values are only below ~ 280 ms, not to mention the very high jitter of about 400 ms, making the quality unacceptable for real-time applications. Non-congested 802.11 also has high maximum delay, but this can be considered as an outlier since 99% of the values are below 37 ms.

A BC-MAC based network is able to handle several times more traffic before congestion than an 802.11 based network. Also, packet loss behavior is greatly enhanced by introducing BC-MAC, and, in fact, under $G = 0.5$, BC-MAC still has a very low packet loss ratio as compared to the loss of 802.11 under $G = 0.1$. However, delay behavior is very similar with both setups in the congested network state, proving that the observed phenomenon is not caused by the used MAC and PHY techniques. As was seen in Fig. 61, the long tailed delay behavior seems to be rather well-controlled (the distribution is smooth and free of high peaks), suggesting that the behavior might be modeled with some theoretical dis-
tribution. One such a long or heavy tailed distribution is the Pareto distribution, the PDF (probability density function) of which can be written in the form (Gordon 1995)

\[ f_X(x) = \frac{\alpha x^\alpha}{(x + \beta)^{\alpha+1}}, \quad x \geq 0. \]  

(31)

Now we can test fitting the Pareto distribution to the results. One way is to calculate the parameters \( \alpha \) and \( \beta \) from the moments (Gordon 1995)

\[ E[X^n] = \frac{n! \beta^n}{\prod_{i=1}^{n-1} (\alpha - i)}, \quad \alpha > n \]  

(32)

by fitting them to the measured data, but usually this does not lead to good fitting in the case of long tailed distributions, as seen in (Prokkola 2001). The maximum likelihood method would lead to the best fit, but we are not interested in the absolute fitting but rather in whether the Pareto distribution could even be used for modeling the delay behavior in this case. Hence, approximate fitting is done by adjusting the parameters on the basis of inverse CDF log-log plotting, where the inverse CDF is plotted to a log-log scale (Willinger, et al. 1997). This gives a good overall fitting to the tail and also reveals the tail behavior.

The log-log plot of inverse CDF is shown in Fig. 62 with the approximate Pareto fit. As seen, the Pareto fit models the tail behavior surprisingly well. The figure also shows an exponential fit, which, clearly, fails to model the long tailed behavior of the delay. The tail of the simulated delay bends slowly or follows almost a straight line in the log-log scale, which is a clear sign of long or heavy tailed behavior. The fitted Pareto distribution has \( \alpha \approx 1.23 \) and \( \beta \approx 0.03 \). The used form of Pareto distribution is heavy tailed when \( 0 < \alpha < 2 \), and, if we take a look at the moment equation 32, we see that the second moment does not exist when \( \alpha = 1.23 \) (more precisely, the second moment does not exist when \( \alpha < 2 \)). The meaning of this is that the variance does not exist or may be considered to be infinite, making the behavior unstable.

In practice, if the delay is to be modeled accurately with the current form of Pareto distribution, we should also consider the limitation that the real delay has a lower limit, while the used distribution starts from zero.
As a conclusion of the 20-node default network case study, this closer examination reveals interesting characteristics of the studied scenario and setups: It seems that when a multi-hop ad hoc network is driven to a congested state, the delay will start to follow long tailed distributions even though the traffic models are traditional (not long tailed). In a non-congested state, however, network delay behaved as could be expected on the basis of the used traffic models, i.e., since the interarrival distribution follows exponential distribution, delay behavior could be expected to follow traditional Poisson queuing delay models (e.g., M/D/1 model), taking, of course, into consideration the multi-hop nature (Gross & Harris 1997). The on-demand route establishment time seems to follow long tailed models regardless of the traffic load. When a network becomes congested, the number of route discoveries explodes and its effects become visible in the total end-to-end data packet delay behavior. Hence, the most probable cause for the observed long tailed delay behavior in the congested network state is in the nature of the on-demand type routing protocol and multi-hop ad hoc network. In addition to delay behavior observation, this test showed that there are lots of differences in the traffic loads in different parts of the network, making, e.g., the problem of charge more difficult.
4.6 The effect of the AART parameter and mobility on the performance

In AODV, there is an interesting parameter called active route timeout (AART). The purpose of this parameter is to “predict” the route breakage before it actually happens, so that the MAC does not need to try multiple retransmissions before noticing the broken link and the routing protocol does not have to try to repair the route, since both processes can be quite slow. The problem, however, is that when this prediction fails, i.e., the parameter value is too small; it can lead to considerably worse performance than without this timeout. One example is a completely stationary network, where one can anticipate that the use of the route timeout functionality will only ruin the performance of the network.

A compact study for discovering the overall effects of this parameter and its relationship to the level of mobility is carried out. The main point is to show that as AART is a network layer parameter it works with BC-MAC, i.e., BC-MAC does not fail under the use of AART. In addition to different values of AART, different mobility parameters will be tried in this test case, while the rest of the thesis focuses only on MLV and stationary network. This study is restricted to BC-MAC, since the number of different cases will easily become too large to handle, putting too much weight on this test case. The following table summarizes the special parameters used.

Table 17. Simulation parameters of the AART test case.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC</td>
</tr>
<tr>
<td>Area</td>
<td>500 m × 500 m</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20 active</td>
</tr>
<tr>
<td>Mobility models</td>
<td>no mobility, stationary nodes;</td>
</tr>
<tr>
<td></td>
<td>MP (Pedestrian mobility), node speed: 0.2–1.5 m/s;</td>
</tr>
<tr>
<td></td>
<td>MLV (Light vehicular mobility), node speed: 0.2–6.0 m/s;</td>
</tr>
<tr>
<td></td>
<td>MV (Vehicular mobility), node speed: 0.2–20.0 m/s.</td>
</tr>
<tr>
<td>AART</td>
<td>2 s, 20 s, Infinity (Inf)</td>
</tr>
</tbody>
</table>

At first, we take a look at the network control load as a function of $G$ in Fig. 63, which presents the stationary and pedestrian mobility (MP) cases. As seen, AART has a drastic effect on the control load behavior. In a stationary case, there is no need to cut the routes, and, thus, choosing not to use AART (setting to infinity) is clearly the best choice and gives a steady control load of about 0.4. If AART is
There is an increasing trend in the control load with decreasing $G$, being at worst about 2.5 on average. The smaller AART is, the earlier the control load begins the rise as a function of decreasing $G$. This is quite obvious, since the smaller $G$ is, the longer is the IAT (Interarrival Time) of packets, and, thus, with smaller AART it becomes more probable that a route will time out between packet arrivals. Thus, it is also obvious that with a decreasing traffic load, the increase of the control load finally saturates, representing a situation where routes must be re-established for every arriving data packet. The effect of AART in a stationary scenario is very similar to the effect of relative mobility in mobile scenarios. This is not a surprise, since AART tries to predict the loss of routes caused by the relative mobility.

![Fig. 63. Network control load [bit/bit] with different AART values in stationary and MP scenarios.](image)

However, in the case of MP, we notice that turning off AART no longer results in better control load performance, but approximately similar performance as in the case of AART = 20 s. This suggests that the benefits of AART become visible already with the slow MP. This can be verified from packet loss behavior shown in Fig. 64: With MP, turning off AART results in the worst performance. With AART of 2 s or 20 s, there is no big difference in packet loss. It seems that just before the point of congestion, the performance is better with AART of 20 s, but this cannot be drawn as an exact conclusion, since there is some clear statistical...
undulation in the graphs. Nevertheless, the difference is small. Since the packet loss is about the same with AART of 2 s and 20 s, but there is a clear difference in the network control load in favor of AART of 20 s, it can be concluded that with MP one should use a longer AART than 2 s but still not turn the functionality off. Defining the exact optimal value is out of scope of this work.

![Graph showing packet loss ratio with different AART values in stationary and MP scenarios.](image-url)

**Fig. 64. Packet loss ratio with different AART values in stationary and MP scenarios.**

When looking at the packet loss ratio performance in a stationary scenario, the observation is clear: The longer AART is, the better the performance, and the best performance is obtained by turning off the whole route timeout functionality. Interestingly, it is seen that without the route timeout functionality, the network can easily reach practically zero packet loss, but when AART is used, the network acts as a mobile network from the performance point of view, especially when looking at the trend of the stationary case with AART = 2 s; it follows the MP curves very well with AART < Inf. The packet loss saturates to about $4 \times 10^{-4}$, not to mention the high network control load, ruining the performance for nothing. As a conclusion, it can be stated that in a stationary or nearly stationary network, which, e.g., sensor networks often are, AART should not be used, or at least the timeout should be very long.

Next, faster mobility, light vehicular (MLV) and vehicular (MV) mobilities in particular, will also be included in order to see their effects. For examining the effects of mobility, it is more convenient to fix the traffic load and show the re-
sults as a function of mobility. In practice, the results will be shown as a function of maximum node speed. As the fixed offered traffic load, we take two cases: a lightly loaded network case, where $G = 0.001$, and a highly loaded, but not yet congested, network case ($G = 0.3$).

Packet loss as a function of maximum node speed is shown in Fig. 65. The first observation is that the curves cross each other in various places and the overall trend is hard to find especially with low node speeds. However, with closer inspection one can extract the relevant information. It is seen also in this form of presentation that in a stationary network, the performance is always the best when AART is not used, and practically zero packet loss can be reached under a low traffic load. In particular, when small AART (2 s) is used in a low traffic stationary scenario, packet loss becomes quite high. The main packet loss trend rises as a function of mobility. However, with increasing mobility, packet loss with AART of 2 s does not increase as much as it does with longer AART, and, in fact, AART of 2 s turns out to give the best performance beyond MLV speeds. It is also seen that the rising trend of packet loss is not as steep with AART of 2 s as compared to other values. This is intelligible, since topology will change quicker with faster node mobility, and, thus, a small AART will succeed better in estimating the route breakages.

![Fig. 65. Packet loss ratio as a function of mobility level with different AART values under traffic loads of 0.001 and 0.3.](image-url)
It can be observed that under a low traffic load and an AART of 2 s, the packet loss at first even decreases with increasing mobility. This is related to the fact that light mobility can help in connectivity issues, and also that the use of AART begins to show some benefits with increasing mobility. However, at some point, the benefits of mobility turn into disadvantages, since, e.g., with vehicular speeds, the topological changes of the network are already so rapid that the routing protocol has difficulties in forming the routes as fast as needed. Therefore, after some point, we typically see a constantly rising trend in packet loss as a function of mobility. By increasing mobility further, the network would at some point become practically unusable, but, as seen, even vehicular speeds do not yet cause this, but instead the packet loss level is still quite reasonable, being about 0.1% or below. Thus, it can be concluded that BC-MAC with AODV can handle even high mobility scenarios well.

Interestingly, there are great differences in the trends of curves with the same AART but with a different traffic load. From these graphs, drawn as a function of mobility, this behavior is difficult to explain. This highlights the importance of evaluating network performance as a function of offered traffic load, which is the main method of presenting and analyzing results in this thesis.

Next, end-to-end delay jitter is presented in Fig. 66. This figure gives quite the same information as the packet loss figure. In a stationary case, AART should not be used, and low jitters of about 5 ms under a low traffic load and 20 ms under a high traffic load are reached in a stationary network. However, if AART is not used, the jitter increases rapidly with an increasing level of mobility, and the benefits of using AART become evident. Already with pedestrian mobility speeds, AART of 20 s becomes the one that gives the performance. Then, by further increasing the speed of the network nodes, AART of 2 s will begin to perform better than AART of 20 s, at node speeds of about 4 or 5 m/s depending on the traffic load. This clearly suggests that AART affects the ad hoc network performance a lot, and, obviously, the AART value should be tied to the level of network mobility in order to reach the optimal performance. Beyond vehicular speeds, it is seen that jitter has a rising trend similar to packet loss.

In general, this test case corroborates the assumption that mobility and handling of it is a network layer problem, and the routing protocol has an important role in this, while lower layer design is more about the traffic handling capability. Nevertheless, lower layer design can enhance the performance under mobility by giving abilities like BC-MAC being much more robust in keeping the links up
than 802.11. However, the routing protocol needs to be able to use these capabilities as efficiently as possible, giving rise to cross layer design.

Fig. 66. Delay jitter as a function of mobility level with different AART values under traffic loads of 0.001 and 0.3.

The effects and optimization of AART have been considered, e.g., in (Taramaa 2005), which also found that this parameter has a considerable effect on performance. Thus, this should be taken into account when planning a real ad hoc network, and, clearly, adaptive AART would be the best choice. However, as stated earlier, from now on, the focus is only on the case of AART = Inf, since this case is the one that is the most illustrative for presenting the work of this thesis.

4.7 BC-MAC’s dependency on the spreading factor

Since the BCCA method is based on the use of CDMA, it needs a big enough spreading factor, firstly, to allow a wide enough spreading code family size for the nodes and, secondly, to allow a big enough processing gain to separate the overlapping transmissions. As seen earlier, 802.11 is quite robust for the spreading factor, since it does not exploit the available processing gain anyway. For this reason, a brief study will be carried out, showing how dependent BC-MAC’s performance is on the spreading factor. Naturally, this is a quite wide topic, and the dependency is related to a lot of things like network size, node density, propagation environment, etc. Thus, it is not possible to make a full analysis here, but the
study will give a good insight into the topic. The standard network scenario is used, and the following table summarizes the special parameters used:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, BC-MAC (C-code), BC-MAC (NSCD), 802.11 (20 reTx tries)</td>
</tr>
<tr>
<td>Mobility models</td>
<td>no mobility</td>
</tr>
<tr>
<td>Spreading factor</td>
<td>1, 4, 11, 63, 128</td>
</tr>
</tbody>
</table>

As seen in Table 18, several different values of $F_{sf}$ are tested with different setups. Testing NSCD is interesting, since it should be especially vulnerable to a small spreading factor. We also test pure BC-MAC without BCCA (i.e., using BC-MAC with C-code only). 802.11 will be the point of comparison, but the number of reTx tries is set to the same level as with BC-MAC to enable a fully fair comparison. Only the stationary scenario is studied, since it is simple enough to allow the effect of $F_{sf}$ to be extracted from the results.

### 4.7.1 Simulation results and analysis

The most obvious effect of diminishing $F_{sf}$ is that the effect of MAI will be more severe. Thus, the number of possible successful simultaneous transmissions is, on average, reduced. This can be observed most easily from Fig. 67, which shows the number of collisions per packet. The collisions graph is again presented as a function of the normalized total offered traffic load at the MAC level instead of the typical application level offered data traffic load. As the figure shows collisions per correctly received packets only, it gives a good insight into how the different cases are able to exploit the simultaneous successful transmissions.

As expected, Fig. 67 shows that with a higher spreading factor, overlapping transmissions can be separated more easily: The highest value of $F_{sf} = 128$ allows the highest number of collisions (simultaneous successful transmissions) in the studied cases. The figure also shows the effect of NSCD. While earlier in Chapter 4.5.1 we saw that there were hardly any effects from the use of NSCD, we now see that with smaller $F_{sf}$, the effect of using NSCD becomes more visible: It clearly worsens the advantage of the BC-MAC’s key idea, which is the ability to benefit from simultaneous transmissions. This behavior is quite understandable, since random spreading codes are used and there is no mechanism for optimizing the
spreading code usage. Since Gold-code-like spreading codes with a family size of $F_{sf} + 2$ are assumed, with $F_{sf} = 4$, there are only 6 codes available, and, hence, collisions of the transmissions with the same codes are very common. At this point, it should be also recalled that in reality, with bit wise spreading, the spreading code length $= 4$, which allows only $2^4 = 16$ different spreading codes, all of which are not even usable. Hence, the case of $F_{sf} = 4$ without NSCD, i.e., 20 unique codes for all 20 nodes could not be done in reality in the way it is simulated here. However, this is an interesting and important case from the theoretical point of view. Also, we could imagine that the case without NSCD represents a case of code usage optimization method, distributing the available codes optimally.

![Fig. 67. Number of collisions per correctly received packet with different values of $F_{sf}$ as a function of offered traffic load measured at MAC (BC-MAC with and without NSCD).](image)

When the ability to separate overlapping transmissions is weakened, it makes the collisions more severe. Effectively, this causes reTx failures at MAC, since the channel becomes less reliable with smaller values of $F_{sf}$, as seen in Fig. 68. One important observation is that with a sufficiently high spreading factor, the performance saturates. As seen, increasing $F_{sf}$ from the default 63 to the considerably higher value of 128 has practically no meaning to the reTx failure behavior, and the same is seen also in the case of collisions (Fig. 67). This is quite obvious, since with a certain node density and traffic load, there is on average a certain maximum amount of simultaneous transmissions in a node area. Hence, suffi-
ciently high $F_{sf}$ will allow the separation of all the overlapping transmissions, on average, and, thus, increasing $F_{sf}$ beyond this point no longer increases the average performance. Calculating this “sufficiently high $F_{sf}$” is out of the scope of this thesis and is not an easy task, since it is a function of at least node distribution, traffic load, channel model, and effective radio range.

Fig. 68. Number of re-transmission failures at MAC with different values of $F_{sf}$ (BC-MAC with and without NSCD).

Lower $F_{sf}$ causes an increased number of reTx failures, as was assumed. Between $F_{sf} = 63$ and $11$, there is already a clear difference in the reTx failure performance. However, when $F_{sf}$ is dropped to $4$, the performance is considerably worse than with $F_{sf} = 11$. In addition, it is seen that the lower $F_{sf}$ is, the more the performance suffers from the use of NSCD, as was anticipated on the basis of the NSCD operation. With $F_{sf} = 11$, the effects are already clearly visible, and with $F_{sf} = 4$, the use of NSCD almost doubles the number of reTx failures under a high traffic load.

Next, packet loss ratio behavior is depicted in Fig. 69. As could be assumed on the basis of the reTx failure behavior, smaller values of $F_{sf}$ will result in an increased packet loss ratio. It is also seen that increasing $F_{sf}$ beyond $63$ does not affect the packet loss ratio behavior, so $F_{sf} = 63$ can be hence safely assumed to be sufficiently high for this scenario to get the most out of BC-MAC in terms of separating overlapping transmissions. In fact, it is likely that the sufficiently high value is even less, being somewhere between $11$ and $63$. The effect of NSCD is
also seen in packet loss. However, not until with a very small value of $F_{sf} = 4$ will the use of NSCD clearly deteriorate the packet loss performance. Furthermore, even though the performance of the worst case has a distinct difference as compared to the best case, the performance is not yet poor, and, in fact, it is still better than that of 802.11. Thus, despite the pessimistic assumptions for the use of NSCD, it still performs quite well even with very small spreading factors.

![Fig. 69. Packet loss ratio with different values of $F_{sf}$ (BC-MAC with and without NSCD).](image)

Fig. 70 presents packet loss ratios of different setups than Fig. 69, and also includes narrowband and wideband versions of 802.11 (with 20 reTx tries). As seen, the performance of 802.11 is practically not affected by the processing gain even in this real ad hoc network situation: The cases of $F_{sf} = 4$ and 63 give about the same performance except when dealing with very low packet loss ratios under low traffic load. This verifies our hypothesis which we made on the basis of the results from the AP scenario earlier. Fig. 70 now also shows clearly that BC-MAC with NSCD and $F_{sf} = 4$ still clearly outperform 802.11.

Fig. 70 also shows an interesting curve of BC-MAC, where $F_{sf} = 1$. This means that there is no spreading, no extra processing gain, and hence no prerequisites for CDMA. However, we still make an assumption that there is a mystical way of coding the transmissions in order to use the BCCA principle. In practice, one such a way could be the use of long scrambling code over several bits or even over the whole PHY frame. Hence, even though there are very poor chances of separating overlapping transmissions, the nodes can see only transmissions meant
for them, and the basic principle of BCCA works. This special setup also highlights the potential of simulation based studies, since this kind of situation would be very hard to test with real equipment. If we now compare the performance of this special setup to the BC-MAC that uses only C-codes (BCCA is disabled), we see from Fig. 70 that there is a quite big gap in packet loss performance between these two. BC-MAC without BCCA performs clearly worse even though it has the wideband channel available with $F_{sf} = 63$. It also performs clearly worse than 802.11, while the special case of BC-MAC with $F_{sf} = 1$ has about the same level of performance as 802.11. This interesting test clearly shows that the key idea of BCCA truly is the way in which it is able to separate simultaneous transmissions. BC-MAC tries to encourage simultaneous transmissions with looser CA methods than 802.11, and, hence, it performs poorly without BCCA. This test again highlights the failure of 802.11 in an ad hoc environment: Efficient CA can be counterproductive, since instead of blocking the simultaneous transmissions, the lower layers should provide a way to allow an efficient usage of the spatial capacity of ad hoc networks. Even though BCCA and BC-MAC do not yet optimally use the full potential of the spatial capacity of ad hoc networks, they already show that very high gains are achievable through the lower layer design.

![Packet loss ratio with different values of $F_{sf}$ (BC-MAC with and without NSCD vs. 802.11 + BC-MAC without BCCA).](image)

Fig. 70. Packet loss ratio with different values of $F_{sf}$ (BC-MAC with and without NSCD vs. 802.11 + BC-MAC without BCCA).
There were no surprises in delay, jitter, or other performance metrics, so they are not shown and we can draw conclusions from this important study on BC-MAC’s performance dependency on the spreading factor. Clearly, as expected, decreasing the spreading factor decreases the performance of BC-MAC. Also, the smaller the spreading factor, the more the use of NSCD worsens the performance. However, BC-MAC did not seem to be very vulnerable to the \( F_{sf} \), since even with \( F_{sf} = 11 \), which is clearly smaller than the default spreading factor of this work, the performance was very good. In addition, even with the very small spreading factor of 4, BC-MAC still clearly outperformed 802.11, even though NSCD was used, leaving only 6 spreading codes for 20 nodes. As a result, even though this study was brief, it clearly proved that the functionality of BC-MAC does not collapse, even though wideband SS signaling is not always available.

4.8 Radio channel modeling and its effects on the performance

In the previous section, there was a test with BC-MAC without BCCA, and the result was that it performs worse than 802.11. However, in our previous work, it was shown that BC-MAC even without BCCA outperformed 802.11 (Prokkola & Bräysy 2004b). Even though contradictory, this is not an error. The reason behind this mismatch is that the study of (Prokkola & Bräysy 2004b) was carried out with a cut propagation model, while in this thesis we have been using more realistic propagation models. This is a nice example of how inaccurate lower layer modeling can lead to incorrect results, and not only gives an incorrect picture of the absolute performance, but can also mix the ranking between different systems, algorithms, and methods. It is often approximated that even though the absolute performance is perhaps not accurate, simplified models can still be used for comparing different setups.

The research community has noticed the effect of inaccurate modeling, since several real ad hoc network tests carried out in the field show that the results do not match well with simplified simulations or mathematical analysis (Chin et al. 2002, Kotz, et al. 2004, Lundgren, et al. 2002). In this section we will study the effects of propagation modeling and fading.
4.8.1 The effect of radio channel propagation modeling

Compared propagation models and scenarios

In the most simplified propagation models, no actual power attenuation is calculated, but the signal just arrives at the receiver inside a certain radius around the node. Such a simple model is not included in this study, but it is approximated in a way that is often used in ad hoc network studies: A high power cut propagation model (CPHP) is used as the most unrealistic model, in which a powerful signal \( (P_T = 100 \text{ mW}) \) propagating according to the free space loss is cut to 250 m even though a valid signal could be received far beyond this distance. Hence, the signal is very powerful even at the distance of 250 m, and, e.g., background noise has practically no effect. Another cut model used is a low power cut propagation model (CPLP), in which the default FSL model is used, but the signal is cut to the effective radio range of 250 m. Hence, CPLP is practically the default FSL model without accurate MAI calculation beyond the effective radio range.

The standard point of comparison is the FSL model, and as a more realistic propagation model, a forest propagation loss (FPL) model is chosen. FPL gives attenuation in a leaf-tree forest, presumably fitting as well as a propagation loss model to, e.g., a forest-like battlefield environment. FPL is a combination of three commonly known models: free space, plane earth, and Weissberger foliage attenuation models. The method is the same as used in the OPNET Path Attenuation Routine (OPAR) by the Mitre Corporation (Comparetto et al. 2003). At first, in the FPL model, both free space and plane earth attenuations are calculated as (in dB)

\[
L_{PE} = 40 \log_{10}(D) - 20 \log_{10}(h_{tx}) - 20 \log_{10}(h_{rx}) , \quad (33)
\]

\[
L_{FS} = 32.45 + 20 \log_{10}(D) + 20 \log_{10}(f) , \quad (34)
\]

where \( L_{PE} \) = attenuation by plane-earth model (in dB), \( D \) = total path length (in meters), \( h_{tx}, h_{rx} \) = transmitter and receiver antenna heights in meters (both have the value of 2 m in this study), respectively, \( L_{FS} \) = attenuation by the free space model (in dB), and \( f \) = frequency (in GHz). Using the Weissberger model for foliage obstruction loss, the actual foliage attenuation (in dB) is

\[
L_{weiss} = \begin{cases} 
1.33 f^{0.284} \times D_f^{0.588}, & D_f > 14 \text{ m} \\
0.45 f^{0.294} \times D_f^{1.0}, & 0 \leq D_f \leq 14 \text{ m}
\end{cases} , \quad (35)
\]
where, $D_f =$ the distance the signal propagates through the foliage. Finally, the total forest propagation loss is

$$L_{\text{FPL}} = L_{\text{wiss}} + \max(L_{\text{PG}}, L_{\text{fl}}).$$

(36)

In the FPL case, all the nodes are assumed to be inside a forest, and thus $D_f = D$. The FPL model has very high attenuation as a function of propagation distance as compared to the FSL model. Hence, also transmission power has to be considerably higher in order to achieve $d_r$ of 250 m. With FPL, we get the path loss for 250 m to be 132.12 dB when $f$ is 2.4315 GHz in the ISM band. Since antennas are omnidirectional, using the same principles as in the FSL case earlier we get the required transmitted power to be as high as 150 mW.

![Fig. 71. Received power level as a function of propagation distance. Figure reprinted from (Prokkola, et al. 2005), © 2008 IEEE.](image)

In the FPL model, the signal attenuates heavily, posing challenges to the communications as compared to FSL. However, in terms of MAI, the situation is quite interesting. Because of the high propagation loss, the long-range MAI is, in fact, clearly weaker than it is with FSL. This can be easily seen from Fig. 71, where the received power level is plotted as a function of propagation distance. In the FPL model, the curve’s starting point is naturally higher, since the transmission power is higher than in the FSL model. The figure also shows the signal detection level (about $-110$ dBm), which of course intersects with the other curves at the $d_r$ of 250 m. After this distance, the MAI under the FSL model dominates, since the
attenuation under foliage obstruction is much stronger. Take, for example, the interference level at the distance of 500 m from the transmitter. Under FPL, the interference level is about −140 dBm, while under FSL it is over 20 dB higher (about −116 dBm). On the other hand, the short range (< d₀) MAI is clearly stronger with FPL than it is with FSL.

The following table summarizes the basic simulation parameters used in this study case. The standard 20-node scenario will be used, but also dense and oblong 50-node scenarios will be tested. In our earlier work (Prokkola, et al. 2005), we carried out a similar study in an oblong network scenario of 50 nodes in an area of 1500 m × 300 m, but some of the parameters were different than here. In addition to the propagation model, it is interesting to compare how the scenario affects the results.

Table 19. Special simulation parameters of propagation model comparison.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, 802.11</td>
</tr>
<tr>
<td>Propagation models</td>
<td>CPHP, CPLP, FSL, FPL</td>
</tr>
<tr>
<td>Area</td>
<td>500 m × 500 m (+ a test in 1500 m × 300 m)</td>
</tr>
<tr>
<td>Mobility models</td>
<td>no mobility</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20 active (+ a test with 50 active nodes)</td>
</tr>
</tbody>
</table>

Results of a 20-node stationary network

Fig. 72 presents the average number of collisions per received MAC packet with BC-MAC under different propagation models. No major differences exist between the two real propagation models (FSL & FPL) and between the two cut propagation models (CPLP & CPHP). However, under real propagation models, the number of collisions is several times higher than with the cut propagation models, as anticipated. This is the first proof of the failure of cut propagation models, as they fail in modeling the effect of MAI.

Unlike often assumed, this modeling failure has a direct impact on the network layer (and beyond) performance. Consider, e.g., Fig. 73, which shows the average number of route breakages. The basic trend with the models is similar, except CPHP, under which, no route breakages happen before network congestion (offered load of > 0.6). The functional difference between CPHP and CPLP from the network layer perspective is in the stability of long links. In CPLP, the signal is barely receivable near the limit of 250 m, and thus even a weak interfering sig-
nal will cause a long link to fail. In CPHP, the situation is totally different, since the signal is very strong even at the edge of the range. Also, since the background noise has practically no effect on the performance, the processing gain will mostly be able to handle the collisions. Hence, under CPHP, retransmissions will practically always be successful within the reTx limit, which explains the route breakage behavior. This interesting behavior highlights the failure of CP modeling and also the effect of transmission power in the cut models.

![Graph showing the number of collisions per received MAC packet between different propagation models (BC-MAC).](image)

Fig. 72. Number of collisions per received MAC packet between different propagation models (BC-MAC).

It is interesting to note the behavior of the network under the FPL model: Under a low traffic load, route breakages happen even less than under the CPLP model, but breakages increase with an increasing traffic load, and finally they even exceed the breakages under FSL. This phenomenon can be explained quite easily if we recall the power attenuation behavior. The attenuation under the FPL model is so strong that it effectively behaves similarly to a CP model when the network traffic load is very low: The interference power far away from the transmitter is almost negligible. However, as the network traffic is increased, the total interference power in the network is also increased, and, as a consequence, it makes the CP approximation fail with an increasing traffic load. Also, because of the high transmission power, the interference level is high near the interfering node, and,
thus, under a high traffic load the overall network interference level will finally exceed that of FSL, leading to an increased number of route breakages.

Fig. 73. Number of route breakages between different propagation mechanisms (BC-MAC).

The great differences in route breakages will definitely also affect the higher layer performance. Fig. 74 shows this in the form of packet loss ratio. The failure of CP models is, again, seen at its worst in the CPHP case, as the packet loss ratio behavior is clearly the best of the compared ones. However, the differences are not as radical as found in (Prokkola, et al. 2005). One of the reasons is that in this study there are only 20 nodes as compared to the 50 nodes used in (Prokkola, et al. 2005), but also the network size and shape have an effect. Less nodes and a smaller network (less hops) makes the situation easier in terms of packet loss, and, hence, also the different modeling methods affect less.
The differences in packet loss caused by different propagation mechanisms are much clearer with 802.11, as seen in Fig. 75. This is because 802.11 is generally more sensitive than BC-MAC to MAI and channel sensing. Interestingly, the performance behavior of 802.11 under different propagation models is not similar to that of BC-MAC. For instance, if we take a look at the FPL curve of 802.11, it seems that FPL under a low traffic load gives worse packet loss than CPLP, but under a high traffic load, it gives even better packet loss than CPHP. This is most likely related to the implementation of CSMA, packet scheduling, and CA in 802.11, which differ from BC-MAC. Under a high traffic load, several nodes are trying to send simultaneously, raising the power level in the channel. Hence, the channel sensing can function over longer distances under FPL, i.e., the hidden terminals can have information about the ongoing transmissions, while under CP models this will not happen. Under high traffic load, this is advantageous since the performance of 802.11 is limited by collisions. Thus, a real propagation model can ease the hidden terminal problem and help with the timing of packets, leading ultimately to better performance than the cut models. The same phenomenon is present also under FSL, but in this the effect of MAI is so severe that it still leads to worse performance than the CP models. BC-MAC, on the other hand, is not sensitive to the hidden terminal problem or collisions, and, thus, the performance behavior is different. This is the most likely reason for the interesting behavior of
802.11 under FPL. However, this cannot be proved on the basis of these simulation results alone, but this proofing is not even needed, since, though an interesting finding, this is out of the scope of this work.

Throughput comparison between 802.11 and BC-MAC under different propagation mechanisms is shown in Fig. 76. As expected on the basis of the packet loss ratio, 802.11 under FPL gives the best performance as compared to the other 802.11 graphs. The same kind of observations were also made in (Prokkola, et al. 2005), but there FSL gave better results than CPHP (FPL with 802.11 was not tested in (Prokkola, et al. 2005)). Now FSL gives about the same performance as CPHP in terms of maximum throughput. Since the scenario itself is practically the only difference between (Prokkola, et al. 2005) and this test case, it has to be the reason for getting different results. Most likely, the oblong geometry of (Prokkola, et al. 2005) and longer distances are favorable to FSL. These findings are interesting, especially since BC-MAC still behaves quite similarly as in (Prokkola, et al. 2005) (i.e., the ranking and trends of performances caused by different propagation models are about the same). Because of the nature of 802.11, it seems to be sensitive to propagation modeling and also to different scenarios. These kinds of findings highlight the importance of accurate propagation modeling in simulations.
Results of a 50-node stationary network

Until now, we have only been able to guess the effects of node density on the network performance, but next we will test 50 nodes in the default 500 m x 500 m area, which will produce a dense network of 200 nodes/km$^2$. All of the 50 nodes generate traffic. We only test BC-MAC and 802.11 under CHPHP and FSL, since these should already show the main effects of node density vs. propagation modeling. This test will show how the different setups react to the changed node density.

Naturally, the first thing is the increased level of interference, which can be estimated from the number of collisions. With BC-MAC, the number of collisions is clearly higher in the dense network than in the 20-node network (not shown), being at minimum under CHPHP over 0.1 collisions per packet, while with 20 nodes the level was about a decade lower, as seen in Fig. 72. Under the FSL model, collisions of about 0.5 per packet are experienced (not shown). Nevertheless, the relative difference of collisions per packet between CHPHP and FSL is about the same in 20- and 50-node networks.
The next thing is that the denser the network is, the shorter the routes are (in hops). The average route length under CPHP is in the 50-node network about 1.60 hops and under FSL about 1.74 hops under a steady state traffic load (not shown), while with 20 nodes the corresponding values were 1.65 and 1.75, respectively. Hence, even though the differences are not big, it can be concluded that a higher node density leads to shorter routes, as could be expected. It also seems that a higher node density widens the difference of route lengths between CPHP and FSL. This is most likely because a higher node density gives an increased probability of finding shorter routes, which under CPHP are easily used, but because of the increased interference level under FSL, the nodes are not able to exploit the possibility of shorter routers that efficiently. The average route lengths in this scenario under CPHP with 20 and 50 nodes correspond quite closely to the optimal shortest path average route lengths of 1.61 and 1.57, respectively. The average shortest path route length is got with the connectivity analysis tool by averaging over a high number (>1,000) of different random topologies. The results mean that in this scenario under the CPHP model and a moderate traffic load, AODV performs practically as an optimal shortest path routing protocol. Under the FSL model, AODV probably still works well, but because of the interference, the longest links are not stable enough for long time usage, and the routing protocol is forced to use somewhat longer routes of shorter links.

Fig. 77 presents the packet loss ratio behavior of the 50 active node dense network scenario with 802.11 and BC-MAC. The figure also shows the performance of BC-MAC (under FSL and CPHP) for an oblong 1500 m × 300 m network scenario with 20 active and 30 passive nodes (passive nodes do not generate traffic, but will route it). The oblong scenario results were presented for the first time in (Prokkola, et al. 2005), where one can also find the results for 802.11. There are some small differences between the BC-MAC’s performance results shown in Fig. 77 and (Prokkola, et al. 2005). This is because in (Prokkola, et al. 2005), CDo was not used, and slightly better results are got here by using it.

As anticipated, Fig. 77 proves that the differences between FSL and CPHP in the 50-node dense network are greater than they were in the 20-node network (compare to Fig. 74 and Fig. 75). However, the differences are still not even near as big as they are in the 50-node oblong network. Because of the shape of the scenario, the routes become considerably longer in the oblong scenario, being >3 hops on average. More hops mean more transmissions, and thus they increase the overall network traffic, which in turn increases interference. More hops mean also a higher packet loss ratio, since the loss ratios of individual links will be cumu-
lated. This is also clearly visible in Fig. 77, as BC-MAC in the oblong scenario under FSL does not reach packet losses below the level of $10^{-3}$. This is a good example of the challenges of multi-hop communications. While BC-MAC gives practically zero packet loss in our standard network scenario, in the studied oblong network scenario packet loss saturates and never reaches zero, even though the average number of hops is only about doubled between these two scenarios. With 802.11 the situation is a lot worse: In the oblong scenario, as shown in (Prokkola, et al. 2005), the average packet loss always stays above 0.01, being practically useless for many applications.

Fig. 77. Packet loss behavior of 50-node dense and oblong networks under different propagation models with BC-MAC and 802.11.

As regards modeling, the errors of propagation modeling are also cumulated over multiple hops, thus causing bigger differences between different propagation models. Consider, e.g., BC-MAC in the oblong scenario: under FSL model, packet loss saturates to the level of $10^{-3}$, while under CPHP practically zero packet loss is achieved, giving, thus, far too optimistic view about the performance. Hence, the longer the routes are, the more important the accuracy of propagation modeling will be. Moreover, this does not mean only propagation modeling, but also lower layer modeling in general. It can be concluded that the node density
affects the level of error of CP modeling, but the shape of the topology (at least via route lengths) has greater effect.

One important observation made from Fig. 77 is that BC-MAC does not seem to be very sensitive to the node density. Even though the node density is raised from 80 nodes/km² to 200 nodes/km², packet loss behavior as a function of \( G \) has not changed considerably (compare to the packet loss in Fig. 74). Also, 802.11 has no extra difficulty to handle higher node density, so both of the examined setups can handle also higher node densities well. The situation might change at some point when increasing the node density further, but this speculation is pointless, since a more in-depth study on the effects of node density is out of the scope of this work.

Despite the fact that the packet loss performance as a function of \( G \) is about the same with both of the examined node densities, the per node performance is, naturally, very different. Hence, even though the points of congestion in the 50-node dense network are about the same as in the 20-node network, the throughputs of single nodes are, naturally, much worse. This can be seen in Fig. 78 where the packet loss ratio is plotted as a function of the average offered data traffic load per a single node (in kbits/s). With BC-MAC, the maximum traffic handling capability ratio between 20- and 50-node networks is close (or somewhat better most of the time) to the difference in node density 20/50 = 0.4 except the very low packet loss ratios. With 802.11, the ratio seems not to be that constant: Under a high traffic load, the 50-node network performs considerably worse than the 20-node network, but with a decreasing traffic load, this difference shrinks.

Both 802.11 and BC-MAC seem to handle the higher node densities quite well. An interesting point, however, is that the absolute performance difference between 802.11 and BC-MAC is notable. Compare, for example, achievable data rates before the point of congestion in the FSL case: As seen in Fig. 78, in a 50-node BC-MAC network individual users get clearly better data rates than the users in a 20-node 802.11 network. If, for example, a packet loss of \( 10^{-3} \) is taken as the limit for acceptable performance, every one of the 50 BC-MAC users can utilize about a 10 kbit/s average data rate, while 20 802.11 users under the same conditions can utilize only 1 kbit/s average rates. So, despite the 150% higher node density, BC-MAC can still offer 10 times the capacity per user as 802.11.

Per node capacity is an interesting topic in ad hoc networking. BC-MAC performs well also under higher node densities, but BC-MAC cannot eliminate one of the fundamental problems of ad hoc networking: Ad hoc network capacity is a decreasing function of the number of nodes in the network (Gupta & Kumar...
Nevertheless, a great improvement in performance over the traditional methods is gained.

![Graph showing packet loss behavior for different networks and MAC protocols](image)

**Fig. 78.** Packet loss behavior of a 50- vs. 20-node network under different propagation models with BC-MAC and 802.11 as a function of the offered mean data traffic load per a single node.

**Summary of the propagation modeling study**

Overall, these findings with 20-node and 50-node networks suggest that comparing different methods with inaccurate models might give a wrong idea of the performance. The results in (Prokkola, et al. 2005) already verified that totally incorrect absolute performance results can be got if highly simplified models are used. Now, we also got some proof that at least in some situations simplified models can also lead to errors when comparing different methods in addition to the errors in the absolute performance. Also, some protocols seem to be more sensitive to modeling than others, and hence different scenarios can result in very different performance behavior. Overall, the errors of the propagation modeling cumulate over multiple hops, which make accurate modeling more crucial in larger networks. This is of course somewhat frustrating from the simulation perspective, since simulating larger networks require more computational power, while increasing the simulation accuracy increases the computational power requirement.
further. In addition, scenario and node density affects the errors caused by modeling. Even though propagation modeling is far below the network layer in the communication stack, different propagation models still cause quite interesting effects in the performance measured above the network layer.

The most important proof considering this work is that BC-MAC outperforms 802.11 regardless of the propagation modeling. BC-MAC did not shown any weak points even when used in the difficult FPL channel. Also, the behavior of BC-MAC is somewhat more predictable and is not as sensitive to modeling errors as 802.11.

4.8.2 The effect of fading

In addition to MAI and path loss, there is also another phenomenon typically present in the wireless communications: fading. This causes high variability to the received signal power level, having a remarkable effect on the received SNR. It can be categorized as fast or slow fading, depending on the signaling interval and coherence time of the channel. Fast fading is caused by multi-path propagation of the radio signal, while slow fading is caused by obstacles (e.g., buildings, trees) on the radio path. In slow fading (or shadowing), the obstacles act differently in different locations, and hence the received signal strength will vary as a function of time under mobility (Sounders 1999). Some locations will suffer increased loss, while others will have less obstruction and an increased signal strength in respect to the average path loss. The simplest and a very popular way to model a noisy channel without fading is to use an additive white Gaussian noise (AWGN) channel, which is practically the model we have been using. The usual reason for the use of an AWGN channel is that the analysis is simpler than with fading channels. Two commonly used fast fading channel models are called the Ricean fading channel and the Rayleigh fading channel (Proakis 2001).

Fading is often present, especially in non-line-of-sight (NLOS) links, and, hence, its effects should not be ignored. Thus, verification should be obtained that BC-MAC’s performance does not collapse under fading, since in BC-MAC there are no special functions for tackling it. In this study, we will consider only slow fading, since it should effectively already bring enough of the desired variability to the link performance when considering the overall network performance. Accurate fast fading modeling would be considerably more difficult to implement to a network simulator.
Fading (fast fading, in particular) cannot always be considered a negative thing. For example, advanced receiver structures, like RAKE, are able to remarkably reduce the effect of fading. Moreover, the MIMO (Multiple Input, Multiple Output) principle (e.g., (Kunnari & Iinatti (2005)) intentionally exploits the multiple paths in the radio channel, enabling even higher performances in proper conditions than without fading. However, since BC-MAC does not assume any of these to be present, we will limit this study to the traditional receiver structure.

In slow fading, the received signal strength variation can be reasonably modeled with log-normal distribution. Therefore, the total path loss (in dB) will be a sum of the average path loss and the fading component $L_{\text{fad}}$, which is a zero-mean Gaussian random variable with standard deviation $\sigma_L$. Hence, the PDF of $L_{\text{fad}}$ is given by (Sounders 1999)

$$p(L_{\text{fad}}) = \frac{1}{\sigma_L \sqrt{2\pi}} e^{-\frac{L_{\text{fad}}}{2\sigma_L^2}}.$$  

(37)

The standard deviation $\sigma_L$, also called location variability, has typical values in the range of 5–12 dB (Sounders 1999). In this study, we will examine the effect of fading with a location variability of 5 dB and 10 dB. FPL will be used as the basic propagation model.

The effect of fading on network layer performance has been studied also, e.g., in (Takai, et al. 2001), where the packet delivery ratio and end-to-end delay as a function of different fading models (AWGN, Ricean, and Rayleigh) was investigated. In (Takai, et al. 2001), there was also a comparison between free space loss and two-ray loss, but the transmission powers were not set to comparable values, and, therefore, in the free space loss case the signal range was considerably longer, giving unfair conditions for comparison. Also (Mullen & Huang 2005) and (Han & Abu-Ghazaleh 2005) consider aspects related to ad hoc networking under a fading channel to be important.

The idea of this fading test case is not to perform a very detailed fading channel study, but more to scratch the surface and show that the operation of BC-MAC does not fail even under a serious fading channel. We will compare 802.11 and BC-MAC, but leave 802.11 RTS/CTS out of this study, since (Han & Abu-Ghazaleh 2005) already showed that the RTS/CTS mechanism does not bring anything especially beneficial in a fading channel environment but instead can be even considered harmful in some situations. As we have shown earlier, RTS/CTS
possesses a link status double checking feature, (Han & Abu-Ghazaleh 2005) shows that the effect of this harmful feature is enhanced in a fading environment.

The following table summarizes the simulation parameters used.

<p>| Table 20. Special simulation parameters of the fading test case. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, 802.11</td>
</tr>
<tr>
<td>Propagation model</td>
<td>FPL</td>
</tr>
<tr>
<td>Fading model</td>
<td>Slow log-normal fading with $\sigma_L$ of 0, 5, 10 dB</td>
</tr>
<tr>
<td>Area</td>
<td>500 m × 500 m</td>
</tr>
<tr>
<td>Mobility models</td>
<td>no mobility, MLV</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20 active</td>
</tr>
</tbody>
</table>

The effect of fading on the reception probability

Fading affects communications in many ways. The most important point is that fading causes variations in the received signal strength and, hence, uncertainty in the reception. Maybe the most illustrative way is to examine the packet loss ratio behavior as a function of radio link length. Let us carry out a simple test, where two BC-MAC nodes are communicating, and the other node is moving away from the other. No routing or retransmissions are used in this test, but the focus is only on the pure packet loss ratio. In addition to simulation, we are also able to calculate the theoretical packet loss behavior in this special case.

For the theoretical calculation, let us concentrate purely on packet loss, and simplify the assumptions in a way that packet loss will always happen if the received power level < detection power level threshold ($P_{\text{dth}} = -140.36$ dB). The probability that the fading component will exceed $z$ dB is given by (Sounders 1999)

$$p(L_{\text{fad}} > z) = \int_{L_{\text{dth}}}^{\infty} \frac{1}{\sigma_L \sqrt{2\pi}} e^{-\left(\frac{d_L}{2\sigma_L^2}\right)} dL_{\text{fad}} .$$  \hspace{1cm} (38)

For a given range, we get the average path loss with equation (36). With this, we can calculate the fading margin of a given range as

$$L(d_1)_{\text{in}} = P_{\text{dth}} - L(d_1)_{\text{FPL}} .$$  \hspace{1cm} (39)

Hence, a packet will be lost if the fading component exceeds $L_{\text{in}}$ in the given range. The fading component can also have negative values, meaning that the
received signal will be more powerful than expected on the basis of the average path loss. Hence, packets may also be received further away than the average effective radio range, giving nonzero average packet loss for ranges beyond $d_{rr}$. Therefore, the overall packet loss as a function of $d_l$ is given by

$$p_{\text{loss}}(d_l) = P(L_{\text{fad}} \geq L(d_l)_{\text{lin}}) = \int_{L_{\text{lin}}+L(d_l)_{\text{lin}}}^{\infty} \frac{1}{\sigma_L \sqrt{2\pi}} e^{-\frac{L_{\text{fad}}}{2\sigma_L^2}} dL_{\text{fad}}. \quad (40)$$

Fig. 79. The effect of fading on the packet loss ratio behavior as a function of link length.

Fig. 79 shows the results of this test with different intensities of fading. The simulated results are 1 s average results. As seen, when there is no fading the signal reception fails almost immediately after the $d_{rr}$. This is because of the powerful attenuation of the signal. The probability of reception failure increases rapidly as a function of link length despite the random nature of the bit errors. In addition, there are no interfering transmissions, which would add some random effect to the reception. Fading, on the other hand, clearly widens this transition region, and the bigger $\sigma_L$ is, the wider the transition region is. Thus, the sharp transmission range becomes blurred as the intensity of fading increases. This is easily seen in both the theoretically expected results and the simulated results, which, in fact, follow each other quite well. Also, as expected on the basis of the theory, the transition region widens in both directions: The signal can corrupt well before $d_{rr}$, but packets can also be received well beyond $d_{rr}$. For example, with $\sigma_L = 10$ dB,
packets can be received even well beyond distances of 300 m. It is clear that this kind of behavior will have effects on the overall network performance, since, e.g., many ad hoc network routing protocols are based on the idea that the transmission range is more or less constant.

Similar findings were made also in the beginning of paper (Mullen & Huang 2005), despite the different fading model. This shows that the phenomenon of fading itself is the most important in this kind of study. Naturally, if one is trying to model the accurate performance under a certain kind of environment, one should also consider choosing the accurate fading model for that environment.

**Results of a 20-node stationary network**

At first, a special scenario of stationary nodes is presented. In this scenario, we can see the effect of fading itself and are not confused by the extra characteristics brought by mobility. Certainly, this kind of situation does not correlate with the real world, since fading is actually caused by mobility, and it should not exist with stationary nodes if the environment is also more or less static. However, simulations are not restricted to the real world and give us the possibility to examine this interesting special situation.

The first observations naturally relate to the maximum radio range. By considering the maximum distances, where packets were received in this test, we conclude that fading has a considerable effect on the link distance. Packets were received even well beyond 350 m with 10 dB fading, while about 300 m maximum distances were observed for 5 dB fading (not shown). One could imagine that this would shorten the average route lengths, but this is obviously not the case, as seen in Fig. 80. In fact, more intense fading seems to extend the average route lengths. The reason for this is that fading makes the links less reliable. So despite the fact that packets can occasionally be received via very long links, even the shorter links become unreliable under fading. As seen in Fig. 79, the problems with 10 dB fading begin already with links of 150 m. In addition, interfering signals will go through a fading channel as well, and the effect of MAI will also be more variable. As a result, AODV tries to use shorter radio links and, thus, longer routes.

The effect of fading for making the links more unreliable is proven by Fig. 81, which presents the route breakage behavior. As seen, the intensity of fading directly affects the level of route breakages regardless of the setup. The breakages with 802.11 are constantly on a higher level than with BC-MAC, but the propor-
tional effect of fading seems to be about the same for both setups. An increased number of route breakages cause route re-establishments, increasing control overhead.

Fig. 80. Average route length behavior under different fading intensities with BC-MAC and 802.11.

Fig. 81. Route breakage behavior under different fading intensities with BC-MAC and 802.11.

As can be assumed, this will have a direct impact also on the packet loss ratio, which is shown in Fig. 82. The effect of fading on packet loss is similar as it is on route breakages: Packet loss ratios as a function of offered traffic load seem to
worsen steadily with increasing fading intensity. However, even with the high intensity fading of 10 dB, the effect on packet loss is not yet crucial even though clearly visible: BC-MAC’s traffic handling capability is weakened about by 50% from the non-fading case, while the minimum achievable packet loss of 802.11 is raised to the level of $10^{-4}$. As a result, a fading channel will weaken the performance but does not cause any special drop in the performance of BC-MAC. Thus, BC-MAC can be said to be fading safe. It is then up to the PHY structure to tackle the problems brought by fading.

Fig. 82. Packet loss behavior under different fading intensities with BC-MAC and 802.11 (stationary node case).

This straightforward test already showed that, if fading exists in the intended real environment, it should definitely not be ignored in ad hoc network studies. In a denser network or in a network with longer routes, the problems caused by fading are probably more severe. In a sparse network, fading can, in fact, help in connectivity problems, since packets can be received occasionally far beyond the nominal transmission range, but, on the other hand, fading will increase the overall unreliability of all the links. When considering the setups both, 802.11 and BC-MAC perform quite well even under strong fading.

170
Results of a 20-node mobile network

Next, the same basic scenario is used, but it is interesting to see whether mobility brings any new features to the observed performance. Also, this scenario mirrors better to real world. Let us check directly the packet loss ratio from Fig. 83, which presents non-fading and 10 dB fading cases. As seen, the overall trend of the performance is quite similar to what was shown earlier in the mobile network case (e.g., in Section 4.5.2) even though there was no fading. The effect of mobility, relative mobility in particular, is so intense that it clearly dominates the overall behavior. Fading also has a visible effect on the performance, but it is not dominating. Mainly, fading only worsens steadily the performance as compared to the non-fading case. However, an interesting exception to this is found as the packet loss of BC-MAC under 10 dB fading at a moderate traffic load seems to be even lower than without fading. The explanation for this peculiar behavior can be found in connectivity.

By examining the number of route discovery failures with different setups (not shown), one notices that under a moderate traffic load, clearly more route discoveries will fail under the AWGN channel than under the fading channel. In the AWGN channel, the radio range is quite constant, and, hence, the temporary connectivity problems easily lead to failed discoveries. There does not need to be a totally disconnected part in the network, but also weak connectivity cases can cause route discoveries to fail, since the RREQ messages in AODV are just flooded to the network, and link level retransmissions cannot be used. Hence, if the destination node is at some point reachable only via a single link, the failure of the whole discovery only requires that the forwarding of the RREQ over this particular link fails in that single shot. AODV, of course, tries the route search a couple of times before giving up (3 tries here), but the failure probability in a weak connectivity case can still be quite high. In a fading channel, on the other hand, the transmissions can be occasionally received via much longer ranges than $d_n$ would suggest. Thus, in a way, a fading channel can improve the connectivity. This is the reason why at some points one can get even better performance under a fading channel. The downsides caused by fading still dominate the overall performance, but this relationship between mobility, connectivity, and fading was an interesting finding.
Fig. 83. Packet loss behavior under different fading intensities with BC-MAC and 802.11 (mobile node case).

The delay presented in Fig. 84 also shows interesting behavior: Under a high traffic load, a fading channel causes extra delay, as could be expected, but under a low traffic load, delay under a fading channel is smaller than in an AWGN channel with both setups. This can be explained quite easily. By observing the average number of hops of the data packets in Fig. 85, one can easily notice that routes under fading and a low traffic load are on average clearly shorter. Roughly, more hops mean more transmissions and lead to greater delay. By comparing Fig. 84 and Fig. 85, it can be seen that the delay differences before the points of congestion between the studied cases follow the differences in average route lengths very well. The crossing points, where the delays between AWGN and fading channels are equal, also match crossing points of route lengths.
Fig. 84. Delay behavior under different fading intensities with BC-MAC and 802.11 (mobile node case).

Fig. 85. Route length behavior under different fading intensities with BC-MAC and 802.11 (mobile node case).
Even though the explanation for the delay behavior is easy, the reason for the behavior of the route lengths is more complicated, since in the stationary scenario, the situation was reversed. This can be explained by the route breakage behavior, for which the low traffic part is shown in Fig. 86. 802.11 under a fading channel constantly more route breakages than without fading. Route breakages, as discussed earlier in Chapter 4.5.2, cause AODV to find better routes, and as a result, the shortest routes are got with 802.11 under fading. However, this happens only in the mobile scenario, since because of the continuously changing topology, there are constant new opportunities for better routes, and the node density also increases towards the center of the area, making the links shorter (more reliable) on average. Extra route breakages caused by fading force AODV to find better routes. In a stationary network, in contrast, there will be no new opportunities for better routes, and, thus, since shorter links are more likely to hold longer under fading, the average routes will be longer. As a result, fading causes routes to be longer in a stationary network and shorter in a mobile network for 802.11. Naturally, this is a function of mobility, and there must then also be a level of mobility (somewhere between no mobility and MLV), where fading does not affect the route lengths.

![Route breakage count behavior under different fading intensities with BC-MAC and 802.11 under a low traffic load (mobile node case).](image_url)
With BC-MAC, the situation is quite different, and, in fact, it functions in quite the opposite way when compared to 802.11. As stated earlier, BC-MAC is efficient in keeping the routes functional. Hence, under fading channel, BC-MAC is able to keep up the long links, even beyond $d_{cr}$, when the nodes move. The route breakage behavior verifies this assumption, since clearly fewer breakages occur with BC-MAC under fading than without fading, while with 802.11 the behavior is reversed. Thus, routes with BC-MAC are considerably shorter under a fading channel than under an AWGN channel. However, from the delay point of view, this does not lead to as good result as letting the routing protocol find better routes: The delay with BC-MAC under fading is still worse than with 802.11 with and without fading (Fig. 84). The reason for this is that the long links are slow in the sense that they are unreliable and BC-MAC needs several retries to get data through.

In all the figures, there is also a special case of BC-MAC under a fading channel, where the reTx try limit is dropped to 7, as in 802.11. As seen, BC-MAC’s delay behavior is greatly enhanced with this modification, and, in fact, the lowest delays of all cases are reached with this setup. This is a direct result of the dropped reTx level (not shown). As could be expected, the number of route breakages has clearly increased, and, actually, BC-MAC under fading with a reTx try limit of 7 gives the highest number of route breakages. Thus, BC-MAC will notice the broken routes faster, but the drawback is that functional routes will also be detected to be broken with higher probability, leading to an increased packet loss, as seen in Fig. 83. Despite the increased packet loss ratio, the loss behavior is still clearly better than with 802.11, and the point of congestion remains unchanged. An interesting observation is that despite the highest number of route breakages with this setup, the routes are not much shorter than with BC-MAC with 20 reTx tries, as seen in Fig. 85. The reason for this is that BC-MAC is still able to build and keep up long routes, while 802.11 is forced to use shorter routes. The number of route breakages metric does not tell, e.g., whether the route repair functionality following the breakage was successful or not, i.e., was the route finally lost due to the breakage. In fact, by observing route repair behavior of BC-MAC (not shown), it can be observed that the success rate is considerably better than that of 802.11, and it has not been noticeably affected by the decrease of the reTx try limit. This is exactly as it should be, since with RREQs, retransmissions are not used anyway. The reason why the route breakage behavior of BC-MAC is worse than with 802.11 in the first place is simple: Since BC-MAC on average uses longer routes than 802.11, it is more likely that the routes consisting of more
hops will break up more easily. The longer routes do not really matter now because the average end-to-end delay is low anyway.

Overall, the dropped reTx limit greatly sped up BC-MAC’s ability to detect broken routes but also brought some uncertainty to the detection. However, BC-MAC’s ability to keep up longer routes did not weaken notably. The inclusion in this study of the setup of BC-MAC with a lower reTx try limit again clearly showed that by changing the limit, there is the possibility to almost directly trade between delay and packet loss. The tradeoff seems to work with and without fading even though the reasons for certain behaviors are slightly different depending on the channel. It would be interesting to study this dependency between mobility, the reTx try limit, and fading further, but this is already out of the focus of this work.

The main finding of the analysis in this section was that BC-MAC does not collapse even under a strong fading mobile channel, so from now on it is quite safe to assume that BC-MAC works in almost any kinds of channel conditions. At least the BCCA/BC-MAC method itself should not provide any limitations to operating under special channel conditions. Nevertheless, fading brings interesting features to the performance behavior and has considerable effects on the absolute performance. Thus, in studies aiming at accurate behavior, fading should not be ignored.

4.9 Ad hoc network performance under different traffic flows

The purpose of data networks is to carry data traffic. Thus, the effect of the traffic itself on the network behavior cannot be ignored. Traffic flows can have very different properties depending on the application, and these differences will affect the performance of the whole network. It might even be that some systems work perfectly under CBR traffic but fail under bursty modern data traffic. Thus, it should be verified that BC-MAC will also work with different traffic models and session models.

4.9.1 The effect of data packet size

Obviously, the data packet size will have effect on the performance. In particular, the performance of 802.11 RTS/CTS should be highly dependent on the packet size, since the whole idea of the handshaking mechanism is that the loss of a short RTS or CTS packet is much less severe than the loss of a possibly long data pack-
et. By now, an average sized data packet of 4,096 bits (512 B) has been used. In this section, we will test what happens to the performance if the data packet is very short or very long.

As a short data packet, a 64 B (512 bit) packet is chosen. With this small packet, the communication headers will be about the same size as the data portion, thus leading to inefficient operation. As a representative of a long data packet, a 1,450 B (11600 bit) packet is taken. This is about the same as the typically used maximum packet size in the modern Ethernet based internet, since the Ethernet protocol has the limitation that it can carry only 1500 B packets at maximum. Naturally, lower layer protocols could do the packet segmentation on the basis of their limitations, but link layer segmentation is practically hardly ever used, and the packets are segmented already at the higher layers in real networks. Since the performance of RTS/CTS is heavily dependent on the payload size, it is interesting to see whether 802.11 RTS/CTS with this long data packet will show significantly better performance and perhaps even challenge BC-MAC.

Only a stationary scenario will be considered. The following table summarizes the special parameters in this test case.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, 802.11 RTS/CTS, 802.11</td>
</tr>
<tr>
<td>Area</td>
<td>500 m × 500 m</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20 active</td>
</tr>
<tr>
<td>Mobility model(s)</td>
<td>no mobility</td>
</tr>
<tr>
<td>Traffic model</td>
<td>VBR-M (different packet sizes)</td>
</tr>
<tr>
<td>Interarrival times</td>
<td>exponentially distributed</td>
</tr>
<tr>
<td>Data packet sizes</td>
<td>512 bit, 4,096 bit, 11,600 bit</td>
</tr>
</tbody>
</table>

**Simulation results and analysis**

The most obvious thing affected by the data packet size is the control overhead. The actual control load is not significantly changed, but the relative one is, since the data portion size changes. Thus, with a smaller data packet, the use of the network is more inefficient. This can be easily seen in Fig. 87, which shows the normalized total network control load for BC-MAC with different packet sizes. The figure also shows theoretical approximations for 4,096-bit and 11,600-bit packets. The approximations are made with the assumption that the control load
for a particular packet size ($l_{\text{app}}$) can be approximated from the reference control load ($\Gamma_{\text{ref}}$) with the reference packet size $l_{\text{ref}}$ simply by scaling as

$$\Gamma_{\text{app}} = \Gamma_{\text{ref}} \frac{l_{\text{ref}}}{l_{\text{app}}}.$$  \hfill (41)

In this, we used the 512-bit packet case as the reference for the other cases.

As seen in Fig. 87, the approximation is quite accurate under a low and moderate traffic load. However, under a high traffic load, there is some inaccuracy. This is of course because the points of congestion are different with different sized packets. Because of the inefficiency, with a short data packet, the network starts to get congested earlier than with longer data packets. Thus, the approximations made on the basis of the control load behavior of a 512-bit packet overestimate the control load for longer data packets under a high traffic load. The figure shows very clearly the inefficient behavior of the network under a 512-bit packet: The steady state normalized control load of about 3 indicates that on average 3 control bits need to be transmitted in order for one data bit to get through. In contrast, the steady state control load with the longest data packet is only about 0.13.

![Fig. 87. Total network control load [bit/bit] behavior of BC-MAC with different packet sizes.](image)

In addition to control load, it can be also assumed that the application packet size has a direct effect on the end-to-end delay, since, naturally, it takes more time to
transmit a longer packet, and, thus, it experiences longer delay. Delay behavior is shown for BC-MAC in Fig. 88. In a single idealized link, the delay can be very accurately estimated on the basis of the packet size. Thus, a similar approximation as in the control load case can be tried also for delay. However, we must now take into consideration also the overhead bits, since delay is affected by all the transmitted bits and not only by the data portion. These approximated delay results for 4,096-bit and 11,600-bit packets are also shown in the delay figure. As seen, the approximation gives some idea about the delay level under a low and moderate traffic load, but the accuracy is much worse than in the control load case. Also, again, the high traffic load part approximation fails. Delay is a result of a complex function of several variables in the network, and, thus, a simple approximation like this does not function well in the ad hoc network case. High traffic load approximation could be perhaps enhanced if the point of congestion, or traffic load, was also tied to packet size difference.

Overall, there is nothing really surprising in the delay behavior. The points of congestion are clearly visible in the delay figure. It can be seen that with 11,600-bit and 4,096-bit packets, the network starts to get congested at about the same points, while in the short data packet case the congestion builds up much earlier.

**Fig. 88. Delay behavior of BC-MAC with different packet sizes.**
The closeness of the points of congestion between the longer packet sizes is also visible in packet loss Fig. 89. However, the difference to the 512-bit packet curve is already clear. This kind of behavior is of course understandable since the relative difference between 4,096-bit and 512-bit packets is much bigger than the difference between 11,600-bit and 4,096-bit packets. Nevertheless, even the curve with a very short packet is not yet alarming – the performance is still good. It seems that packet loss behavior vs. data packet length with BC-MAC is quite logical: Approximately, the curves just move to the left with decreasing packet size, while the shape remains more or less the same.

The increased loss of a small data packet is practically only caused by the decreased efficiency. This can be easily seen from Fig. 89, which also shows graphs of packet losses as a function of the MAC level offered traffic load. When plotting the graphs in this way, the effect of the transmission efficiency is normalized. Since the error correcting capability of the channel code is always 5% regardless of the packet size, the graphs give quite a good picture on the effect of the application packet size on the MAC layer behavior. As seen, the packet loss graphs with different packet sizes as a function of MAC level offered traffic are very similar to each other: Only a small difference is observed before the point of congestion, but the packet loss behavior after that is merged into a single curve. Thus, the packet size has practically no effect on the actual MAC layer point of congestion in BC-MAC, meaning that MAC layer performance is not very dependent on the packet size.

Fig. 89. Packet loss behavior of BC-MAC with different packet sizes.
Packet loss behavior with different data packet sizes for 802.11 with and without RTS/CTS is shown in Fig. 90 (a). With a quick glance we already see that the different data packet sizes do not change the performance of 802.11 even nearly as much as would be needed to compete with BC-MAC. With 802.11 RTS/CTS, the performance behavior is quite logical, as with BC-MAC: With a longer data packet, the performance becomes better because of the more efficient use of the network. However, while with BC-MAC the performance curves were practically just shifted with a different sized packet, with 802.11 RTS/CTS the low traffic load performance remains the same with all packet sizes, and differences start to appear with increasing the traffic load. Again, the difference between 11,600-bit and 4,096-bit packets is not that big, but with a 512-bit packet the performance is already very poor. It should be noted that 802.11 RTS/CTS cannot reach packet losses below $10^{-3}$ in this scenario even with the long 11,600-bit data packet, proving that packet size is not the reason for the bad behavior.

802.11 without RTS/CTS performs in quite a different way. Under a high traffic load, the longer data packet clearly results in better performance, but under a low and moderate traffic load, the situation is the other way around: A longer packet results in worse performance. The exact reason for this special behavior might be difficult to find out and is not in the focus of this thesis, but we can speculate on this briefly. In 802.11 RTS/CTS, CA is effective and collisions are rare. Thus, after a successful handshake, the data packets will be most likely transmitted successfully. And since longer data packets are more efficient from the network resource usage point of view, longer packets will then result in better performance. In BC-MAC, collisions have not that much meaning, and the only thing that remains is the efficiency of the transmission, and then longer packets will again result in better performance. 802.11, on the other hand, is very vulnerable to collisions, and if RTS/CTS is not used, collisions happen more often. By also taking into account that longer packets lead to longer transmissions, it is obvious that the possible collision period is also longer, and, thus, collisions are more likely with longer data packets (this is verified by the simulated results, but not shown here). Thus, with long packets, packet loss ratio is bigger than with short packets. Under a high traffic load, however, the efficiency becomes more dominating than the collisions, and, thus, the network becomes congested earlier with a smaller data packet.
Fig. 90. Packet loss behavior of 802.11 with and without RTS/CTS with different packet sizes (as a function of the offered application data traffic load in (a), and as a function of the offered MAC level load in (b)).
Fig. 90 (b) shows the packet loss ratios of 802.11 as a function of the offered MAC level load. As seen, with 802.11 RTS/CTS the packet loss behavior with different packet sizes is in fact quite similar. Thus, slightly surprisingly, 802.11 RTS/CTS also, as BC-MAC, seems to be quite insensitive to packet size when only MAC level traffic is considered. This is a positive feature in a MAC protocol, since it means the protocol is fair to different sized packets. However, in the case of pure 802.11, the situation is worse. The packet loss behavior of 802.11 in Fig. 90 (b) is somewhat similar to Fig. 90 (a). There are clear differences in packet loss with different packet sizes, and, also, there is again a point where the different packet loss curves cross each other. After this crossing point, the loss curves again become separated, and a convergence of curves similar to BC-MAC is not seen. Of course, it should be noted that due to the logarithmic scale, the differences in packet loss behavior with different packet sizes under a low traffic load with 802.11 RTS/CTS are bigger than those of 802.11. In addition, also with 802.11 RTS/CTS, the different curves cross each other, but the effect is less eye-catching than it is with 802.11. Nevertheless, under a high traffic load, 802.11 RTS/CTS behaves clearly steadier than 802.11 also in the absolute sense.

Overall, the results of testing different packet sizes were quite predictable. Only the packet loss results of 802.11 gave something to consider, but this does not give rise for any deeper study. Packet size has obvious effects on the performance, since the shorter the packet, the more inefficient the transmission is, leading to worse performance. However, none of the setups gave any special indication of particularly bad or good performance under any packet size. We tested a very short data packet and a long data packet in addition to the typical 4,096-bit packet, and got limits to the performances, while BC-MAC maintained its position. Thus, there is no point in testing out other packet sizes in this work.

When the performance was examined as a function of MAC level traffic, an interesting finding was that BC-MAC is very robust in terms of data packet size differences. This indicates that BC-MAC treats different sized packets equally. This is most likely thanks to the intelligible deferment period calculation of BC-MAC, which is tied to the packet size (see equation 9). 802.11 RTS/CTS also had reasonably steady behavior with different sized packets under a high traffic load. However, the performance of pure 802.11 seemed to be heavily dependent on the packet size. This is most likely caused by the channel access and MAC functionalities of 802.11, where the deferment period and back-off calculations are not dependent on the packet size. Thus, the channel access waiting times might be too long for short data packets causing inefficient operation. These kinds of conclu-
sions were also drawn in (Koivula 2006), where 802.11 based ad hoc network performance was studied with different sized VoIP (Voice Over IP) streams. Then again, for long data packets, the waiting times might be too short.

4.9.2 The effect of traffic burstiness

In this section, we will study the effects of different IAT processes. So far, we have been using a VRB-M type source traffic model. It is a good model for several situations and can thus be used as a typical traffic source. However, there are several applications which produce, e.g., CBR traffic flows, where both the packet size and IAT are constant. Also, several ad hoc network studies are carried out with a CBR traffic model, so CBR is included in the study. Typically, CBR traffic is easier for the network to handle than VBR (see, e.g., (Prokkola, et al. 2003)), but CBR can also cause some interesting artifacts to the results (see, e.g., (Prokkola, et al. 2007)). Modern data traffic is usually very bursty in nature, and the source traffic models then follow long or heavy tailed distributions. As a representative of a bursty source traffic model, a model where the interarrival times follow Pareto distribution (31) is used. The burstiness of Pareto distribution can be directly affected by modifying the $\alpha$ parameter, since the smaller $\alpha$ is, the longer the tail of the distribution is, and the more bursty the generated traffic is. This provides a simple way to examine the effect of the level of burstiness. For the burstier case, $\alpha = 1.6$ (VBR-P16) is set, and for a point of comparison, $\alpha = 2.0$ (VBR-P20) is set. With $\alpha = 1.6$, Pareto distribution is heavy tailed with infinite variance, while $\alpha = 2.0$ is the limit for the Pareto distribution being more or less traditional with finite variance. We have studied the effect of different traffic models earlier in (Leppänen, et al. 2003) and (Prokkola et al. 2004), but these studies were made with simplified simulation models, and BC-MAC was not included. In this study, we include for BC-MAC also a very bursty case, where $\alpha = 1.2$ (VBR-P12), to really bring out the effects of burstiness.

As the packet size test showed, the performance of 802.11 RTS/CTS did not show great improvements even with a very long data packet, although it is ideal for the use of the RTS/CTS mechanism. Thus, there is no need to include RTS/CTS to this test case. In addition to different interarrival distributions, an interesting test is carried out with VBR-M type traffic, where the data packet length will be exponentially distributed instead of being constant as so far. With the exponential packet length distribution, the maximum frame limit of 1,450 B is also set. However, the data packets exceeding this limit are not cut, but split
above the network layer, thus simulating fragmentation. The following table summarizes the parameters, etc used.

**Table 22. Special simulation parameters of the 20-node traffic burstiness test case.**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, 802.11</td>
</tr>
<tr>
<td>Area</td>
<td>500 m x 500 m</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20 active</td>
</tr>
<tr>
<td>Mobility model(s)</td>
<td>no mobility</td>
</tr>
<tr>
<td>Traffic model</td>
<td>CBR, VBR-M, VBR-P20, VBR-P16, VBR-P12</td>
</tr>
<tr>
<td>Interarrival times</td>
<td>constant, exponentially distributed, Pareto distributed</td>
</tr>
<tr>
<td>Data packet sizes</td>
<td>constant 4,096-bit, exponential (with 4,096-bit mean)</td>
</tr>
</tbody>
</table>

**Simulation results and analysis**

The results of BC-MAC will be examined first. Even a quick glance at the simulation output reveals that the results are quite surprising: Traffic processes seem to have only minor effects on the BC-MAC’s performance. This is seen, e.g., in the packet loss ratio behavior depicted in Fig. 91. Closer examination reveals that if, e.g., $10^{-3}$ is taken as a limit for acceptable performance, with VBR-P12 type traffic, $G = 0.40$ is reached, followed by VBR-P16 with $G = 0.41$, and VBR-P20 with $G = 0.42$. With CBR and VBR-M, loads of about 0.45 and 0.46, respectively, are reached. Thus, as could be expected, burstier traffic leads to worse performance, but the differences are very small. Bursty traffic occasionally has high traffic intensities, far beyond the average, and also long periods of time with no traffic. The long idle periods can ease congestions in some situations, but typically the high intensity periods dominate the average behavior and also cause uncertainty to the network behavior. At the low packet loss ratio levels ($< 10^{-4}$), there are some clearer differences between different traffic processes. It can be seen, for example, that CBR seems to give about the best performance, but the undulation due to the simulation accuracy with these low loss levels is already so high that nothing exact can be said.

The most obvious differences between traffic processes are seen in jitter, which is presented in Fig. 92. As seen, clearly the highest delay with a low and moderate traffic load is got with VBR-M, where packets follow exponential distribution. This is quite obvious, since different sized packets affect the delays and, thus, jitters directly. With a constant packet size, only the queuing delays have an
effect. Under a low traffic load, the queuing delays are minimal, and, thus, real
differences in jitter are seen only under a higher traffic load. As could be ex-
pected, CBR gives the best performance, because there are no bursts which could
create extra queuing delays. It is, however, possible that CBR will sometimes
give also worse performance, as seen in the packet loss ratio under a higher traffic
load. This is because in the tested CBR case the network is very static. The start
times of the CBR traffic flows are random, so if two neighboring nodes start gen-
erating the flows in such way that the packets will collide, then the generated
packets will always collide during the whole simulation iteration, leading to extra
retransmissions and delays. Then again, it is likely that most of the time, under a
low traffic load, flows will be generated in such way that there is not even a need
for deferment periods, and therefore the operation is TDMA-like.

Fig. 91. Packet loss behavior of BC-MAC with different traffic processes.

It is slightly surprising that VBR-M leads to worse jitter performance under a low
traffic load than VBR-P cases. This is likely caused by the long idle periods of the
burstier traffic models, easing congestions under a low traffic load. However, with
an increasing traffic load, the performance behaves as could be expected, i.e.,
burstier traffic congests the network earlier. Nonetheless, it should be recalled that
the differences between different traffic processes (excluding the exponential
packet distribution case) are practically negligible even in the jitter case. In the
delay behavior (not shown), the differences between all traffic processes are very small.

From the simulation technical point of view, highly bursty traffic models are tricky. If we consider, e.g., the used Pareto distribution, it loses its moments with decreasing $\alpha$ (see eq. 32). Finally, even the expectation value becomes infinite when $\alpha < 1$. Even though this “transition” from finite to infinite expectation value is theoretically strict, the real average value behavior is not. Instead, the average values start to become more and more unstable when $\alpha \to 1$. In the simulations, this is seen already with $\alpha = 1.6$, as the number of simulated packets varies already clearly more than with $\alpha = 2.0$ or VBR-M. With $\alpha = 1.2$, the situation becomes even problematic, since the variation is so intense that the intended mean offered traffic loads no longer hold. In order to get to the mean values hold, longer simulation times would be needed. However, due to limited simulation time, it was decided to use sample averages instead of the intended mean offered traffic load values, giving reasonable results. In fact, sample averages would be the only option if one wished to study the interesting situation with $\alpha < 1$.

![Fig. 92. Jitter behavior of BC-MAC with different traffic processes.](Image)

The performance behavior of BC-MAC was quite straightforward, so we can now turn our attention to 802.11. Packet loss behavior for 802.11 is shown in Fig. 93 with different traffic processes. An immediate observation is that the nature of the traffic process clearly affects the performance of 802.11 more than BC-MAC. The
biggest differences occur under a low traffic load, while with an increasing load, the differences narrow, and finally under a high traffic load, even the ranking between different traffic models can change. As could be assumed, the best performance under a low and moderate traffic load is got with CBR traffic. The difference, e.g., between CBR and VRB-M is notable.

This is a nice example of how important the traffic process is also in a networking performance study – one can get too optimistic results by just examining the performance with simplified models (CBR in this case). Interestingly, VBR-M with exponential packet length distribution gives clearly the worst performance under low and moderate traffic loads. This is mostly related to 802.11’s bad performance with long data packets, which was proven in the previous section, since one gets occasional long data packets with exponential distribution. This can be seen from the collision behavior in Fig. 94 as the exponential distribution clearly leads to a larger number of collisions. In addition, varying packet size seems to be harder to handle for 802.11 than constant long packet size (compare the packet losses in Fig. 93 and Fig. 90 (a)).

Fig. 93. Packet loss behavior of 802.11 with different traffic processes.

Overall, the packet loss behavior of 802.11 follows the collision behavior very well. Even though the collision figure is calculated from the successful packets, it also gives an idea about the total number of collisions (i.e., including collisions
which have caused packet losses). The collision figure also shows that a high CBR traffic load causes more collisions than the other traffic types. This is because of the CBR’s static properties explained earlier, which cause performance deterioration (seen in packet loss, Fig. 93) under a high traffic load.

As regards burstiness, it can be concluded from packet loss behavior that increasing burstiness from VBR-M to VBR-P16 does not affect the performance much. VBR-P16 leads to worse performance than VBR-P20, which is worse than VBR-M, but the differences are almost negligible.

![Collisions behavior of 802.11 with different traffic processes.](figure)

**Fig. 94. Collision behavior of 802.11 with different traffic processes.**

Overall, this test case of different traffic processes gave quite nice results: BC-MAC is almost immune to the level of traffic burstiness. Burstiness causes temporary high traffic loads and connives in building up temporal and spatial congestions. However, BC-MAC is by its nature quite good at handling different traffic loads, and thus the effect of burstiness is not shown until a high traffic load, and even then the differences between traffic processes are almost negligible. There were no major differences between any of the studied traffic processes, but only the exponential packet size distribution case caused a clearly visible effect in jitter. 802.11, on the other hand, is known to be vulnerable to congestions and collisions, and, thus, the performance is affected clearly more by the level of burstiness. More specifically, having basic variability (VBR-M) instead of CBR results in a clearly worse performance, but increasing burstiness beyond VBR-M no
longer practically worsens the performance. Moreover, packet length distribution has an evident effect on the packet loss performance of 802.11: 802.11 clearly has trouble handling variable sized packets.

This part of the study showed several interesting findings in the performance behaviors. Most of the results are straightforward and easily explained, but there were also some phenomena in the behavior of 802.11 performance that cannot be explained thoroughly without going into more details. Also, it would be interesting to continue this traffic process study further, but unfortunately it must be recalled that the major points regarding this work have been already achieved, and, thus, there is no need to go deeper into this topic.

4.9.3 The effect of session models

In addition to the source traffic models studied in the last two sections, the session model also has a part in describing the application behavior. Session model here means the way in which the nodes actually communicate with each other. So far, the model of dynamic connections has been used, but next we will evaluate how, e.g., static connections, which have been used in several ad hoc network studies, affect the performance. In the SC model, the connections are randomly chosen in the beginning of the simulation iteration, as in the DC model, but the difference is that the connections do not change. Thus, this kind of model can give quite an optimistic view of the performance, at least when considering routing overhead, since the routes can be unchanged during the whole iterations, excluding the few possible route breakages occurring from congestions, etc.

DC and SC models are both flat random models, where each node can be chosen as the traffic destination with equal probability. This should result in a somewhat balanced traffic distribution in the network (of course, the traffic load is always higher in the center of network because there will most likely be more routing traffic than in the edge areas of the network). However, what if some node in the network had a connection to the fixed network and would, therefore, be more likely to be the destination for the traffic? Would this ruin the performance of BC-MAC, since the main benefits of BCCA are gained when there are several simultaneous active transmissions in a small area considering several nodes? This kind of fixed network connection (FNC) model and its effects on the performance will be tested in this study. Again, the default 20-node scenario is used, so in the FNC case, one of the 20 nodes is assumed to have a fixed network connection.
The FNC model functions as follows: E.g., in the FNC-50 model, the other nodes first have a 50% probability of choosing the preselected FNC node as their destination, and if the FNC node is not chosen, the destination is chosen randomly among all the nodes (including the FNC node). The session lengths are similar to dynamic connections, following exponential distribution with a mean value of 100 packets. The FNC node always follows the DC model, thus modeling the traffic from the fixed network. As an extreme case we take FNC-100, where all the traffic of the pure ad hoc nodes is always sent to the FNC node.

Since the purpose is to extract the effect of session models only, mobility is not included in this test case. The following table summarizes the special parameters used.

Table 23. Special simulation parameters of the 20-node session model test case.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, 802.11</td>
</tr>
<tr>
<td>Area</td>
<td>500 m × 500 m</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20 active</td>
</tr>
<tr>
<td>Mobility model(s)</td>
<td>no mobility</td>
</tr>
<tr>
<td>Traffic model</td>
<td>VBR-M</td>
</tr>
<tr>
<td>Session models</td>
<td>DC, SC, FCN-100, FCN-50</td>
</tr>
</tbody>
</table>

Simulation results and analysis

We can go directly to the packet loss ratio, which for BC-MAC is shown in Fig. 95. As seen, the FNC model shifts the point of congestion, as could be expected. FNC-100 gives worse performance than FNC-50 and DC, since all the traffic is sent to a single node, which will eventually become a bottleneck instead of the traffic being distributed among all the nodes in the network. Thus, even though routing would be mostly successful, the final links to the destination will fail with an increasing traffic load. FNC-50 also seems to get congested slightly earlier than DC, but the difference is not that significant. By examining the average route lengths (not shown), it can be noticed that session models do not seem to affect to the route lengths, which indicates that the traffic is still quite well distributed in the network even though, e.g., in FNC-100, all the nodes send traffic to a single server node. This can be assumed also on the basis of packet loss behavior, since even under FNC-100, BC-MAC’s performance is still very good.
In contrast to the expected behavior, one surprising thing is noticed in Fig. 95: SC leads to worse performance than DC, despite the quite opposite assumptions. The routing control load behavior (not shown) is of course a lot worse under DC than it is under SC, but the packet loss ratio under a low traffic load remains at a higher level under SC than under DC, and the point of congestion is shifted. This interesting behavior needs further investigation. First, we can take a look at the results of individual simulation iterations to see how they are distributed (not shown because of the large amount of material). It is noticed that while the packet loss seems typically to be similar or less than with DC, there are some scenarios where packet loss is clearly higher than in other scenarios, thus dominating the average behavior. With $G = 0.003$, for example, there is a single iteration with a packet loss ratio of $3.51 \times 10^{-4}$, while the average of all the other iterations under this load is as low as $7.218 \times 10^{-6}$. In this particular iteration, there are also some failed route discoveries, which typically do not exist with a traffic load this low. The same kinds of findings are discovered also with other values of $G$. Thus, we need to track one of these special iterations to see what is happening.

![Packet loss behavior of BC-MAC with different session models.](image)

We will take under examination the mentioned special iteration with $G = 0.003$. Tracking more accurate behavior is not straightforward, since only average values are calculated in the simulations, so the accurate iteration behavior is no longer available. However, with OPNET, the random events are ignited by setting ran-
dom generator seed values. As a result, we are able to repeat and track single simulation iterations and then study the accurate behavior. The special iteration was found by repeating the simulation set with $G = 0.003$, while examining which iteration caused the higher packet loss ratio. Since the scenario is stationary, the most important thing is the simulated scenario topology. For the tracked special case, this is shown in Fig. 96. For more information, the question of which connections will take place in the iteration was also examined. Thus, the connections are also drawn in the figure in the form source node $\rightarrow$ destination node.

Fig. 96. A random scenario, which gave clearly worse results than on average ($G = 0.003$).

It can be easily seen from Fig. 96 that this is an obvious weak connectivity topology. Clearly, nodes have been distributed to clusters with only some links between them. In addition, node 19 is clearly separated from other nodes, and it has only very long and unreliable connections to nodes 14 and 17, while nodes 13 and 16 are already practically too far away from node 19 to have permanent links.
Node 5 has a session to node 19, while node 19 also has a session to node 5, forming a bi-directional session. Considering the unreliable links to node 19, and the fact that traffic now flows both ways in the single route between nodes 19 and 5, it is not a surprise that packets will be lost, leading to increased average packet loss behavior in this particular iteration.

Similar findings are also observed in the other special iterations. Thus, what was learned here is that even though the SC session model is easier for the routing protocol to handle, average performance is still typically worse than under DC, mainly because of the special topologies. Weak connectivity topologies will also happen under DC with equal probability to SC, but the possible difficult sessions are not permanent, while all the sessions are permanent under SC. What can also happen under SC is that a single node is a destination of multiple traffic flows, inevitably causing the communications to be more unreliable (e.g., in Fig. 96, nodes 9, 16, and 6 all share the same destination node 7). These situations occur also under DC, but they are only temporary, distributing the traffic more equally to the network in the long run.

The packet loss performance of 802.11 with different session models is shown in Fig. 97. Similar kind of behavior is observed as with BC-MAC. SC gives worse performance than DC and also worse than FNC-100 under a low traffic load. With an increasing traffic load, FNC-100 crosses SC, and gives the worst performance under a high traffic load. FNC-50 gives a packet loss of something between FNC-100 and DC. However, the difference between FNC-100 and DC is very small, and under a very low traffic load, packet loss under FNC-100 is even lower than under DC, unlike with BC-MAC. This is intelligible, since 802.11 is meant for centralized networks, and under FCN-100 the network part around the server node will behave more in a centralized fashion. This favors the operation of 802.11, while the server node links will of course still be easily congested, as with BC-MAC, making the performance under FCN-100 slightly worse than it is under DC under a higher traffic load. Having centralized network parts is not favorable to the BCCA method. Thus, the gap between FNC and DC models is wider with BC-MAC than with 802.11. However, even under FNC-100, BC-MAC’s performance is clearly better than that of 802.11.

Nothing particularly interesting not already seen in packet loss is seen in the other statistics, so there is no point in analyzing them. As a result of this session model study, it can be stated that session models do have clear effects on the performance, as expected. However, the tested models did not cause any thoroughly significant behaviors. A bit surprisingly, the SC model seems to be harder for the
network to handle than DC (with both, 802.11 and BC-MAC) at least in this scenario. However, it should be noted that SC is mostly easier for the network to handle than DC, but the average behavior is dominated by a few difficult iterations. FNC, which unbalances the flat ad hoc network behavior, typically leads to worse performance than DC. However, 802.11 can also benefit from the partially centralized network behavior of FNC, and thus the packet loss performance under FNC is close to that of DC under a high traffic load, and even better under a low traffic load. BC-MAC gives a steadily worse performance under FNC models than under DC, which is intelligible, since the benefits of the R-code protocol are weakened when centralized behavior is introduced. Nevertheless, this difference between FCN and DC is quite small, and overall BC-MAC is able to handle also the most difficult FNC-100 case very well, and the performance gap to 802.11 remains clear. This is an indication that our earlier claim holds: The multi-hop ad hoc network structure tends to result in a distributed traffic pattern even though the session model would have a centralized nature.

![Fig. 97. Packet loss behavior of 802.11 with different session models.](image-url)
4.9.4 Ad hoc network performance under VoIP traffic

Finally, we will test how the different setups perform under a more realistic real-time application. More specifically, this study will test how the different setups handle VoIP traffic in the default 20-node scenario with and without mobility. Since this is a simulation study, a real application is not used, but the behavior of the application will be modeled as closely as possible. SJPhone by SJ Labs is taken as the modeled application (SJ Labs 2006). We have been using this VoIP application in our recent measurement studies (e.g., in (Prokkola, et al. 2007)) and are thus aware of its properties. By default, the tool generates a CBR traffic stream with a VoIP packet IAT of about 20 ms. The actual VoIP packet size is 33 B. However, in addition to this, before the network layer there is also a UDP header (8 B) and an RTP (Real-time Transport Protocol) header (12 B), and thus the total payload packet size is 53 B (424 bit).

Due to the small packet, the data delivery is inefficient, and the network will probably suffocate to the traffic quite early with an increasing traffic load. One can also wonder about the efficiency of VoIP: 33 B of data vs. 40 B (IP header 20 B) of control information already at the network level. Moreover, if we take into account the MAC and PLCP overhead, we get 33 B vs. 91 B in a typical wireless network case, giving the control load share to be ~ 73% of the total traffic flow. In addition, this calculation did not take into account the ACK’s, retransmissions or multiplication of the traffic in a multi-hop environment.

The number of supported VoIP sessions is the most interesting outcome of this study, and, thus, the results will be drawn as a function of active VoIP sessions. However, mapping the number of connections to traffic loads is also quite easy. For application (transport layer), there are 424-bit packets arriving at rate of 50 pkts/s, leading thus to bit rate of 21,200 bit/s per VoIP session. When taking into account that the traffic flows in both directions and the physical layer data rate, which still is 1 Mbit/s, we get that one active VoIP connection corresponds to the $G$ of 0.0424. Thus, the maximum 10 connections cause $G$ of 0.424. This does not sound much, but this scenario is definitely very challenging first because of the high overhead due to a small data packet, and, second, fast traffic always flows in both directions for each connection, while neither 802.11 nor BC-MAC has the means to specially deal with two-way traffic, since there is no duplexing at the physical layer.

The next table summarizes the most important parameters of this scenario.
Table 24. Special simulation parameters of the VoIP test case.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studied setups</td>
<td>BC-MAC, 802.11</td>
</tr>
<tr>
<td>Area</td>
<td>500 m × 500 m</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20 active</td>
</tr>
<tr>
<td>Mobility model(s)</td>
<td>no mobility, MLV</td>
</tr>
<tr>
<td>Traffic model</td>
<td>SJPhone VoIP</td>
</tr>
<tr>
<td>Session models</td>
<td>VoIP sessions between node pairs (from 1 to 10 pairs)</td>
</tr>
</tbody>
</table>

Before going into the simulation results, let us introduce a special performance metric for this special scenario. Voice quality is often evaluated with subjective quality, which means how real users experience the quality. For subjective quality, real user tests need to be performed, and the quality is typically addressed with a mean opinion score (MOS). A typical way is to categorize MOS values grading from 1 to 5 according to the following table (ITU-T 1996):

Table 25. MOS values defined by ITU.

<table>
<thead>
<tr>
<th>MOS</th>
<th>Quality</th>
<th>Description of impairment</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Excellent</td>
<td>Imperceptible</td>
</tr>
<tr>
<td>4</td>
<td>Good</td>
<td>Perceptible but not annoying</td>
</tr>
<tr>
<td>3</td>
<td>Fair</td>
<td>Slightly annoying</td>
</tr>
<tr>
<td>2</td>
<td>Poor</td>
<td>Annoying</td>
</tr>
<tr>
<td>1</td>
<td>Bad</td>
<td>Very annoying</td>
</tr>
</tbody>
</table>

The values in the table are absolute values, but if we want to describe MOS results as continual values, we could think that, e.g., fair is from MOS of 2.5 to 3.5, etc. Overall, it can be said that value 1 (to 1.5) is unusable. Also, MOS of 2 (1.5 to 2.5) is so bad that the user would not stand that kind of quality for a longer time. From MOS of 3 upwards (> 2.5), the quality is at a practical level where the users could actually consider using that kind of system. The methods to perform quality tests and how to interpret the results are described in (ITU-T 1996).

MOS values represent subjective QoS statistics, but we are measuring objective QoS statistics in this work. However, there is a relation between these two, since, obviously, the worse the objective quality is, the worse also the subjective one is. The mapping between these two is the tricky part, and it is dependent on a lot of things (e.g., codec and packet loss profiles). There has been a lot of research on this (e.g., (Mohamed 2003), (ITU-T P.862 2001)), and good results have been got. In this work, we will be using pseudo-subjective quality assessment (PSQA)
described in (Varela 2005), which provides also a good review around the topic of subjective and objective QoS for VoIP. The functions developed and tuned in (Varela 2005) give estimates for the listening MOS on the basis of the measured objective metrics. In this work, we use the simulated performance values as an input to the algorithm to get estimates for MOS.

It must be recalled that the acquired MOS values will not be 100% accurate, since the PSQA functions have not been tuned especially for the performance behavior of this ad hoc network case of ours. However, the error should not be significant, and, thus, we will get a good overall view of the possible subjective performance.

Simulation results and analysis

We start the analysis from the packet loss ratio, which is presented in Fig. 98. Immediately it is seen that 802.11 has serious trouble in handling several VoIP sessions: The packet loss ratio is at a moderate level only with a single active VoIP session and increases already with two sessions well above the 3% loss level, which is considered as a high loss ratio for conversational voice traffic. This is not a surprise, since if we just think of the traffic loads, a single VoIP session causes a $G$ of 0.0424, which is already quite troublesome for 802.11 with such a small data packet size. This can be easily seen in observations made in section 4.9.1, where different packet sizes were tested. For instance, consider Fig. 90 (a) and the smallest packet of 512 bits, which is near to the packet size in this VoIP test: The loss ratio increases very quickly as a function of $G$, similarly as now observed in Fig. 98.

The difficulty of carrying VoIP-like traffic is also seen in the case of BC-MAC, since its loss ratio also increases quickly as a function of VoIP sessions. However, BC-MAC is able to uphold considerably more VoIP sessions with the same packet loss ratio limits. The 3% loss level is not exceeded until 5 VoIP sessions in both the mobile and stationary scenarios. This correlates also quite nicely to the work in Section 4.9.1. As was seen in Fig. 89, the troubles with a 512-bit packet began after $G = 0.2$, while 5 VoIP sessions equals to $G = 0.212$. This indicates that the effect of packet size dominates the performance, but the two-way traffic flow does not seem to have as much meaning as we feared. Of course, exact conclusions cannot be drawn just based on this observation, but if the effects of two-way traffic flows had been severe as compared to the one-way traffic, they should have been visible already. It should also be recalled that in the DC session
model with every node involved, the traffic will practically flow in both directions in most of the links due to routing, and the two-way data sessions are of course also possible. Thus, in order to really see the effect of two-way traffic as compared to pure one-way traffic, a special scenario would be needed. However, this is not necessary, since for this work, it only matters that the performance of BC-MAC does not collapse under pure two-way traffic, and it does not.

Fig. 98. VoIP packet loss ratio as a function of active VoIP sessions.

When comparing the differences between mobile and stationary scenarios with BC-MAC, a typical behavior is seen, as the loss ratio under MLV saturates about to the same level as was seen in Section 4.5.2 in Fig. 52, where the effects of mobility were studied in more detail. Also, mobility obviously brings challenges, and thus the loss rate is continuously somewhat higher than in the stationary scenario. However, with 802.11, a very different observation is made: The loss rate with mobility is in fact clearly better excluding the single VoIP session case. At first, this sounds peculiar, but there is a good explanation for this, and similar observations could actually have been made earlier. By now, we have been mainly considering data transfer applications, and, thus, the interesting packet loss ratio region has been $< 10^{-2}$, and we have not paid much attention to differences under high loss ratios. However, if we take a closer look, e.g., to Fig. 38 and Fig. 52 and
compare the packet loss ratios of 802.11 in them, we can observe the same thing as in this VoIP test: 802.11 in a mobile scenario performs better than in a stationary scenario under a high traffic load. This was actually noticed in the throughput behavior in Section 4.5.2, as the mobile scenario led to better maximum throughput as compared to the stationary scenario. This, of course, is caused by the properties of the RWM, as was discussed: 802.11 in the RWM MLV scenario enjoys clearly shorter routes than in a stationary scenario. In this VoIP test, 802.11 with RWM has average route lengths of about 1.5 or 1.6, decreasing even to 1.3 under several VoIP sessions, while all the other cases have route lengths of about 1.8 and decreasing to 1.6 with an increasing number of VoIP sessions (not shown). However, since the negative effects brought by mobility dominate under a low and moderate traffic load, we did not pay much attention to this before. In this VoIP test, on the other hand, the operational region is from a moderate to high traffic load, and the positive effects on the packet loss ratio caused by mobility are visible.

Fig. 99 presents the most interesting performance metric of this test – estimated VoIP MOS. As seen, MOS has clear correlation to packet loss, and, thus, BC-MAC clearly outperforms 802.11. BC-MAC without mobility is able to reach an MOS of about 4.1, which is about the maximum of the assumed codec (GSM codec with 13,200 bit/s). After 3 sessions, MOS starts to decrease quite steadily, but a fair voice quality can still be maintained even with 7 simultaneous VoIP sessions, despite the fact that packet loss is already > 10% at this point. With mobility, BC-MAC performs worse, but 5 VoIP sessions can still be supported with fair quality. It can be also noticed that even though BC-MAC with mobility can reach good quality with a small number of sessions, it cannot reach to MOS > 4 because of the basic packet loss level of $10^{-3}$.

When considering 802.11, good quality can only be achieved with only a single active VoIP session. Fair quality is achievable with 2 VoIP sessions, but when there are 3 or more simultaneous sessions, the quality is already poor. Also, similar behavior is seen as in the packet loss: 802.11 with mobility performs mostly better than without mobility. The quality with mobility decreases fast to the level of MOS = 2, but after this the slope is very gentle, and, finally, it even seems to cross the quality of BC-MAC (mobile scenario). However, this does not really matter, since after MOS goes below 2.5, the quality is already too bad for actual usage.
Fig. 99. Estimated VoIP MOS (listening) as a function of simultaneous active VoIP sessions.

Next, we consider the end-to-end delay shown in Fig. 100. As seen, this tells quite the same story with packet loss ratio and MOS: 802.11 has serious trouble handling VoIP sessions. For conversational applications, 400 ms is considered high delay, hindering the conversation notably. The stationary 802.11 scenario exceeds this delay level already with 3 VoIP sessions, while with 2 VoIP sessions, the delay is already over 100 ms, which is near to the 150 ms limit of moderate delay. It must be also recalled that these are average delays, so the instantaneous delays can occasionally be considerably higher. The variation is, in fact, quite high, and the average jitter follows the average delay quite well (not shown). Thus, in real applications, jitter buffers should be used to avoid application level packet loss, but the drawback is that the effective delay becomes longer.

From the delay figure it is seen that, e.g., for 802.11, 5 VoIP sessions produces already an average delay of $> 1$ s. This is unusable for conversational applications, and, thus, MOS of these high values were not drawn in Fig. 99. The used MOS value presents listening voice quality, but does not take into account conversational issues like the effect of delay. In the delay behavior it is also seen that, e.g., for BC-MAC (stationary scenario), the delay for 7 VoIP sessions is already $> 500$ ms, which most likely means that the connection is unusable for conversation. Thus, even though the listening quality would be acceptable still for 7 ses-
sessions, in practice the delay is too high. Nevertheless, 6 sessions still seem to enjoy reasonable delay.

![VoIP packet delay as a function of simultaneous active VoIP sessions.](image)

**Fig. 100. VoIP packet delay as a function of simultaneous active VoIP sessions.**

802.11 in a mobile scenario performs again better than in a stationary scenario, and in fact the delay curve of 802.11 with mobility even crosses the delay of BC-MAC (mobile scenario). However, this crossing does not happen until the delay is already near the high delay region. The delay behavior of BC-MAC under mobility by default is not optimal, as was observed in Section 4.5.2. The high delay is caused by the high number of reTx tries, but as it was found, tradeoff with packet loss and delay can be easily made just by changing the number of reTx tries. Decreasing the reTx try limit in the case of VoIP conversation would, of course, decrease the pure voice quality because of packet loss but, on the other hand, enhance the conversational quality by enhancing the response time due to lower delay. This is a nice example how MAC level parameters have direct impact also to application layer user experience.

This VoIP traffic test showed concretely that due to the advantages of BC-MAC, considerably more simultaneous VoIP sessions can be supported. BC-MAC is able to support up to 4 simultaneous VoIP sessions with good quality in the studied scenario, and even up to 6 or 7 with fair quality, while 802.11 can support
only 1 VoIP session with good quality and 2 sessions with fair quality. The test also showed an example case where the tradeoff between delay and packet loss is something that should be considered, since, e.g., the default reTx try limit of BC-MAC causes high delays in a mobile network case, hindering the use of conversational applications.
5 Implementation issues

It was shown in the performance evaluation part that the proposed methods of this thesis outperform the traditionally used methods. However, since there is typically no such thing as a free lunch, also the downsides of the proposed methods should be discussed. The main drawback of the proposed methods is that they are more complex than, e.g., the point of comparison, 802.11. Thus, we will next evaluate what would be needed for implementing NSCD, BC-MAC, and BCCA.

5.1 NSCD

5.1.1 Dependencies

NSCD is principally meant for distributing spreading code information, and, thus, it can be used also more commonly for this purpose. In that sense, NSCD is not dependent on BCCA or BC-MAC. In fact, NSCD should be seen as a real method for performing the actual code distribution, which, e.g., recoding and spreading code usage optimization methods could exploit. For example, BCCA and NSCD are independent from each other, and NSCD should be seen as an additional feature to be used with BCCA in order to provide flexible code usage. However, the use of NSCD requires support from BCCA, since the basic BCCA implementation would be for fixed spreading codes only.

In MAC, it is only required that the MAC can convey (via inter-layer control messages or with a virtual field in a packet) the spreading code information from the network layer (NSCD) to the lower layer (BCCA or another mechanism using spreading code information). This is the case if the spreading codes are stored at the network layer. More modifications would be needed to MAC if the codes were stored there, but since the network layer code storing is more straightforward to implement, this is assumed here. Even though cross-layer issues are considered, it should still be kept in mind that the fewer layers the implementation of a certain feature needs modifications to, the better. Generally, it could be stated that in order to really enable flexible cross-layer design, some standard interlayer control message support should always exist in the equipments. However, since this typically is not the case, the use of NSCD would in many cases need some minor modifications to the MAC part as well.
NSCD is a method relying on routing protocol operation, and, thus, routing protocol with sufficient properties is required. Our implementation is done on AODV, but NSCD is not limited to AODV but could be easily implemented on other routing protocols as well. E.g., DSR implementation would be very straightforward, since in many ways it operates similarly to AODV. However, since every routing protocol operates slightly differently, NSCD implementation is also routing protocol dependent, and thus only AODV is considered here.

5.1.2 Implementation requirements

Since NSCD operates tightly with the routing protocol, direct modifications to the routing protocol are needed in order to provide NSCD operation. However, the logic of NSCD is quite simple, and the idea directly suits AODV control information exchange structure. Thus, in the AODV case, no extra control messages are required. Because of this, it makes no sense to estimate the computational costs, etc., since they are minimal when, e.g., compared to AODV complexity. Considering, e.g., the OPNET simulator source code, NSCD takes about 200 code lines, which is only about 3% of the ~ 6,000 lines of code of the AODV core implementation, which does not even include the additional code blocks belonging to the AODV implementation. When considering the actual implementation, a good paper is (Saha, et al. 2007), which compares computational complexities of different ways of implementing ad hoc network routing protocols, including AODV.

An extra table is needed for spreading code information. In our implementation to the simulator, we have used a separated table. From the memory usage sense, it would be better to use the existing routing table of AODV. However, since the table only needs to store the destination node’s id and spreading code, the required memory space is practically nothing compared to the capabilities of modern computers. E.g., the typical case could be that the IP address (32 bits) is used for id, and if as much as 64 bits are reserved for spreading code information, 12 B (96 bits) is needed for storing a single spreading code info entity. Thus, for \( n \) nodes, \( n \cdot 12 \) B of memory space is needed, posing no challenges on modern computers even if thousands of nodes were present. Moreover, having a need to store information of thousands of nodes is unlikely, since only the nearby node information is needed. The AODV routing table already contains several times more information than the spreading code table would need, since there are several entries, timers, etc. for each node in the table. Therefore, even though the addition of spreading code information to the routing table would save 4 B of
memory per node when compared to a separated table, it does not make much sense from the performance point of view. In this sense, more weight should be put on the way in which AODV is implemented. For example, if in the particular AODV implementation it is easy to add spreading code information to the AODV routing table, it makes sense to put it there, but if modifying the table structure is complicated, a separated table is preferred.

Obviously, NSCD does not require any extra HW, so as a conclusion, perhaps the most challenging task here is to find a suitable AODV implementation and make the required modifications there. In addition, one should bear in mind that NSCD modifies the AODV control packets, making it non-standard.

5.2 BC-MAC

5.2.1 Dependencies

When discussing BC-MAC in this thesis, it typically means BC-MAC with BCCA at the PHY. However, BC-MAC itself is an example of a random access MAC solution and is, as such, not dependent on NSCD or BCCA. Even the current simulation implementation could be used with other physical layers or upper layers by just turning off the BCCA functionality. However, since BC-MAC is designed especially for BCCA, it does not make much sense to use BC-MAC with another kind of channel access solution. Of course, if there are similar ideas to BCCA, such as dual channel access in a frequency domain, BC-MAC could suite those cases as well.

5.2.2 Implementation Requirements

There are really no special implementation requirements for BC-MAC, since it is after all very similar to IEEE 802.11 MAC. In fact, given that in BC-MAC there is no RTS/CTS handshaking and PCF mode, BC-MAC is an even simpler MAC solution than 802.11. Computational complexity is very much comparable to 802.11 operating in DCF mode without RTS/CTS. Both apply a stop-and-go ARQ system and follow the principles of np-CSMA. BC-MAC uses a random deferment period with CDo, which is not more complex than the exponential back-off + slotted deferment periods of 802.11. As known, there is already a plethora of 802.11 based devices available at cheap prices and built on different platforms.
Thus, the implementation of BC-MAC would not require anything special. In general, when considering MAC implementations, there is a good paper (Ilipoulos & Antonakopoulos 2000) on this topic, presenting a way to simplify the protocol development.

5.3 BCCA

5.3.1 Dependencies

BCCA is a channel access method (lower layer MAC solution), and, as such, it is not dependent on NSCD or BC-MAC. BCCA can be realized with different higher layer MACs, and, in fact, the first BCCA simulator implementation was not for BC-MAC, but for 802.11 MAC. However, BCCA is not independent of MAC, since the use of it requires support from MAC. At minimum, MAC needs to be able to tell which spreading code to use in which situation. The BCCA functionality is located at the lower layer MAC and PHY layer, and it does not have access to the higher layer packet addressing information. Therefore, it makes no sense to build intelligence to BCCA for the spreading code usage. Thus, upper layers need to manage the spreading codes, and when MAC hands a PDU to BCCA, it must also tell which spreading code to use for transmission. Furthermore, during the initialization (the device is turned on), MAC must be able to initialize BCCA, i.e., tell which two spreading codes are to be tracked at the receiver. The codes should not be hard coded.

BCCA needs spreading code information from MAC, but it does not care where this information is originally from. Thus, either the MAC has to have the intelligence for spreading code management, or the intelligence is in the higher layers (cf. NSCD). As a result, the minimum requirement for MAC is that it must be at least able to convey spreading code information from higher layers to BCCA.

If one wishes to keep spreading codes at PHY, it is possible to implement BCCA so that it also manages the spreading code table. However, when MAC hands a PDU to the BCCA, it must also tell BCCA the destination address of the packet. In this way, MAC does not need to know about spreading codes, but there is still a need for interlayer messaging. Alternatively, in order to make BCCA independent of MAC, BCCA should be able to decode upper layer PDUs, increasing the complexity of BCCA. In both cases, BCCA would need to do some kind
of mapping between addresses and spreading codes. In addition, the managing of spreading codes would be quite difficult. For these reasons, this way of BCCA implementation is not preferred in our work.

5.3.2 Implementation Requirements

BCCA implementation clearly requires more effort than the implementation of NSCD and BC-MAC. To start from Fig. 16, which presents an idea-level view of the possible BCCA receiver, we already see that extra HW/SW is needed when compared to 802.11, since 802.11 needs only a single receiver branch. The actual splitting into SW and HW parts depends on the implementation architecture, and, e.g., in software radio architecture (Buracchini 2000), everything after ADC would be SW. As in Fig. 16, the RF/IF front end and ADC parts could be similar to those of 802.11, but after that two separated branches in the PHY are needed. As suggested in the figure, it is perhaps possible to implement BCCA by having only the absolutely required parts in separated branches and reusing the rest of the PHY. However, this would require an extra buffer module, and some extra logic would be needed for separating the messages from each other. The buffering causes delays, and it is hard to estimate what kind of logic is needed to make the receiver functional in reality. Thus, it might be that the drawbacks of this solution weigh more than what would be saved in the components, so it is perhaps wisest to implement the whole PHY after ADC separately for both channels.

In addition to extra HW/SW, the possibility for spreading code management is needed. This is different to 802.11 DSSS, where a single Barker sequence is always used, and thus it can be hard coded to the devices. In BCCA, the stations have their own R-code, which should not be hard coded, but rather it will be set to the device (by a management SW directly or via the MAC layer) after the device is turned on, or it is stored to a flash-like reprogrammable memory. Also, the C-code should be freely alterable to allow flexible code usage and future possibilities. In addition, the code management should be so straightforward that the codes (both R and C) can be changed quickly as needed. This requirement comes from the spreading code usage optimization option, which is definitely something that would be sensible to utilize if BCCA was used more widely in denser networks and/or the spreading code family size was limited. After all, NSCD already provides a mechanism for easy and dynamic spreading code distribution, so capabilities for code usage optimization should exist also in the BCCA implementation.
On the transmitter side, the spreading code changing capability is very important, since it is required that the spreading code has to be alterable on the transmission basis. Thus, a quick and straightforward way of changing the transmission spreading code should be implemented to BCCA. Without this exception, the transmitter is very similar to the 802.11 transmitter.

When compared to 802.11, it could be approximated that two times the HW/SW of 802.11 DSSS PHY is needed for implementing a BCCA PHY. This is of course a pessimistic approach, since the changes in the transmitter are not that big, and, even though two receiver branches are needed, the RF/IF parts in the receiver can be shared. 802.11 based technology is nowadays very mature, and for example a single-chip 802.11 b/g solution AR2417 by Atheros Communications, Inc., costs less than 5 $ (Atheros Communications 2007). Furthermore, it should also be pointed out that already 802.11b including CCK (Complementary Core Keying) to allow 5.5 Mbit/s and 11 Mbit/s PHY bit rates is far more complex than the basic 802.11. In fact, almost 3 times more computational effort is required for CCK demodulation than for DBPSK/QPSK demodulation (Matús, et al. 2004). Hence, implementing BCCA should not be up to the technology or the cost.

When discussing the PHY complexity, it should be recalled that complexity is comparable to 802.11 only when the same chip rate (11 Mchip/s) is used. The HW requirements and costs rise as a function of chip rate, since this is tied directly to the need of which clock frequency the HW must function in. And, as known, the faster the HW is, the more it costs. Higher chip rates also need wider bandwidth from ADC, which typically has been a limiting factor in, e.g., software radio design. ADC capability is measured in sampling resolution (bits per sample) and sampling rate. The resolution should be high enough to satisfy dynamic range requirements and to minimize the quantization noise effect, and the sampling rate should satisfy at least the Nyquist criteria (“required sampling frequency” ≥ 2×“highest desired frequency to be sampled”). However, for more efficient demodulator synchronization, higher sampling rates are typically used. For example, in (Emira et al. 2006), 802.11b implementation is presented where 44 MHz sampling (instead of 22 MHz Nyquist frequency) and an 8-bit resolution are used. The paper (Emira et al. 2006) also provides interesting comparisons on what effects the sampling resolution and sampling rate have on the BER of the system. It is seen that 802.11b can be implemented with 22 MHz sampling, but the BER behavior is considerably worse than with 44 MHz sampling.
As a result, it can be stated that BCCA is easily implementable when considering similar chip rates as in 802.11. However, if a high chip rate system is desired, the costs will naturally rise. For example, devices with a chip rate of 63 Mchip/s (used mainly in this work) would be implementable, but the costs would most likely be considerably higher than in 11 Mchip/s implementation. Comparing, e.g., ADC prices by Intersil Corporation (Intersil 2007), we find that an 8-bit 160 MHz ADC chip, which would be capable of 63 Mchip/s implementation, costs over six times more than an 8-bit 35 MHz ADC chip (enough for 11 Mchip/s implementation). However, as it was seen in the analysis part of this thesis, BCCA works very well also with 11 Mchip/s. Thus, for real implementation, one needs to weigh up the pros & cons and needs. For example, in a military environment, a wide band would be needed for enhancing LPI/LPD properties and maximizing the performance, while in civilian usage, a narrower band would be good enough.

5.3.3 Frequency band considerations

In this work, the use of 2.4 GHz ISM band has been assumed, which, defined by ITU-R, covers the band 2400 MHz–2500 MHz. In the use of 802.11 (b/g), the band is divided into 14 channels separated by 5 MHz covering central frequencies in the range of 2412 MHz–2482 MHz (Santamaría & López-Hernández 2001). This applies to most of the world, but in some countries the regulations differ slightly (e.g., in Europe 2.4000–2.4835 GHz band is allocated to ISM-like use). The 11 Mchip/s signal occupies a bandwidth of about 22 MHz (power level < 30 dB from the central frequency, (IEEE 1999)), thus making the adjacent channels overlap. Because of the processing gain, the overlap does not usually matter, but naturally it increases MAI and weakens the performance. Also, a strong signal in adjacent channel can make CSMA to block transmissions, enhancing the exposed terminal problem. The only difference between the BCCA signal and 802.11 is that the spreading code in the transmission is different. This does not have a considerable effect on the spectral properties (shape of the frequency domain signal, etc.), and, thus, requirements/specifications for 802.11 apply also to BCCA. Also, since ISM bands are license free, there would be no restrictions in the use of BCCA. Moreover, there would be no difficulties in the coexistence of 802.11 signals and BCCA. Interference between 802.11 and BCCA is similar as between two 802.11 systems. Therefore, in adjacent channels, there will be some interference weakening the performance, but if far enough channels are used (> 11 MHz
difference), the interference is meaningless. In addition, since the Barker sequence of 802.11 is not likely used even as the C-code in BCCA, 802.11 and BCCA could basically coexist also in the same channel. 802.11 and BCCA would not see each other, but MAI would of course weaken the performance.

Like in the HW consideration, it must be also remembered that the above discussion applies to an 11 Mchip/s signal only. If higher chip rates are used, the occupied bandwidth naturally also increases. A good approximate rule for the required RF bandwidth is $2 \times r_{\text{chip}}$. In the ISM band usage, it is recommended that SS systems are used, but the channel structure of 802.11 is only for the use of 802.11 and is not an ISM usage requirement. Thus, in ISM, devices occupying wider bandwidths can be freely used. However, as there is only 100 MHz totally available in a 2.4 GHz band, it runs out quickly for very wideband devices. Considering, e.g., the chip rate of 63 Mchip/s, this kind of signal could be perhaps just fitted to the ISM band with strict band pass filter design, but in any case it would occupy the whole band. Strict band pass filtering, however, might distort the signal too much, in which case, a wider available band would be needed. But, as shown in Chapter 4.7, BCCA works well also with lesser spreading gain, so fitting the BCCA device to the ISM band by dropping the chip rate is not a problem.

Since there are strict transmission power requirements in the ISM band, the transmission power should be the same regardless of the bandwidth. Consequently, the 63 Mchip/s signal would have a spectral power density of only about 17% of that of an 11 Mchip/s signal, leading to a considerably easier interference scenario than in the coexistence of two 11 Mchip/s signals in the same band. The more spreading there is, the more simultaneous users there can be in the same channel (when using CDMA), and, thus, even though the 63 Mchip/s signal would allocate the whole ISM band (and perhaps even more), there can still be other (potentially several) users in the same band. In this context, one can also understand that even though, e.g., a 1 Mbit/s connection occupying a > 100 MHz band sounds inefficient, it is not the whole truth, since a single user does not use all the resources of the channel and there can be several users existing simultaneously in the same band. Therefore, one should investigate the bandwidth efficiency by taking into consideration the whole network, not only a single user. Fig. 101 illustrates the use of the 2.4 GHz frequency band.

The conclusion is the same as in the implementation requirements: If an 802.11-like 11 Mchip/s signal is used in BCCA, there are no difficulties when operating in the 2.4 GHz ISM band. From the implementation perspective, BCCA would in practice be a digital baseband implementation, and, accordingly, the
RF/IF part could be changed based on the desired frequency band. Thus, also other ISM bands like a 5.8 GHz band could be considered. However, if wider bandwidths are desired, the free ISM bands might not be enough. Again, as a result, the requirements are based on the needs. A totally different question is the use of other, licensed, frequency bands. There are strict rules regarding licensed frequency bands describing what kind of devices are allowed and who is allowed to use them. But of course, e.g., military and public authorities typically have their own bands, where wideband BCCA-like devices could perhaps operate. If lower frequencies are used, the signal range would become enhanced, but the RF-HW would be more difficult and expensive to implement. Also, it might be difficult to find wide enough free frequency bands. At higher frequencies, free bands are easier to find, but the signal range will become shortened.

Fig. 101. A frequency band view of a WLAN-like signal (11 Mchip/s) vs. two wideband 63 Mchip/s signals in the 2.4 GHz ISM band.

5.3.4 Architectural considerations

Next, architectural considerations of the BCCA transceiver are discussed. It should be noted that the idea of this section is not to present a real implementation architecture, but rather to consider what kinds of blocks would be required on the transmitter and receiver sides and, moreover, what would be different when compared to 802.11 implementations.

To consider how the possible BCCA transceiver structure could look, let us first consider the 802.11 DSSS transceiver structure, since when comparing to this, it is easy to get an idea of how complex it would be to implement BCCA. In (Chang & Sunwoo 1998), 802.11 DSSS modem ASIC (Application Specific Integrated Circuit) chip implementation is presented. Based on this, the transmitter architecture of 802.11 is shown in Fig. 102. In the architecture, the first block
performs CRC encoding and data scrambling. Next, the data is split into in-phase ($I$) and quadrature ($Q$) parts for QPSK modulation (in BPSK, only the $I$ part is in use). After this, the data is differentially encoded for DBPSK or DQPSK modulations, followed by the Spreader, which effectively multiplies the data signal by the spreading signal $c(c)$ (fixed Barker sequence) generated by the Sequence Generator. After the Spreader, the chip-stream is converted to an analog signal (DAC), which is led to the IF/RF parts.

The receiver structure of 802.11 is presented in Fig. 103 (based on (Chang & Sunwoo 1998)). After the RF/IF parts, the analog signal is converted to a digital baseband signal (ADC). The Matched Filter de-spreads the signal by correlating the input signal and the Barker sequence. Next, differential decoding is removed assisted by AFC (Automatic Frequency Control), which is used to remove the carrier frequency offset. Finally, the signal is descrambled and CRC is checked. An essential part of the receiver is also the Timing Recovery block, which is responsible for receiver synchronization by acquisition and tracking of the symbol position.

The presented implementation is only a common overview of one solution for 802.11 DSSS PHY implementation, but it gives a good idea about the basic components required, and from this it is easy to move to BCCA implementation.
A possible solution for BCCA capable transmitter architecture is shown in Fig. 104. In the figure, the new blocks or blocks needing modifications over 802.11 are shown with a different color. As seen, as the first block in PHY from the MAC side, there is a new block Spreading Code Interpreter. The function of this is to read the transmission spreading code information from the PDU when it arrives as a virtual field and remove the virtual field before releasing the PDU further for transmission. Alternatively, if inter-layer control messaging is used, the block receives the spreading code information in a message, e.g., just before it starts to receive the corresponding PDU from MAC. In either case, the Spreading Code Interpreter guides the Sequence Generator for spreading code usage. The Sequence Generator is modified so that it is controllable rather than fixed, i.e., based on spreading code information it can generate the requested sequences \( c_2(c) \) for spreading the transmission with the given code.

![Fig. 104. A possible BCCA transmitter architecture.](image)

The possible BCCA receiver architecture is shown in Fig. 105. At first glance, it is already seen that considerably more modifications are needed at the receiver side than at the transmitter side, as was expected. Closer examination reveals that the C-Code branch is almost the same as in 802.11. The only difference is that instead of having a fixed Matched Filter, there is now a need for a controllable Matched Filter, where the tracked spreading code can be initialized to the desired one, and perhaps changed during operation (future use of the possible spreading code usage optimization). The new R-Code branch is identical to the C-Code branch, but, of course, it is used to track different spreading code. The RF/IF parts and ADC are common for both branches.

For spreading code initialization and possible dynamic management, a Spreading Code Manager block is also needed. The spreading code information is
obtained by inter-layer control messages, or similar, directly from other communicating layers or via some management system. In fact, the real pieces of equipment typically have management entities (or a “management layer”), which could also handle the spreading code management, meaning that there would not necessarily need to be a separated block for that.

Fig. 105. A possible BCCA receiver architecture.

As a result, it is seen that modifications to the transmitter side are quite simple, as only one extra block is needed plus modifications to the Sequence Generator. Furthermore, the blocks/modifications are quite simple when compared to the existing blocks in the 802.11 transmitter. On the receiver side, about twice the amount of HW/SW is needed when compared to the 802.11 receiver. The required extra blocks are, however, not especially complex, since they are very much similar to those already used in 802.11. In addition there is no need to double the ADC and RF/IF blocks, which are likely to be the most expensive parts.
5.4 On the energy efficiency

As stated earlier, energy consumption as such is not included in the performance study part. However, the topic can be briefly looked into, as it is important nowadays.

By evaluating the architectural considerations shown in the previous section, it can be observed that the amount of HW/SW needed in the implementation of BCCA is about twice the amount of that needed in implementing the lower parts of 802.11. Thus, it could be very roughly estimated that running BCCA would take about twice the energy needed in running basic 802.11 (in reality it might be even less). Based on this, one could draw conclusions that 802.11 is better in terms of energy efficiency. However, this is far from the whole truth, since when discussing efficiency, what one will get when using extra energy must be taken into account.

If we take a look again at the performance results presented, e.g., in Section 4.5.1, we recall that BC-MAC allowed the network to handle 20 times more traffic than 802.11. Hence, spending two times the energy of 802.11 leads to a 20 times better performance in that scenario. This does not sound like a bad deal. Moreover, if we take a look at Fig. 41, we see that with an increasing traffic load, the total network control load of 802.11 rises rapidly as a function of the offered traffic load when compared to BC-MAC, reflecting almost directly to the energy consumption. Of course, this too is not the whole truth, since, e.g., under very low traffic loads, BC-MAC’s network control load equals that of 802.11, but listening to two channels of BCCA when compared to one of 802.11 certainly does raise the cost of transmitted data. On the other hand, several MAC methods have been developed for low traffic load situations, where sleep mode is used to save energy (e.g., for sensor networking: Choi, et al. 2007), and, thus, the cost per bit in terms of energy consumption becomes potentially considerably lower. Another issue of course is whether these methods, designed mainly for sensor networking, are suitable for data networks.

As a result, we see that there are several ways of looking at the energy efficiency matter, and in reality there are also several things which affect it in addition to the amount of HW/SW. However, quick reasoning and the high performance of BCCA when compared to 802.11 suggests that energy efficiency is very likely not the stumbling block of BCCA. Nonetheless, an accurate study should be carried out on this topic in order to reveal the real features of BCCA’s and BC-MAC’s energy efficiency.
5.5 Conclusions regarding the implementation complexity

As seen, NSCD and pure BC-MAC are fairly simple to implement, and the actual need for extra computational power or memory, as compared to the basic ad hoc implementations, is minor. Also, the implementations can be done with SW.

BCCA needs modifications to the physical layer and some extra HW/SW, making it more complicated to implement. However, the HW needed is much similar to that used in 802.11, being cheap and mature technology. BCCA could also be used in the same frequency band with 802.11. A proposal for BCCA receiver and transmitter architectures was also presented.

Nevertheless, if a higher spreading factor is desired by increasing the chip rate, the implementation would require faster HW. HW today still has limited capabilities and, e.g., 63 Mchip/s devices would most likely be expensive to implement. However, the required spreading factor is up to the application: As it was seen in the performance analysis part, BCCA with the standard 11 Mchip/s spreading gave very good performance, suiting civilian and commercial applications well. The performance difference to 63 Mchip/s spreading is not big, but in special environments (e.g., military) there might be needs to increase the LPD and LPI properties of the signal, encouraging higher spreading factors.

If there really is willingness and a need for enabling high performance ad hoc networking, methods like NSCD, BC-MAC, and BCCA are well feasible. In the real implementation, probably the most challenging thing is that 802.11 has reached a worldwide de facto position for WLANs and for public ad hoc networks, and introducing new, non-standard, solutions might be tricky. However, ad hoc networks will probably be applied mainly for special cases like military or public authority worlds, where new systems are easier to be introduced.
6 Conclusions and Future Work

6.1 Summary and discussion

In this work, BC-MAC vs. 802.11 was studied in various different scenarios with several different parameters. Next, the main results of the performed tests are listed and some discussion is provided.

In a traditional access point network, it was shown that BC-MAC is not at its best when a narrow band channel model with a serious hidden terminal problem is present. The scenario was set in a way that the hidden terminal problem is quite extreme. 802.11 RTS/CTS, on the other hand, is designed especially for this kind of operation, and, thus, it performed very well. However, with a wideband channel, BC-MAC performed almost as well as 802.11 RTS/CTS and better than pure 802.11. The 802.11 based setups gave practically the same performance regardless of the spreading factor. Hence, they work fine also in narrow band cases, unlike BC-MAC, but in contrast they fail to harness the potential of SS techniques. This test also showed the benefits of using CDo in BC-MAC, while later in more ad hoc network-like scenarios the effect of CDo was not that significant.

When there was no AP in the network, the benefits of BC-MAC began to show up even in a single-hop network. BC-MAC already gave clearly better performance than 802.11. Furthermore, when a real multi-hop ad hoc network was tested, BC-MAC really outperformed 802.11. In the used default scenario of 20 nodes in an area of 500 m × 500 m, over 20 times more traffic can be supported in the network if BC-MAC is used instead of 802.11. In a real ad hoc network scenario, the benefits of RTS/CTS mechanism are turned into disadvantages: The complex RTS/CTS mechanism turned out to act as a link status double check, thus causing a high number of route breakages for nothing. This causes a high network control load due to continuous route re-establishments, ruining the performance and saturating the packet loss to the level of ~ 0.1%, while practically zero packet loss was easily achievable with BC-MAC. Also, due to the time consuming handshaking, the delay behavior is considerably worse than with the other setups. The RTS/CTS mechanism blocks the transmissions in a large area, decreasing the spatial capacity of the network. As a result, it can be concluded that the RTS/CTS mechanism is not at its best in a flat ad hoc network environment.

In the 20-node scenario, an interesting test was also carried out by enhancing 802.11 with BCCA capabilities. BC-MAC still performed better, but it was shown
that the performance of 802.11 can be significantly enhanced by using BCCA at the physical layer. Also NSCD was tested, and it was found that the use of NSCD does not practically hinder the performance from the assumption of using known spreading codes. This means that BC-MAC with NSCD could be used also in commercial applications, since users would not have to consider the spreading code handling, making the use of BC-MAC to be practically as simple as the use of 802.11 is today.

Mobility does not change the ranking between 802.11 and BC-MAC, but instead the same difference between the points of congestion exists: BC-MAC can handle several times more traffic than 802.11. However, mobility sets a lower bound to the achievable packet loss ratio, and a zero packet loss ratio is no longer reachable. Packet loss ratios have very similar trends between all the setups under a low traffic load. Handling mobility is up to the routing protocol, and thus the basic trends are the same for all setups. Different MAC’s have some effect on the levels of the curves under a low traffic load, but the MAC design mainly helps in supporting higher traffic loads. It was also found that an interesting relation exists between the mobility and IAT of packets. This relative mobility increases with decreasing $G$, causing an increasing trend in packet loss and delay towards lower loads. Also, it was noticed that with BC-MAC (partially also with 802.11), a tradeoff between packet loss and delay can be made by changing the reTx try limit. This tradeoff is at least important in the sense that the delay tends to easily grow with BC-MAC’s default reTx try limit under high relative mobility.

A detailed performance evaluation of a single nodes’ performance in the time domain was also made. It was interestingly found that when a multi-hop ad hoc network is driven to a congested state, the delay starts to follow long tailed distributions even though the traffic models are traditional. It seems that the reason for this is the on-demand nature, since the route establishment time was found to follow long tailed models regardless of the traffic load. When the network gets congested, the number of route discoveries explodes and starts to dominate the total end-to-end data packet delay behavior. This test also showed that there are clear differences in the traffic loads in different parts of the network.

The operation of BCCA is based on CDMA. Thus, a study was carried out on what happens to the performance of BC-MAC with different spreading factors. As expected, decreasing the spreading factor decreases the performance of BC-MAC. Also, the smaller the spreading factor is, the more the use of NSCD will worsen the performance when compared to the assumption of known codes. This was also expected, since in our pessimistic assumption, NSCD chooses codes ran-
domly from the given code set, whose size is tied to $F_{sf}$. Nevertheless, BC-MAC did not seem to be too vulnerable for the $F_{sf}$ since with $F_{sf} = 11$ the performance was still very good. In addition, even with the very small spreading factor of 4, BC-MAC still clearly outperformed 802.11 even though NSCD was used, leaving only 6 spreading codes for 20 nodes. This brief study proved that the functionality of BC-MAC does not collapse even though wide SS signaling would not always be available.

Performance with different radio channel models was also studied. In the propagation modeling part, overall it was found that comparing different methods with inaccurate models might give a wrong idea of the performance. In particular, the typical cut propagation models highly overestimate the performance. The errors of the propagation modeling cumulate over multiple hops, making accurate modeling more crucial in larger networks. In addition to the 20-node default scenario, a dense 50-node network and an oblong 50-node network were also tested. It was found that the node density affects the errors caused by propagation modeling, but the shape of the scenario affects even more. Most likely, it is not only the shape which affects, but mainly the average route length caused by the shape of the network. Even though propagation modeling is far below the network layer in the communication stack, different propagation models still have quite interesting effects on the performance measured above the network layer. The most important proof considering this thesis is that BC-MAC outperforms 802.11 regardless of the propagation environment and different scenarios. BC-MAC performed very well even in the difficult FPL channel. Also, BC-MAC supported considerably better data rates per single network user in a dense 50-node network than 802.11 could support in a 20-node network. Moreover, the behavior of BC-MAC is somewhat more predictable, and it is not as sensitive to modeling errors and the propagation environment as 802.11. This part of the study also answered the question of scalability: BC-MAC does not introduce any restrictions to the scalability. However, it should also be kept in mind that even though BC-MAC uses the available spatial capacity more efficiently than the traditional methods, it cannot overcome the theoretical limit that the performance of an ad hoc network will decrease as a function of number of nodes.

Fading causes the typically quite strict radio range to be more unpredictable, and thus packets may be received far beyond the nominal radio range, while the reception may also fail even in very short links. The brief fading test showed that if fading exists in the intended ad hoc network operational environment, it should definitely not be ignored in simulation studies, since it affects the performance a
lot. Typically, the performance will get worse as a function of increasing fading intensity. However, in a sparse network, fading can also help in connectivity problems. While fading brings interesting features to the performance behavior, the main finding was that BC-MAC does not collapse even under a strong fading channel. As a result, it is quite safe to assume that the use of BC-MAC is not restricted by different radio channel conditions.

The last part of the work was to test different traffic scenarios. First, different data packet sizes were tested, and, overall, the results were quite predictable. Packet sizes have obvious effects on the performance, since the shorter the packet, the more inefficient the transmission is, leading to worst performance. However, none of the setups gave any special indication of particularly bad or good performance under any packet size. Despite the properties of the RTS/CTS mechanism, the performance of 802.11 RTS/CTS was not considerably enhanced with a long data packet. Performance was also observed as a function of MAC level traffic, and it was found that BC-MAC handles the different sized data packets fairly. This means that in practice only the traffic load affects the point of congestion, not the packet size. The main differences between different setups remained regardless of the packet size.

Different traffic models were also tested. Burstiness causes temporary high traffic loads and is connive in building up temporal and spatial congestions. However, BC-MAC is by its nature quite good at handling different traffic loads, and thus the effect of burstiness was not shown until a high traffic load, and even then the differences between traffic processes were almost negligible. 802.11, on the other hand, is known to be vulnerable to congestions and collisions, and, thus, the performance is affected more by the level of burstiness. More specifically, having basic variability (VBR-M) instead of CBR results in a clearly worse performance, but increasing burstiness beyond VBR-M in practice no longer worsens the performance. Moreover, packet length distribution has an evident effect on the packet loss performance of 802.11.

The tested different session models did not cause any thoroughly significant behaviors. Surprisingly, on average the SC model seems to be harder for the network to handle than DC (with both, 802.11 and BC-MAC). This, however, is caused by a few badly behaving scenarios, while the rest give better performance than DC. FNC, which unbalances the flat ad hoc network behavior, typically led to worse performance than DC. However, 802.11 can also benefit from the partially centralized network behavior under FNC, and thus the packet loss performance under FNC was close to that of DC under a high traffic load. BC-MAC gave
a steadily worse performance under FNC models than under DC, but the difference was quite small, and, overall, BC-MAC was able to handle also the most difficult FNC-100 case very well, and the performance gap to 802.11 remained clear.

Finally, the performance of a more concrete application, real-time VoIP conversation, was studied. It was shown with calculated MOS values that due to the advantages of BC-MAC, clearly more simultaneous VoIP sessions can be supported than with 802.11. In the default 20-node scenario, BC-MAC is able to support up to 4 simultaneous VoIP sessions with good quality, and even up to 6 or 7 with fair quality, while 802.11 can support only 1 VoIP session with good quality, and 2 sessions with fair quality.

Overall, the work clearly pointed out the benefits that can be attained by lower layer design in ad hoc networks. The study showed from several aspects the favorable properties of the proposed methods. Designing the lower layers particularly for ad hoc networking in a cross layer fashion is the key to reaching feasible ad hoc networking solutions. There is a need for this, since the performance of the current solutions collapse quickly with an increasing traffic load and hop count. The channel access mechanism is the worst bottleneck of the traditional 802.11. The drawback of the proposed methods is that they are more complex than 802.11. However, as discussed in the implementation issues Chapter, the difference to 802.11 is not considerable, and with modern HW the implementation of BCCA, which is the most complex part, is not a problem. Thus, if there is willingness, the proposed BC-MAC/BCCA with NSCD could be a potential approach for future ad hoc networks.

6.2 Potential future work

Besides the interesting results, the work for this thesis revealed several points where future research work could enhance BC-MAC’s performance further. Also, some things were found as regards how to enhance the performance of ad hoc networking in general.

In this thesis, it was found that BC-MAC’s performance behavior can be influenced by modifying the carrier sensing method. Especially in a single-hop environment this has a major effect, and, e.g., applying reduced range PSD detection enhances the performance considerably over the typically used PD. In a multi-hop case, however, the improvement was not that evident. Nevertheless, in this regard only the surface has so far been scratched, and further research should be
carried out. After all, BC-MAC design involves a difficult conflict: Simultaneous transmissions should be encouraged, but on the other hand harmful collisions should be prevented. The key to proceed in this is the carrier sensing mechanism.

There are several parameters which could be optimized, and one of them is the reTx try limit. This is important not only for BC-MAC, but also for other ad hoc network MACs in general, while of course there are also some differences between MACs in the way they use the ACKs. It was shown in this work that the reTx try limit has a considerable effect on the performance with both BC-MAC and 802.11. E.g., the reTx try limit affects the tradeoff between packet loss and delay, which is a cause of tradeoff between link breakage detection efficiency and speed. Also, the reTx try count under a fading channel and relative mobility causes a complicated interdependence affecting the performance. Thus, it is quite obvious that an adaptive reTx try limit might be the best solution, but a lot of research is required on how to do this in an efficient and robust way.

Deferment period optimization is another thing to be considered. By now, as a result of straightforward test runs, e.g., the randomness of the deferment period is set to 5%, and the CDo probability is set to 0.3, but no accurate research has been carried out. It is likely that these parameters are not the optimal ones, so what would be the optimal parameters and how would one get them? Also, how much will these parameters affect the performance? According to the tests carried out in the design phase of BC-MAC, it was found that the effect of them is not major, perhaps only a few percentages of the overall performance. However, more accurate research would be needed. The deferment period can be also used for taking into account priority issues, which should be also studied, since QoS support is becoming essential in modern communications.

In this work, the way of simulating was explained thoroughly, and it was found that there are several ways to do it, of which, one was mainly used. Simulation details could be further enhanced. Especially if at some point there was interest to implement BCCA functionalities, it would be important to get more accurate simulation results before that. One of the places which could be simulated more accurately is the physical layer. In this work, the properties of PHY signals have been mainly modeled statistically, which is adequate at this stage, but it would be interesting in the future to build accurate PHY models, where, e.g., real spreading code families and their properties could be tested in the context of a BCCA-based network. Also, the effect of channel models, channel coding, etc. would bring new information on the performance.
The ACK system of BC-MAC currently uses instant ACKs, which gives good performance. However, the protocol is still a stop-and-wait protocol, while more efficient protocols are available. It might be that the more complex protocols do not work properly because of the dynamic nature of ad hoc networking, but this should be researched. Also, data/ACK piggy-packing is something that could improve the performance. In addition to this, modern WLAN MAC standards contain methods which are not available in 802.11 (e.g., frame aggregation, block ACKs), and their performance would be interesting to examine. There are interesting topics like how adaptive modulation and high link data rates would actually perform in an ad hoc environment, what are the tradeoffs between link data rates and spreading vs. total network performance, and how to use them optimally with BCCA. BCCA functionality could be also built on top of a MC-CDMA PHY, enabling easier link adaptation, but how to do that in practice? Also the MESH standard 802.11s and its performance would be an interesting point of comparison, and, also, could BCCA be used with 802.11s?

There is optimization work also at the AODV side. AODV AART is known to be an important parameter, and thus its effects were studied briefly also in this work. The study showed that AART affects the performance a lot, and it typically enhances the performance in a mobile network. However, setting this parameter incorrectly can lead to considerably worse performance. E.g., in a stationary network, AART only ruins the performance. This is one of the main reasons why AART is set to infinity in most parts of this thesis. AART should be taken into account when planning a real ad hoc network, and, clearly, adaptive AART would be the best choice.

NODE_TRAVERSAL_TIME is another parameter in AODV that affects the performance and should be investigated in more detail. Also, when optimizing the whole ad hoc network performance, other routing protocols could be considered. In this work only AODV is considered, but BC-MAC is not tied only to AODV. It is possible that BC-MAC would work even better with some other routing protocol. At least it is known that since AODV is meant for flat ad hoc networks, it does not work well in very large networks.

When considering ad hoc networks’ fundamental problems, BC-MAC cannot solve the problem of connectivity. Thus, new physical layer techniques like adaptive antennas could be used to avoid disconnected networks. With adaptive antennas, the spatial capacity of ad hoc networks can also be increased, since tighter antenna lobes restrict the interference caused to and by other users. In fact, adap-
tive antenna approach has already been taken to extend BC-MAC, and very promising results have been got (Bräysy et al. 2006).

The BCCA method enables the use of CDMA, and currently, in a single transmission, a single frame can be transmitted to a single destination. However, CDMA enables a possibility of summing transmissions of several spreading codes and transmitting them simultaneously to multiple users. Thus, a single node could deliver different packets to several neighbors with a single transmission. Basically, this sounds like the capacity would be perhaps multiplied. This is perhaps true in some situations, but the implementation is far from trivial, since, e.g., summing several spreading codes typically causes MAI, weakening the possibilities for reception. Thus, the sending node should somehow know whether all the intended destination nodes will be able to receive their packets from the summed signal, i.e., is the SINR high enough for all the destination nodes. The use of orthogonal codes would prevent MAI, but the code family size would be rather limited. Another issue is the ARQ method, since if a single node will transmit to multiple nodes, in the current system all the destination nodes will send an ACK immediately back to the source node causing collisions where most of the ACKs (if not all) are lost. This would cause the multi-destination transmission to give a worse performance than the current single-destination transmission. Thus, the ACK sending policy should be somehow modified, e.g., perhaps scheduled between the multiple destination nodes, but this could be quite complicated. In any case, simultaneous multi-destination transmission could be used quite easily at least in multi-cast cases where ACKs are not used.

One quite obvious addition to BC-MAC is power control. Currently, no power control exists, and, thus, it is quite likely that enabling this would enhance the performance perhaps notably, since DS-CDMA is an interference limited method, and spatial capacity is important in ad hoc networking. However, in an ad hoc networking world, power control is not trivial, since communication occurs between several nodes. Thus, fast closed loop power control, typically used in cellular networks, is practically not implementable in a real ad hoc network. Slow closed loop power control should work but it requires extra messaging. However, by using cross-layer thinking, the power control information could be, perhaps, coded, e.g., to data and the following ACK packets, in which case no extra messages would be needed. The ad hoc environment causes problems to this too, since nodes have multiple dynamic neighbors. Thus, a node could perhaps have a continuously updated power control table containing information for all the neighboring nodes. The power management should be quite intelligent in order to
quickly notice node movement and broken links. There are several pitfalls in this. For example, a rule can be easily made that if ACK is not received, it can be assumed that the destination node has moved too far, and thus the Tx power needs to be increased. However, if the reason for packet loss was not in a long link but network congestion, all the nodes in the congested area will try to increase the transmission power, spreading the interference and the effect of congestion to a larger area. Nonetheless, it can be assumed that power control would increase the performance significantly, making it worth researching.
References


234


235


Turner JS (1986) New Directions in Communications (or Which Way to the Information Age). In IEEE Communications Magazine 24: 8–15.


Appendix 1 Spread Spectrum Communications

The three basic SS methods are time hopping, frequency hopping (FH) and direct sequence (DS). The latter is of interest in this work, so it will be introduced briefly.

In DSSS, a high rate signal is used to modulate the actual data signal prior to the transmission. If digital signal processing is used, the modulating signal is a sequence of bits (called chips) with a finite length (a spreading code). The chip rate ($r_{\text{chip}}$) of the spreading code is higher than the rate of the data ($r_d$), and this difference is called the spreading factor, since the modulation will spread the data signal in the frequency domain according to the chip rate. Thus, the spreading factor is

$$F_{sf} = \frac{r_{\text{chip}}}{r_d}. \quad (42)$$

In this work, a signal is defined as a spread spectrum signal simply when $F_{sf} > 1$.

At the receiver, the wideband signal will be de-spread using the same spreading code. As a result, the original data signal is recovered. De-spreading can be done with a matched filter or correlator, both of which effectively multiply the received signal with the spreading code in the correct phase. The reception requires accurate receiver synchronization to the received signal. Synchronization is its own form of art, and in this study, synchronization errors are not modeled.

The spreading code is usually a pseudo-random code, which is generated with a certain algorithm. The length (cycle) can vary from a few chips to as long as needed. Often bit-wise spreading is used, where the spreading code length is the same as the duration of a single data bit. As a result, the whole spreading code is used to spread one data bit and the spreading factor will be directly the code length $N_c$. This is the method used also in 802.11 and in this study.

With SS principles, the channel capacity can be used efficiently. The effect of spreading is that, in addition to the signal spread, the power density also drops to as divided by $F_{sf}$. Thus, there can exist a narrowband signal(s) in the same frequency band with the wideband signal. At the receiver, the narrowband signal will be spread because of the multiplication by the spreading code. The result is that the narrow band signal will be shown as interference with a power level that is dropped to the fraction of $F_{sf}$ from the level of the original narrow band signal. Hence, the SS principle introduces processing gain $G_p$, which equals to $F_{sf}$, if the interfering signal and signal of interest are uncorrelated.
Processing gain is the key idea of SS communications, since it is also present with wideband signals. This enables Code Division Multiple Access (CDMA), where different users use different spreading codes to communicate on the same frequency band at the same time and in the same space. Hence, in pure CDMA, users are free to transmit at any given time, and the whole bandwidth is available to them. CDMA is interference limited rather than bandwidth or dimension limited like FDMA (Frequency Division Multiple Access) and TDMA (Time Division Multiple Access). If third party sources of interference are ignored, the performance of the CDMA network is dependent on the level of Multiple Access Interference (MAI), which is caused by the transmissions of other users in the network.

By proper code design, MAI can be significantly reduced, and there are also spreading code families, where the codes are orthogonal and, thus, do not cause MAI in the ideal case. Orthogonal codes, however, usually require exact timing (code phase), etc. to retain the orthogonality, and also the size of these families is usually small. Hence, orthogonal codes might not be the best choice for ad hoc networks, but rather one might choose a code family with well-controlled correlation properties regardless of the code phases. A good source for the art of spreading code design is provided, e.g., in (Kärkkäinen 1996).

Yet another benefit of SS communications is that it enhances the capture effect. When the receiver is locked (synchronized and receiving data) to the desired signal, and another transmission with the same code arrives, it experiences a similar effect as receiving a transmission with a different code. This is because the newly arriving transmission is likely to be out-of-phase to the signal under reception and will, therefore, be considered to be noise from the perspective of the signal under reception. It will then be up to the autocorrelation properties of the codes and link budget whether the MAI is strong enough to corrupt the ongoing reception. Of course, there is the possibility that the two colliding transmissions will arrive within the same code phase, in which case it is more likely that both transmissions will be lost. However, the probability for this is low if the code is long. As a conclusion, the use of SS techniques will significantly reduce the effect of collisions. The capture effect can be even used as a multiple access scheme (e.g., spread ALOHA multiple access, (Abramson 1994)).

There is also a principle called MCSS (multi-carrier spread spectrum), in which the chips are sent with narrowband orthogonal carriers. This method is related to the popular OFDM (Orthogonal Frequency Division Multiplexing) method. MCSS can be thought as a frequency division version of DSSS, and it has
some favorable properties over DSSS. However, since the interest regarding this work is on DSSS, MCSS is not presented in more detail. A good overview on multi-carrier CDMA is provided, e.g., in (Prasad & Hara 1996).

More detailed information on SS and CDMA is given, e.g., in (Peterson et al. 1995, Glisic & Vucetic 1997, Meel 1999).


294. Gore, Amol (2008) Exploring the competitive advantage through ERP systems. From implementation to applications in agile networks


298. Rabbachin, Alberto (2008) Low complexity UWB receivers with ranging capabilities


304. Popov, Alexey (2008) TiO2 nanoparticles as UV protectors in skin


Jarmo Prokkola

ENHANCING
THE PERFORMANCE OF
AD HOC NETWORKING BY
LOWER LAYER DESIGN